**A REPORT  
ON**

**Deep Learning Algorithms to predict particulate matter concentration over India**

*By*

| **Name of the Students** | **ID No.s** |
| --- | --- |
| Anish Ashish Kasegaonkar | 2020B3A70785P |
| Akshat Johar | 2020A3PS1793H |

***Prepared in the partial fulfilment of the***

BITS F221: Practice School-I Course

**AT**

**Indian Institute of Remote Sensing (IIRS), Dehradun**

A **Practice School-I Station**

of

**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI**

May - July 2022

-

**Acknowledgement**

We would like to use this opportunity to express our sincere gratitude to our industry mentor Dr. Yogesh Kant (Scientist) for his constant support, invaluable guidance and illuminating views on issues throughout the course of the project till now.

We would also like to thank our PS instructor Dr. Rekha Anandrao for her support, comments and constructive suggestions during the project work.

We express our sincere thanks to the Practice School Division, Birla Institute of Technology and Science, Pilani for giving us this opportunity to work on a project at the distinguished Indian Institute of Remote Sensing (IIRS). Finally, we are sincerely grateful to the prestigious institute BITS Pilani, for giving us this opportunity to gain industrial knowledge during Practice School-I.

**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI**

**Practice School Division**

**Station:** Indian Institute of Remote Sensing (IIRS) **Centre:** Dehradun

**Duration:** 7 weeks **Date of Start:** 28th May, 2022

**Date of Submission:** 20th July, 2022

**Title of the Project:** “Development Of Deep Learning Algorithms To Predict Particulate Matter Concentration over India”.

**Names of the:** Anish Ashish Kasegaonkar (2020B3A70785P)

**Students** Akshat Johar (2020A3PS1793H)

**Name and:** Dr. Yogesh Kant, Scientist/Engineer, Department of Marine and

**Designation** Atmospheric Science, IIRS Dehradun

**of the expert**

**Name(s) of the:** Dr. Rekha Anandrao, Department of Electrical and Electronics

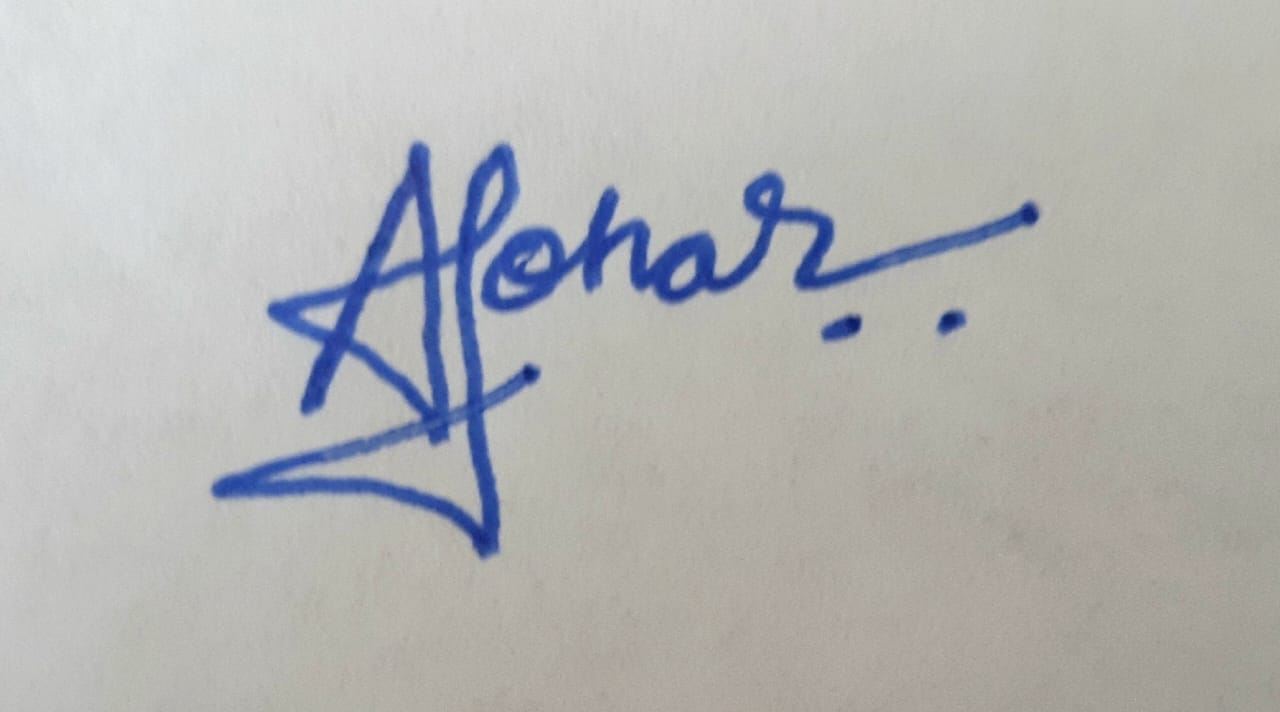
**PS Faculty** Engineering, BITS Pilani

**Project Area:** IT Domain

**Project Link :** [Google Collab Link](https://colab.research.google.com/drive/19lGUNLUgG4eKQKMCoFp1p5u-m6eMx08p?usp=sharing)

**Abstract:** Prediction of PM concentration nowadays is essential for alarming the public well in time when its level rises beyond optimum values to prevent the growth of respiratory diseases. In this work , the PM concentrations are predicted using time series prediction done with the help of LSTM neural networks. The input parameters fed into the two models are Aerosol optical depth (AOD) , Meteorological Data and PM 2.5 and PM 10 levels of different stations across the nation. The 2020 data is used as the training data whereas the 2021 data is used as the testing data. With the help of statistical analysis , the model is chosen which has the least amount of error and highest accuracy.

**Keywords:** Machine Learning , Deep Learning , LSTM

Signature(s) of Student(s) Signature of PS Faculty

**Index**

**Contents Page No.**

1. Introduction…………………………………………………………….5
   1. Objectives……………………………………………………….5
   2. Significance……………………………………………………..5
   3. Literature Review……………………………………………….6
   4. About IIRS……………………………………………………..11
2. Main Text………………………………………………………………12
   1. Technical Requirements…………………………………………12
3. Methodology………………...…………………………………………..16
   1. Approach……………………………...………………………….16
   2. Dataset Creation………...………………………………………..18
   3. Data Visualisation………….……………………………………..22
   4. Model Creation…………………………………………………...24
4. Results and Inference……………………………………………………28
5. Conclusion and Future Scope…………………………………………....30
6. References……………………………………………………………….31

**Introduction**

**Objectives**

* To develop various deep learning algorithms such as Long Short-term Memory (LSTM) to predict Particulate Matter concentration over India.
* To use statistical methods to test the accuracy of models used to predict the PM2.5 and PM10 levels.

