Integrated Dual LSTM Model-Based Air Quality Prediction

Rajesh Reddy Muley, Vadlamudi Teja Sai Sri, Kuntamukkala Kiran Kumar, and Kakumanu Manoj Kumar

Abstract Although air quality prediction is a crucial tool for weather forecasting and air quality management, algorithms for making predictions that are based on a single model are prone to overfitting. In order to address the complexity of air quality prediction, a prediction approach based on integrated dual long short-term memory (LSTM) models was developed in this study. The model takes into account the variables that affect air quality such as nearby station data and weather information. Finally, two models are integrated using the eXtreme Gradient Boosting (XGBoosting) tree. The ultimate results of the prediction may be obtained by summing the predicted values of the ideal subtree nodes. The proposed method was tested and examined using five evaluation techniques. The accuracy of the prediction data in our model has significantly increased when compared with other models.

Keywords XGBoosting · LSTM · Accuracy · Prediction

1 Introduction

The amount of exhaust gas produced by several factories and automobiles continues to climb as industrialisation levels rise, substantially increasing air pollution. People's daily lives are significantly impacted by air quality. Accurate air quality forecasting has emerged as a key strategy for reducing pollution and raising air quality. Data on air quality has caused great worry throughout the world. For predicting air quality, time series data prediction techniques are frequently employed, along with time series prediction models and conventional machine learning techniques. Some

R. R. Muley (✉)
Department of Information Technology, Lakireddy Bali Reddy College of Engineering, Mylavaram, Andhra Pradesh, India
e-mail: rajeshreddi.m@gmail.com

V. T. S. Sri · K. K. Kumar · K. M. Kumar
Lakireddy Bali Reddy College of Engineering, Mylavaram, Andhra Pradesh, India

A. E. Hassanien et al. (eds.), *International Conference on Innovative Computing and Communications*, Lecture Notes in Networks and Systems 731,
https://doi.org/10.1007/978-981-99-4071-4_55

methods imitate the temporal and geographical dependency of air quality data concurrently. However, commonly used machine learning techniques frequently exhibit considerable performance variability under various conditions. Numerous variables, including temperature, wind speed, and geographical arrangement have an impact on air quality. As a result, it is challenging to produce certain and precise prediction results using the popular single model prediction method. Our thoughts in this paper are based on a strategy that has recently been discussed in the literature: integrating various models to predict air quality. When compared with current models, the integrated model can greatly increase the ability to forecast. But there is still much to learn about how to combine the benefits of several models depending on the features of the data collection.

2 Literature Survey

Petr Hájek et al. in [1] genetic algorithms optimise the input variable sets for each forecast of an air pollutant. Based on information gathered by the Pardubice city monitoring station in the Czech Republic, models are developed to predict the specific air quality indices for each air pollutant. The results show that when the root mean squared error is taken into consideration, individual prediction model compositions outperform single forecasts of the common air quality index. As a result, these models can be used to produce air quality index predictions that are more accurate one day in advance.

In order to avoid air pollution in urban areas and improve the quality of life for city dwellers, Kang et al. [2] highlighted the importance of conducting work on city air quality forecasting. Following that, AQI prediction models based on back propagation (BP) neural networks, genetic algorithm optimisation, and genetic simulated annealing algorithm optimisation are established. Comparing and evaluating the prediction outcomes reveal that the BP neural network based on genetic simulated annealing method has a higher accuracy rate, excellent generalisation capacity, and global search ability.

According to Wang et al. [3], who found that air pollution was becoming more severe, the most significant air pollutant, PM2.5 in aerosols, had a negative impact on people's regular output, way of life, and employment, as well as their health. As a result, the forecasting of PM2.5 concentration has taken on significant practical importance. The study selects real-time air quality data that is released, collects historical monitoring data of air environmental contaminants, normalises the data, and then splits the sample data into the two sets in a suitable ratio to form the training dataset and test dataset.

A key component of a smart city is a system for measuring and forecasting air quality, Mahajan et al. [5]. Making a forecast system with great accuracy and a reasonable calculation time is one of the biggest challenges. In this study, we demonstrate that a variety of clustering algorithms may be used to forecast fine particulate matter (PM2.5) concentrations reliably and quickly. We cluster the monitoring stations depending on their geographic proximity using a grid-based methodology. Data from 557 stations that have been distributed throughout Taiwan's Airbox device network is used in the tests and evaluation. The accuracy and processing time of the various clustering algorithms are compared in a final study.