**Significance**

Prediction of PM Levels are essential to alarm the public beforehand in order to avoid respiratory diseases caused by excess concentration of them in the atmosphere. The result of the work would help to determine which model works the best and has the highest accuracy for predicting PM 2.5 and PM 10 levels across the country. The obtained spatial mapping of PM 2.5 and PM 10 would help to visualise the data for further applications. Uniqueness written in the Approach Section

**Literature review**

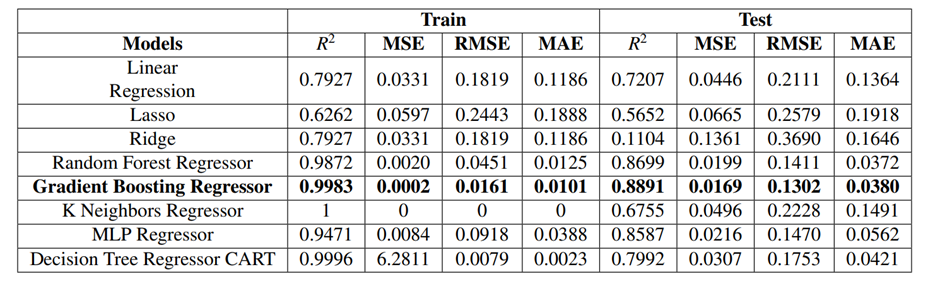
Harish and Yogesh [2] used machine learning predictive models to forecast the PM concentration in the atmosphere. Taiwan Air Quality Monitoring data sets dating from 2012 to 2017 were used for the same. They had applied various predictive models on the given dataset and compared the models on factors such as Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Square Error (MSE), and Coefficient of Determination (R2).

Dataset in hand contained both air pollution data and meteorological data which was responsible for PM concentration in the atmosphere. From the TAQMN data set, parameters were divided into two groups i.e., particular data and chronological data. Chronological data was data which was continuously changing whereas particular parameters changed with area and time. Missing values in the dataset were filled using Fourier arrangement and spline multinomial approaches. After filling the missing values, various ML models were applied to the data set and cross-validation was done to select the best algorithm.

ML Algorithms used were the following: -

* Random Forest
* Gradient Boosting Regression
* Decision Tree Regression
* MLP Regression

Following results were obtained after applying different models: -



As seen, the **gradient boosting regression algorithm** worked the best among other algorithms.

Seng and Zhou [3] designed a predictive time-series model with multi-output and multi-index of supervised learning (MMSL) based on Long Short - Term Memory (LSTM) and used it to predict concentrations of PM, CO, NO2, O3 and SO2 present in the air. The proposed model was cross-validated with other existing time series models to show its significance.

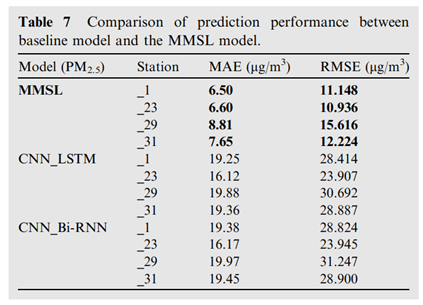
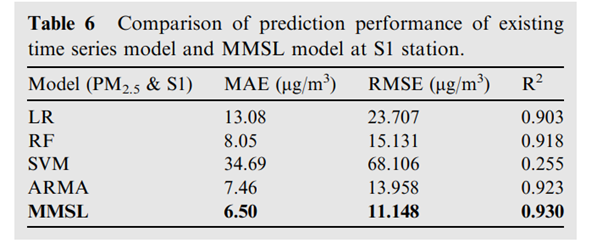
Dataset used includes PM, meteorological data and also gaseous pollutant data of mainly SO2, NO2, O3, CO from January 1, 2016 to December 31, 2017. Missing values in the dataset could be filled using different methods such as entering 0, taking mean, mode, interpolation, etc. Above methods were compared based on the RMSE values.

**Interpolation** had the least RMSE value and hence was used to fill the missing values of the dataset. For Spatiotemporal Analysis of Data, out of 35 stations, the top 5 stations

having the highest correlation were used for data collection to preserve the accuracy of the applied model.

The procedure was as follows. Firstly, the data sequence was converted into a supervised learning sequence format, from sequence to pairs of input and output sequence. D was specified as the input time step and N as the output time step and the values of t, t + 1, ..., t+N were predicted using data obtained at t-D, t-D + 1, ..., t-1. After this, the dataset was divided into training and testing dataset and LSTM parameters were initialised such as maximum epoch, the number of hidden layers and neurons, the number of fully connected layers and neurons. Lastly, the performance of the MMSL model was observed on the testing dataset and the LTSM parameters were saved for future use. The training period value of D was chosen based on the values of MAE, RMSE and R2.

The chosen model was now compared with existing models and results were as follows: -



As seen from above data , the proposed model had best performance amongst other models.

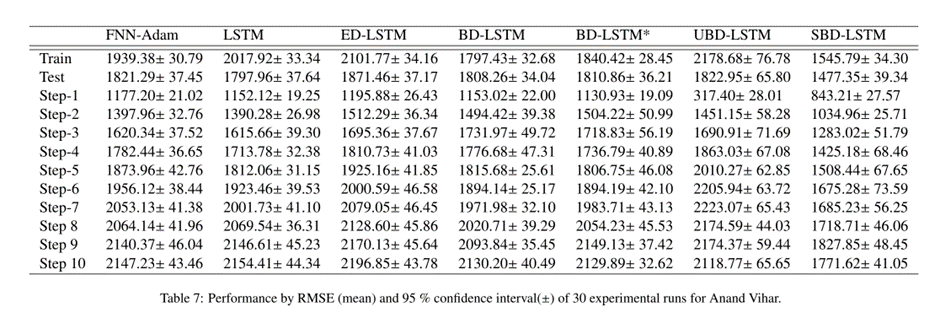
Tiwari and Rishabh [4] used Long Short-Term Memory (LSTM) neural networks models (such as bidirectional and encoder-decoder LSTMs) to predict air quality using different parameters for selected parts of Delhi, India. They also investigated the impact of COVID-19 lockdown on air-quality and the ability of the model to provide quality predictions before and during COVID-19. The prediction performance had been measured using root mean squared error (RMSE).

Dataset used consisted of 12 parameters, some of which included PM2.5, PM10, Benzene, Toluene, NO, NO2, SO2, CO, etc from 1 January, 2019 to 10 December, 2020. For this, four of most polluted areas of Delhi were chosen as monitoring stations, i.e. Anand Vihar, Bawana, DTU and Vidya Vihar. The data was sourced from the Central Pollution Control Board, India. The model used was a simple recurrent neural network (RNN) trained by back-propagation through time (BPTT), along with LSTM to prevent vanishing and exploding gradients. The following LSTM models were used:

* Bi-directional LSTM (BD-LSTM)
* Encoder-decoder LSTM (ED-LSTM)

Both multivariate (11 input nodes, 10 output nodes) and univariate (1 input node, 10 output nodes) were used. First, the impact of COVID-19 lockdown on air quality was analysed, and then different models for multi-step ahead prediction of air quality (PM2.5) were evaluated using Adam optimizer.

The results for Anand Vihar was as follows:



This showed that the **multivariate bidirectional-LSTM model** provided best predictions with better generalisation, despite COVID-19 impact on the air-quality during full and partial lockdowns, as it had the least RMSE. This result was further backed by similar trends at other monitoring stations.

**IIRS Dehradun**

The Indian Institute of Remote Sensing (IIRS) - is a constituent unit of Indian Space Research Organisation (ISRO), Department of Space, Govt. of India [1]. Since its establishment in 1966, IIRS has been a key player for training and capacity building in geospatial technology and its applications through training, education and research in Southeast Asia.. The vision of IIRS is “Achieve excellence and remain in the forefront for capacity building in Remote Sensing & Geoinformatics and their applications.”

**Main Text**

**Technical Requirements**

We were expected to have some basic prerequisite knowledge on deep learning algorithms before moving on to the project, especially about RNNs or Recurrent Neural Network. The detailed information is as follows.

1. **RNNs:**

An artificial neural network that employs sequential data or time series data is known as a **recurrent neural network** (RNN). These deep learning algorithms are included into well-known programmes like Siri, voice search, and Google Translate. They are frequently employed for ordinal or temporal issues, such as language translation, natural language processing (nlp), speech recognition, and image captioning. Recurrent neural networks (RNNs) use training data to learn, just like feedforward and convolutional neural networks (CNNs) do.

RNNs are distinguished by their "memory," which allows them to affect the current input and output by using data from previous inputs. Recurrent neural networks' outputs are dependent on the previous parts in the sequence, unlike typical deep neural networks, which presume that inputs and outputs are independent of one another. Unidirectional recurrent neural networks are unable to anticipate future occurrences, even if they would be useful in predicting how a series will turn out.

1. **LSTMs:**

An advanced RNN, or sequential network, called a **long short-term memory network**, permits information to persist. It is capable of resolving the RNN's vanishing gradient problem. *Figure 1* depicts the typical structure of a LSTM neural network.

The cell state and its multiple gates make up the fundamental idea of LSTMs. The cell state functions as a highway for the transportation of relative information throughout the entire sequence chain. The cell state might, in theory, carry important information when the sequence is processed. The effects of short-term memory are thus diminished because

even knowledge from earlier time steps might travel to later time steps. Information is added to or withdrawn from the cell state via gates as the cell state travels. The gates,

which determine which information is permitted on the cell state, are various neural networks. During training, the gates might learn what information is important to remember or disregard.

1. ***Forget gate***: This gate determines what data should be deleted or retained. The sigmoid function processes data from the previous hidden state as well as data from the current input.
2. ***Input gate:*** This gate is used to change the cell state. A sigmoid and a tanh function are used to process the current input and the prior hidden state. The tanh output is then multiplied by the sigmoid output. Which information from the tanh output should be retained will be determined by the sigmoid output.
3. ***Output gate:*** The next concealed state is decided by the output gate. The state that is used for predictions is the concealed state. To determine what data the hidden state should contain, we multiply the tanh output by the sigmoid output. The concealed state is the output. The new hidden and cell states are then transferred over to the following time step.

| ***Figure 1:*** *LSTM neural network* | | ***Figure 2:*** *GRU neural network* | |
| --- | --- | --- | --- |
| ***Legend:*** |  | | |
|  |

1. **GRUs:**

The GRU is a part of the more recent generation of recurrent neural networks and resembles an LSTM in many ways. The concealed state was utilised to transfer information by GRUs in place of the cell state. A reset gate and an update gate are the only gates it possesses.

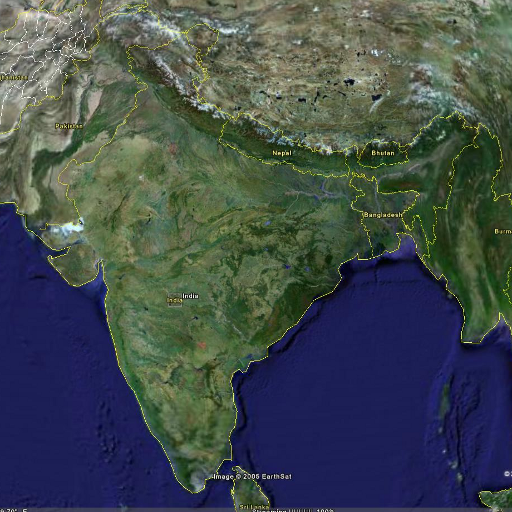
1. ***Update gate:*** The update gate functions similarly to an LSTM’s forget and input gates. It chooses what data to discard and what fresh data to include.
2. ***Reset gate:*** The reset gate is another gate used to decide how much past information to forget.

**Methodology**

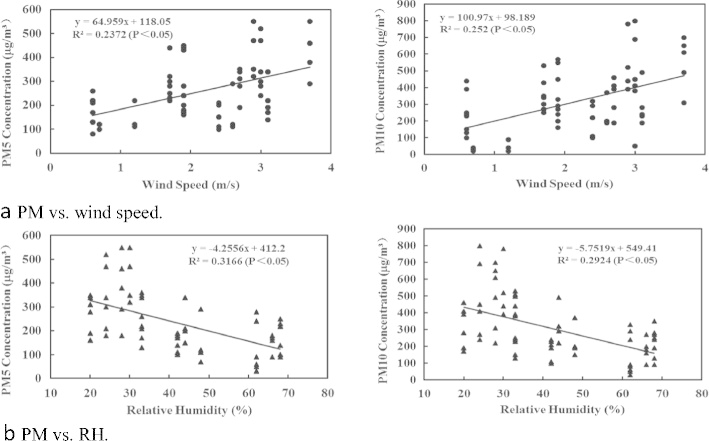
1. **Approach**

Firstly, let's address a few questions related to the project.