3 Existing System

Commonly used machine learning techniques frequently exhibit considerable performance variability under various conditions. Numerous variables, including temperature, wind speed, and geographical arrangement have an impact on air quality. As a result, it is challenging to produce certain and precise prediction results using the popular single model prediction method.

4 Proposed System

In this work, a prediction approach based on integrated dual long short-term memory (LSTM) models was created to handle the complexity of air quality prediction. First, a single-factor prediction model that can independently forecast the value of each component in air quality data is created using sequence to sequence (Seq2Seq) technology. The multi-factor prediction model is then the LSTM model plus the attention mechanism. The model takes into account the air quality parameters such as the data from nearby stations and the weather. The two models are then combined using the eXtreme Gradient Boosting (XGBoosting) tree.

5 System Architecture

See Fig. 1.

6 Flow Chart

See Fig. 2.

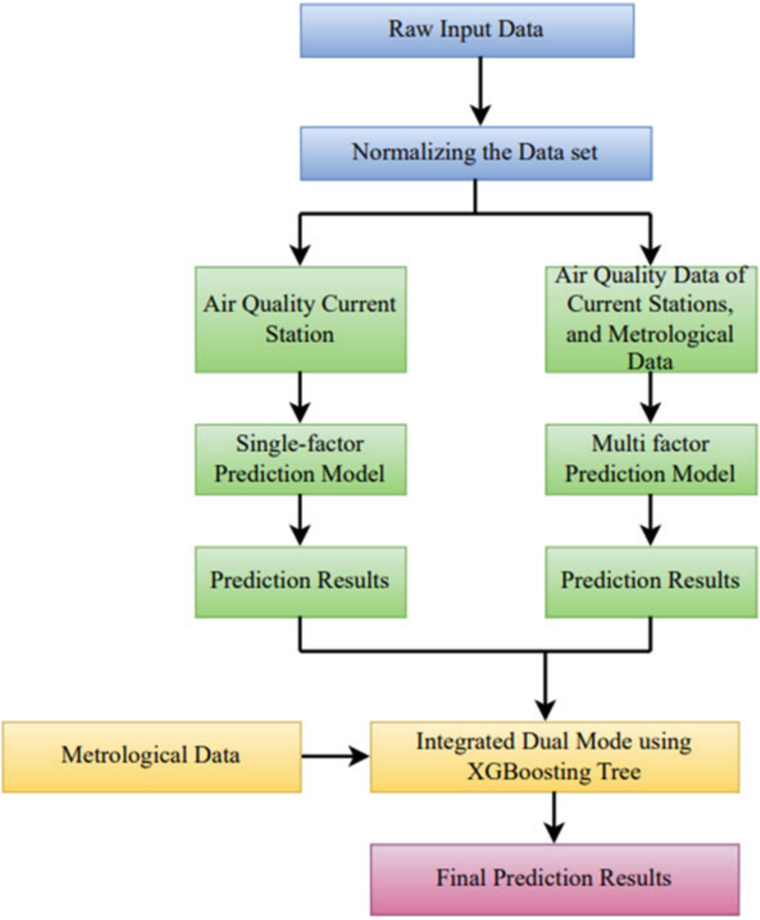

Raw Input Data
Normalizing the Data set
Air Quality Current Station
Air Quality Data of Current Stations, and Metrological Data
Single-factor Prediction Model
Multi factor Prediction Model
Prediction Results
Prediction Results
Metrological Data
Integrated Dual Mode using XGBoosting Tree
Final Prediction Results

Fig. 1 System architecture

7 Results

Single-factor model is subsequently improved using the ATTENTION layer to create a multi-factor (combination of LSTM, sequence 2 sequence, and attention). In order to combine both models and improve prediction accuracy, features from the multi-model are extracted and retrained using XGBOOST.

The screen below displays information from the air quality dataset, which was used to construct this project.

The first row of the dataset's screen in Fig. 3 shows its column names, while the following rows show its values. As the training features, we used PM values, the target variable, and others.