* How is Our Project Different from Existing Work? Current work focuses on direct predictive methods in which only Ground Station Values i.e. PM values are used to predict future PM values. What we Aim to achieve is Indirect Prediction of PM levels with the help of Satellite and Meteorological Data obtained from the organisation.



* Reason behind following the approach? The Reason behind following this approach is that Particulate Matter is highly dependent on factors such as Aerosol Optical Depth (AOD) and other meteorological parameters such as Relative Humidity , Temperature , Wind Direction , etc. With this , we tend to use these factors to predict the PM values.



* Benefits ? The benefits obtained from following this approach is that for many stations across the country the PM data present in the CPCB website is quite inconsistent and hence it becomes difficult to create a highly accurate time series predictive model . With our approach , we eliminate this problem and instead use reliable and consistent Satellite Data to predict PM levels.
* Why cover the entire country for data collection ? As mentioned in the objectives , we are covering about 45-60 stations spread evenly across the country. The objective behind this is to take in consideration the climatic diversity of the country across different regions. With this , the model would be heavily trained which would lead to highly accurate prediction.

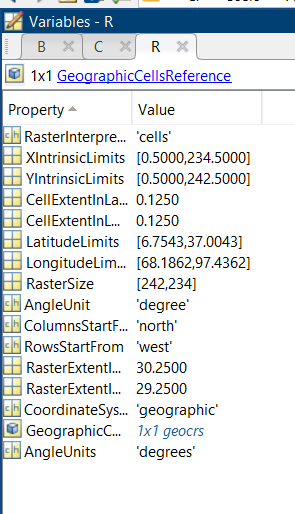
1. **Dataset Creation**

With approach in mind , the main phase of the project was initiated i.e. dataset creation phase. As discussed earlier, we would be using Satellite as well as Meteorological Data from the organisation for prediction purposes. For this we received Geo-TIFF files from IIRS. These TIFF files are tag image image files which contain data in raster format for different parameters.

We received data for two years i.e. 2020 and 2021. Both of them had two sections of Data i.e. AOD data as well as the Meteorological Data which included HPBL. Relative Humidity, Temperature, Wind Direction and Wind Speed. We had to extract data for about 60 stations whose latitudinal and longitudinal values were provided to us along with their PM 2.5 and PM 10 values

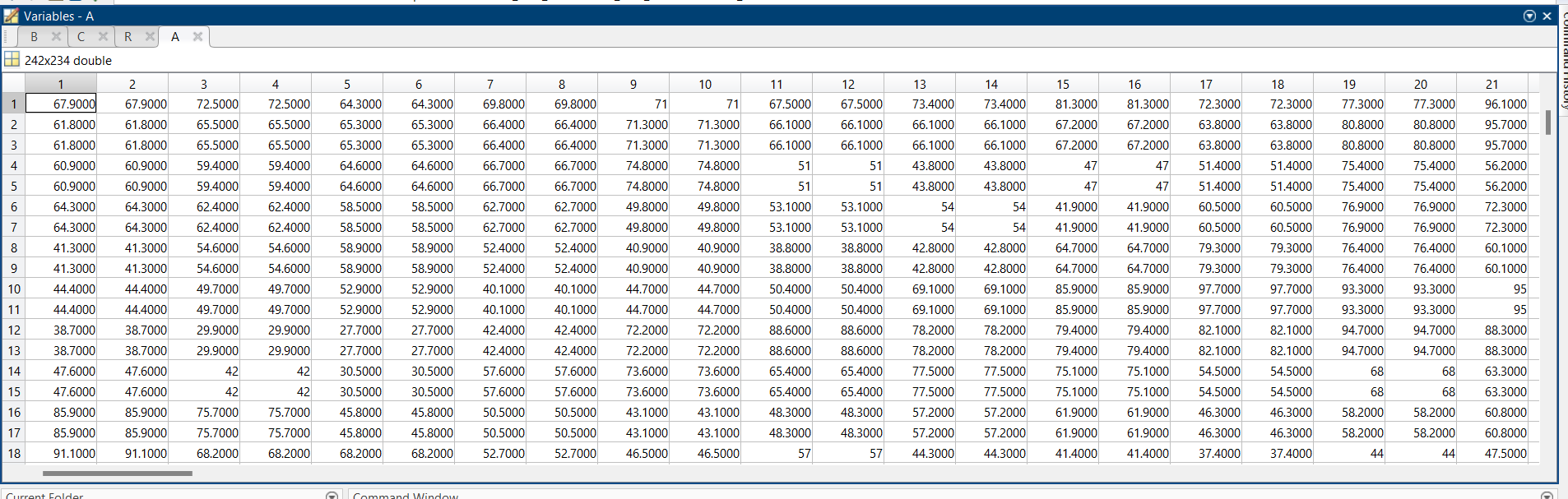
With the help of the Mapping toolbox of MATLAB [9] , we were able to extract as well as view these files. The following code snippets were executed to perform the above mentioned process.



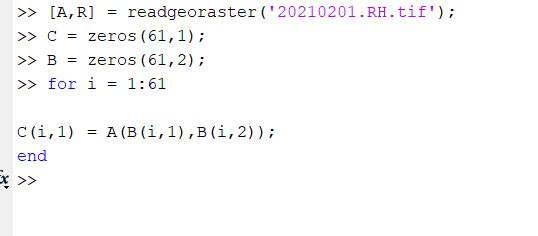
**readgeoreaster** [10] function of the mapping toolbox creates an array by reading geographic or projected raster data from a file. The input argument is the file name which generates two output arguments.The output argument R contains spatial referencing information for the array. The output argument A is a 2-dimensional matrix which contains the raster data (Point Data) . By this , we had successfully converted the image file into point data which was filtered further. The output arguments are displayed below 

As seen , variable R contains different types of information of the 2-D matrix A which contains the point data. A is a 242 X 234 matrix and you can see in the above figure , the columns span from north to south and rows span from west to east, i.e. each cell of the matrix A has a specific latitudinal as well as longitudinal value associated with it . In

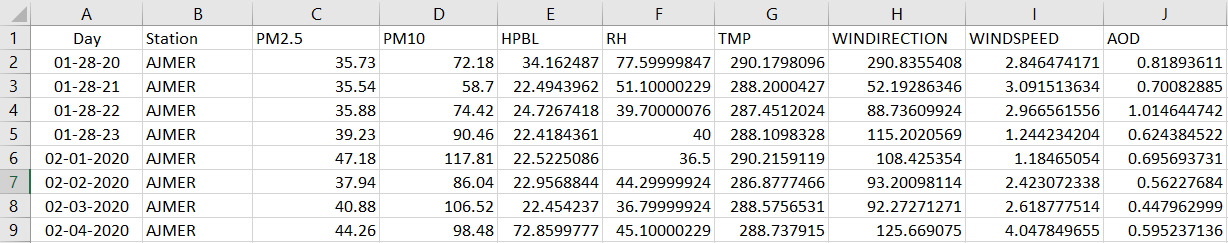
other words , one cell of the matrix describes a point location on the Indian Map. The matrix A is displayed as follows:-



All cells contain different values . These values are basically the parameter measurements at a particular point location on the map. Since we have to take data for 60 stations , a script was written to ease out this complex and tiresome process.

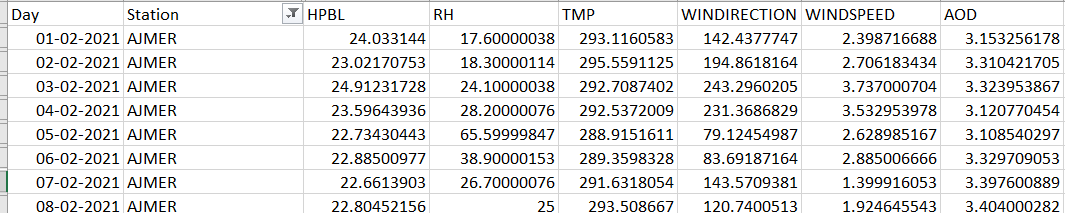


As we know the step size for both columns and rows in matrix A is 0.125 (obtained from output argument R) , latitude and longitude values for the selected stations were converted into corresponding row and column number. This was done as follows Top row has the latitude value of 37 degrees and bottom row has the latitude value of 6.75 degrees whereas leftmost column has longitude value of 68.18 degrees and the rightmost column has the longitude value of 97.4 degree further confirming the fact that each and every location on the map in covered with a step size of 0.125 degree for both latitude and longitude. With the help of a simple mathematical formula, corresponding row and column number for these locations were obtained i.e. the cell corresponding to the selected locations were known now with the help of the Script Written. The Entire Process was repeated for the whole year of 2020 (training data) and the following dataset was created in the excel.



This is the data displayed of the city AJMER for the given days in 2020 . As seen we have successfully extracted the PM levels from ground data and AOD levels along with Meteorological Parameters with the help of Satellite Data. This process is repeated for the different stations and as result a huge Training Dataset is Created which spans the entire year.

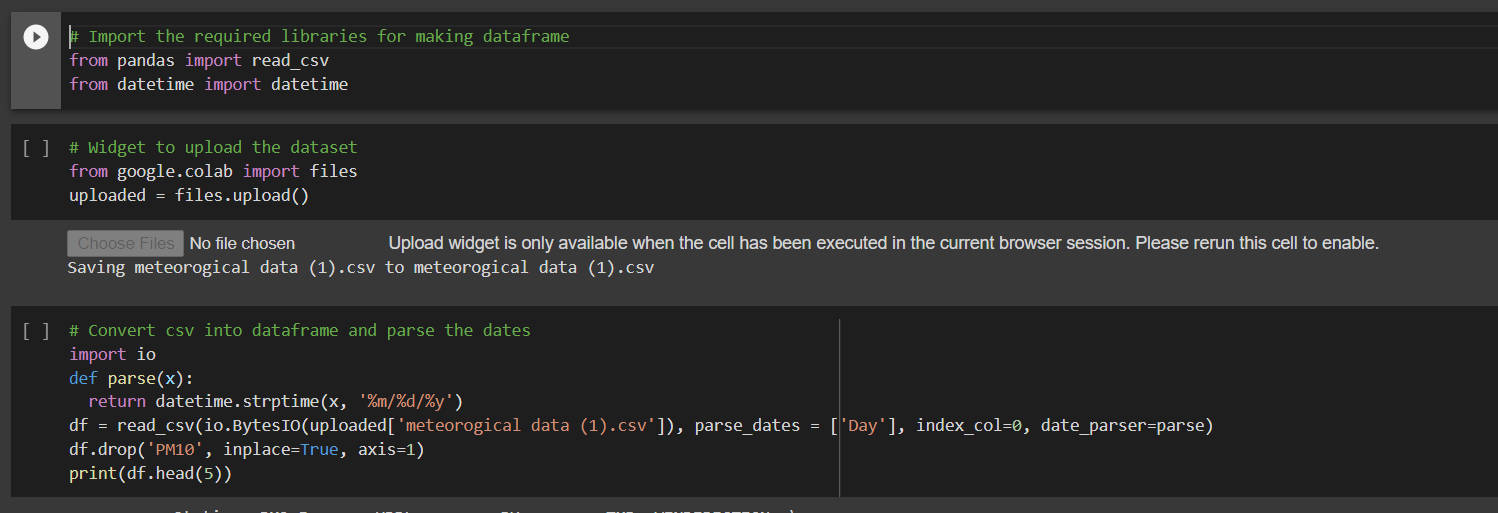
The reason behind training a huge dataset is that India’s Climate across different regions is quite diverse throughout the year. Inorder for developing a highly accurate model , it has to be trained for the entire year to get better predictive results. The same process was then repeated for testing the dataset which included values from Feb 2021. The reason behind choosing Feb as testing data was because it contained the most consistent data as compared to other months which was seen through data visualisation. Feb 2021 data looks like the following:-

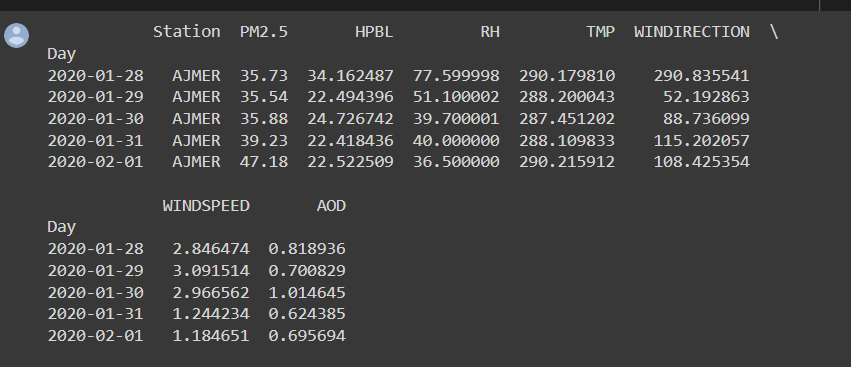


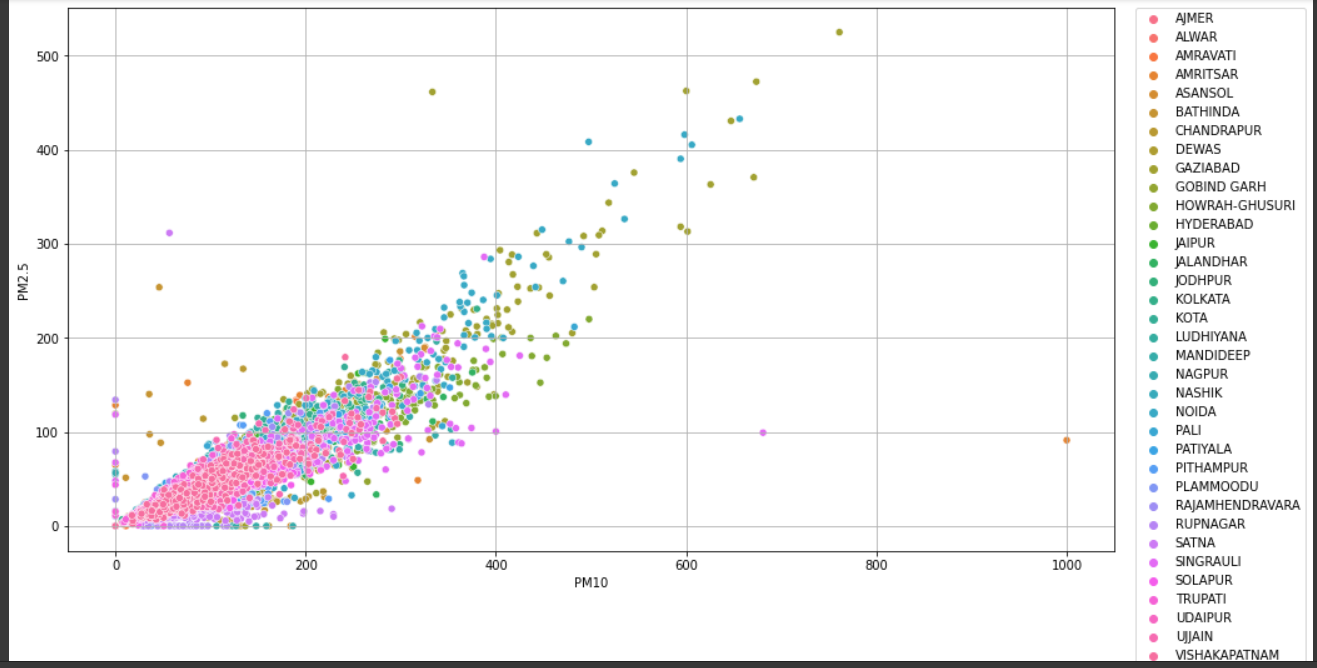
Afterwards, it was observed that there are a lot of garbage values pertaining to AOD which were filtered before running the model on the created dataset.

1. **Data Visualisation**

With Both Datasets Created , Data Visualisation was initiated. Firstly the files were converted into CSV format and were uploaded on the Jupyter Notebook where visualisation was done. Data was filtered according to the station name and PM level (PM2.5 / PM 10). Following code snippet was initiated.



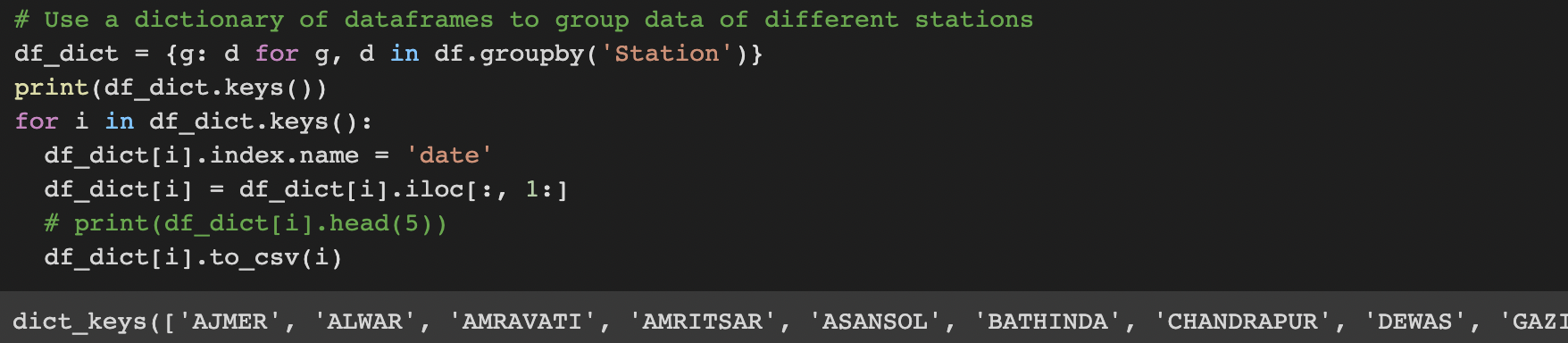
Dates were parsed in order to use them as keys for running the time series model. As seen, a data frame in the notebook was successfully created and further visualisation was done. The Data Frame looked like the following :- 



Above Plot Depicts the variation of PM2.5 and PM10 across different stations accross the whole year (2020).

1. **Model Creation**

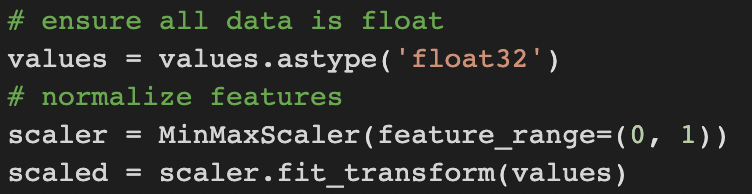
For fitting the data into the neural network, the data was further processed. The dataframe produced above had all stations data bunched up together. As an LSTM model would be run for each of the stations individually, the data had to be grouped according to station name. A dictionary of dataframes was used to accomplish this task.



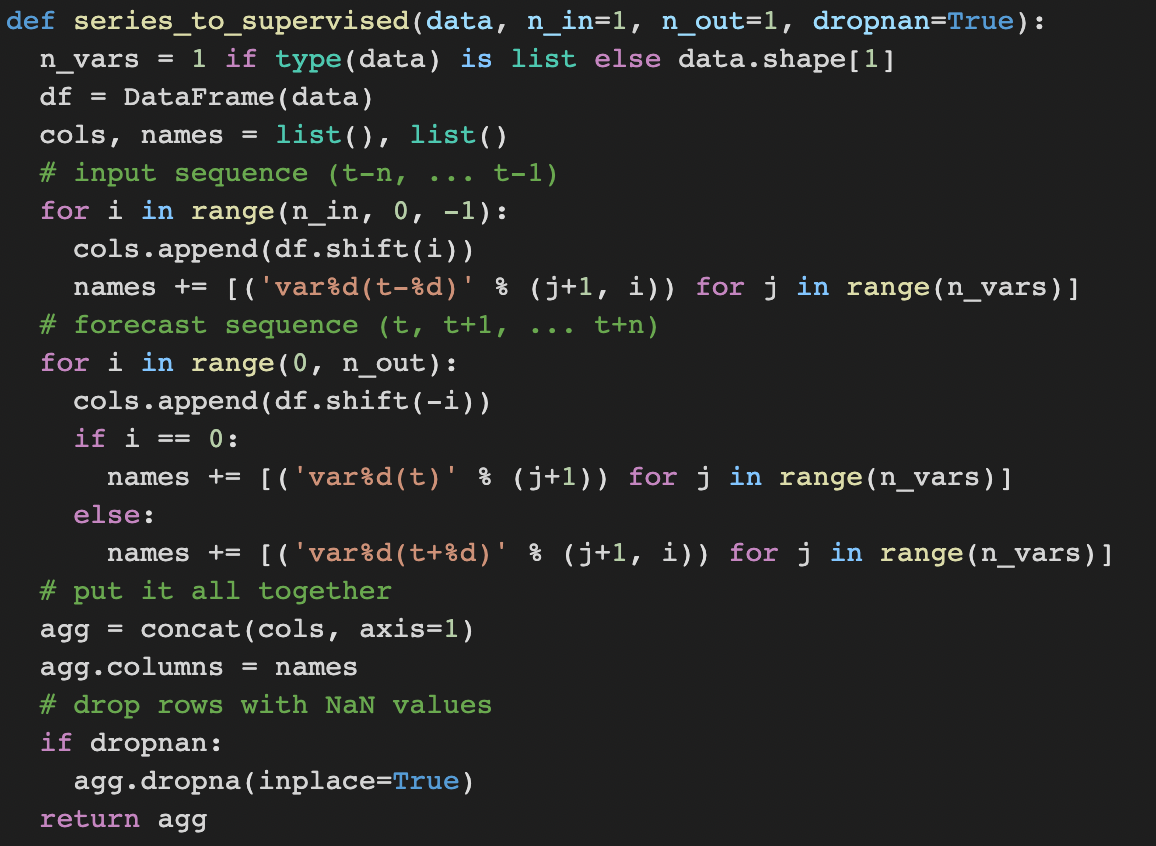
To check if the data had been properly framed until now, a plot of all parameters was prepared for the station of ‘AJMER’.

|  |  |
| --- | --- |

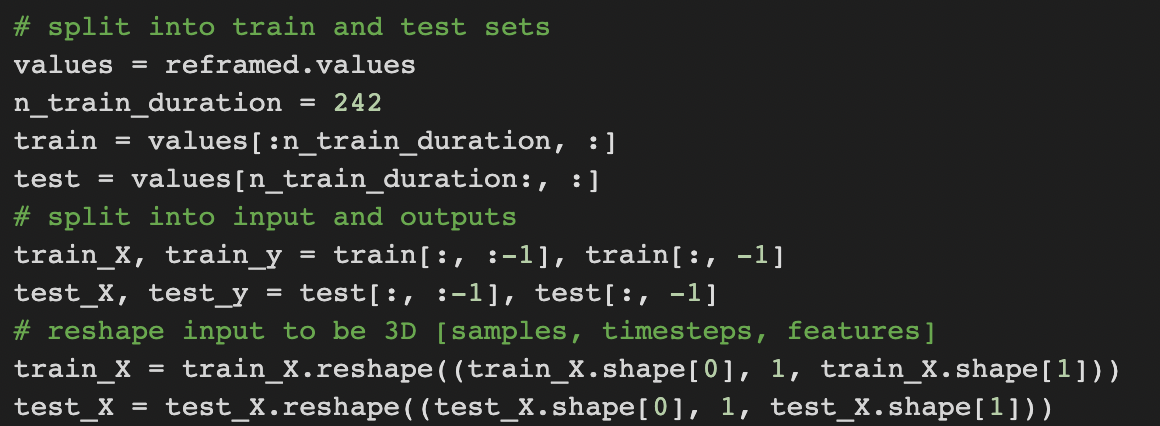
Next, the data was converted to a supervised learning format. The dataset consisted of large and small values, and in order to normalise the data the following code was run.



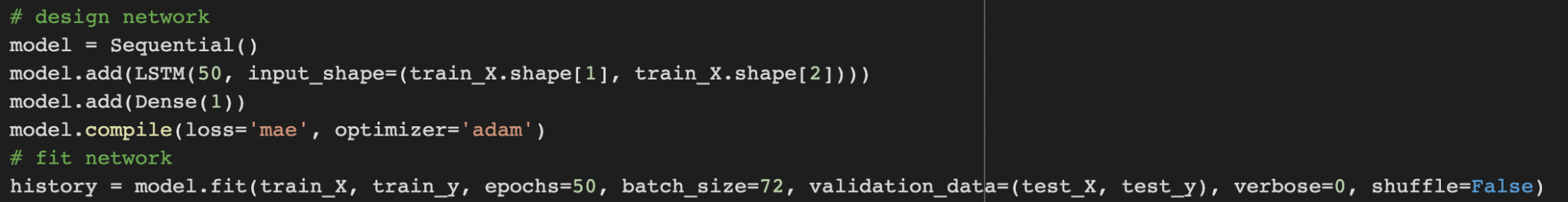
After this, all the values in the dataset lied in between 0 and 1. Finally, the series data was converted into supervised learning data with the help of the following code.



Now, we split the data into training and testing data. For training data we used the data sourced for 2020, and for testing data we used the data sourced for 2021. Each of these had 8 parameters, 7 of which were input parameters and 1 was the output parameter. Finally, the data was reshaped into 3D format so that it could be fed into the neural network.

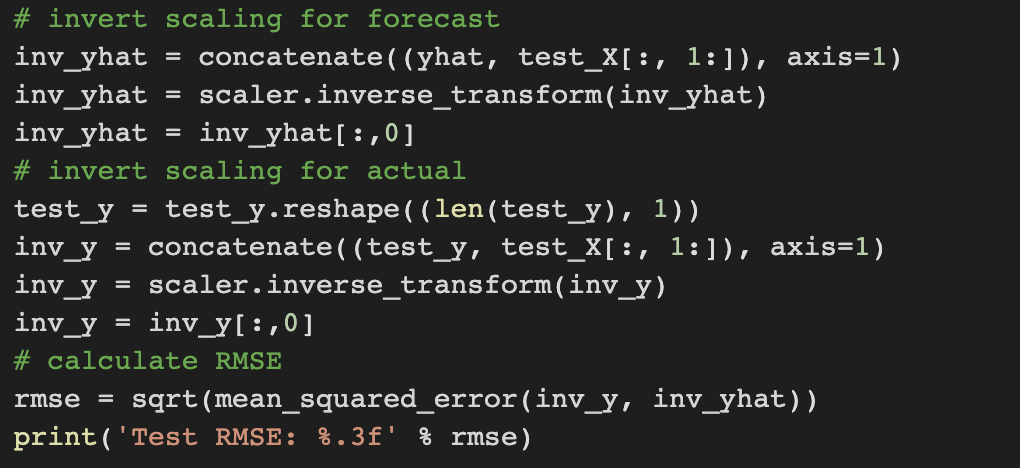


Moving forward to the modelling of the neural network, keras library of Python was used to model the LSTM network. The LSTM model that was used consisted of 50 neurons in the first hidden layer, and 1 neuron in the output layer for predicting PM values. The input shape was 1 time step with 8 features. The loss function used was mean absolute error (MAE), and the efficient Adam version [11] of stochastic gradient descent was also used. The model would be iterated 50 times with a batch size of 72. Lastly, both training and test loss was tracked during training by setting the *validation\_data* argument in the fit() function. The following is the code for the implementation of the model:



**Results and Inferences**

After preparing the model, what was left was to test the data for its accuracy. Root Mean Square Error (or RMSE) was used as a test, as it tends to penalise large errors more. The following code was designed to calculate the RMSE of the predicted values from the actual values.



After calculating the RMSE for PM2.5 and PM10 levels across all the stations, it was found that values ranged between 10 and 20. The predicted value and actual values converged with an increase in window size (as shown in the figure below). The RMSE values calculated were relatively low, indicating that the model was able to accurately predict the PM data.

Some plots for different stations (Ajmer, Amravati, Nagpur, and Nashik) of the predicted values v/s actual values are given below.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

As shown in the plots, the RMSE values mostly ranged between 10 and 20, with a few outliers due to erroneous data. As the window size was increased, the training data series and test data series seemed to converge for most of the stations.

**Conclusion and Future Scope**

After statistical analysis, it is apparent that the LSTM model is able to predict PM2.5 and PM10 levels consistently with quite a high accuracy. The RMSE values were mostly measured between 10 and 20 for different stations across India, and this result is much better than the persistence models used in other studies (having an RMSE of over 30).

Deep learning modelling is a very fast-expanding field, with new and efficient algorithms being developed every other day. One recent model, GRU (Gated Recurrent Units) could

help speed up the processing with a very low loss in accuracy. Such neural networks could help to process large amounts of data, such as the data for the entire terrain of India.

The data collected from the model is in the form of a CSV file of point data. In the future, one could predict the data for every single coordinate point in the GeoTIFF input file and create a spatial heat map to depict the data in a more convenient manner. This data could also be used for further research purposes that involve PM values as input parameters.

**References**

[1] <https://www.iirs.gov.in/institute-profile>

[2] Harishkumar, K. S., K. M. Yogesh, and Ibrahim Gad. "Forecasting air pollution particulate matter (PM2. 5) using machine learning regression models." Procedia Computer Science 171 (2020): 2057-2066.

[3] Tiwari, Animesh, Rishabh Gupta, and Rohitash Chandra. "Delhi air quality prediction using LSTM deep learning models with a focus on COVID-19 lockdown." arXiv preprint arXiv:2102.10551 (2021).

[4] Seng, Dewen, Qiyan Zhang, Xuefeng Zhang, Guangsen Chen, and Xiyuan Chen. "Spatiotemporal prediction of air quality based on LSTM neural network." Alexandria Engineering Journal 60, no. 2 (2021).

[5] Athira, V., P. Geetha, Rab Vinayakumar, and K. P. Soman. "Deepairnet: Applying recurrent networks for air quality prediction." *Procedia computer science* 132 (2018): 1394-1403.

[6] Hossain, Emam, Mohd Arafath Uddin Shariff, Mohammad Shahadat Hossain, and Karl Andersson. "A novel deep learning approach to predict air quality index." In *Proceedings of International Conference on Trends in Computational and Cognitive Engineering*, pp. 367-381. Springer, Singapore, 2021.

[7] Yang, Guang, HwaMin Lee, and Giyeol Lee. "A hybrid deep learning model to forecast particulate matter concentration levels in Seoul, South Korea." *Atmosphere* 11, no. 4 (2020): 348.

[8] Qadeer, Khaula, Wajih Ur Rehman, Ahmad Muqeem Sheri, Inyoung Park, Hong Kook Kim, and Moongu Jeon. "A long short-term memory (LSTM) network for hourly estimation of PM2. 5 concentration in two cities of South Korea." *Applied Sciences* 10, no. 11 (2020): 3984.

[9] <https://docs.google.com/document/d/1UGRi-5v0ghay7PtMFctc7Y0WJtk8Okn61xmmqqsELEM/edit#>

[10] <https://in.mathworks.com/help/map/ref/readgeoraster.html>

[11] <https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>