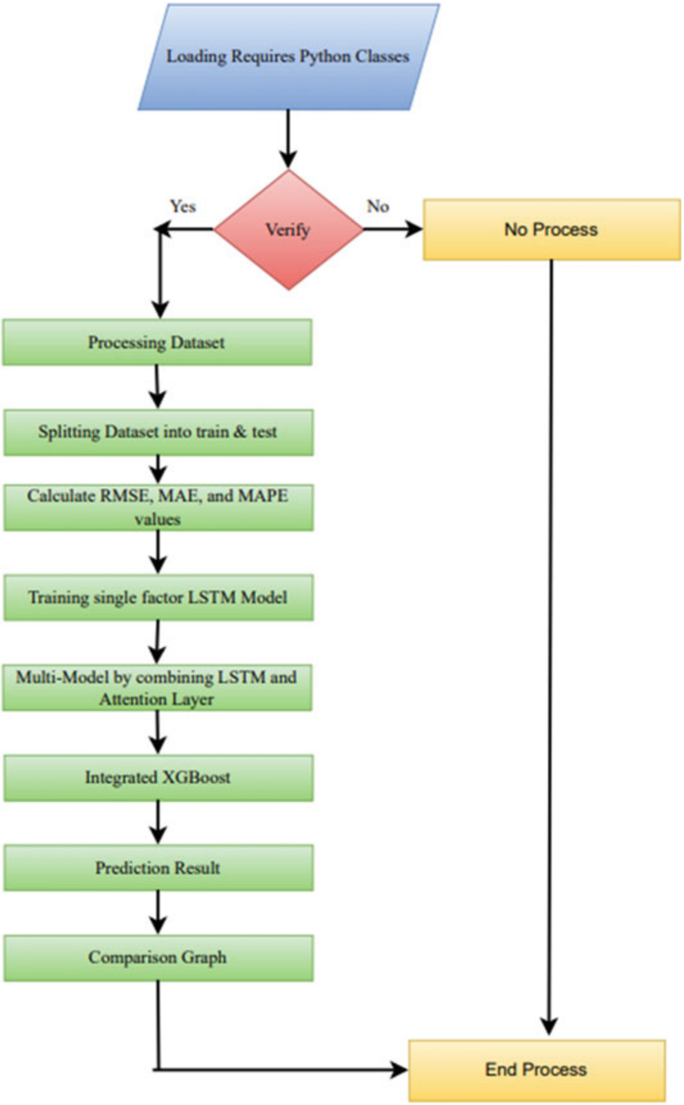
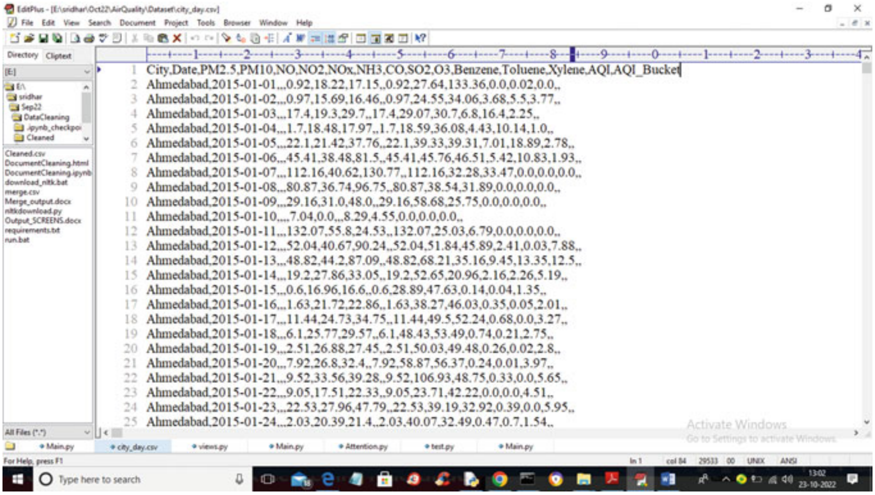

Loading Requires Python Classes
Yes
Verify
No
No Process
Processing Dataset
Splitting Dataset into train & test
Calculate RMSE, MAE, and MAPE values
Training single factor LSTM Model
Multi-Model by combining LSTM and Attention Layer
Integrated XGBoost
Prediction Result
Comparison Graph
End Process

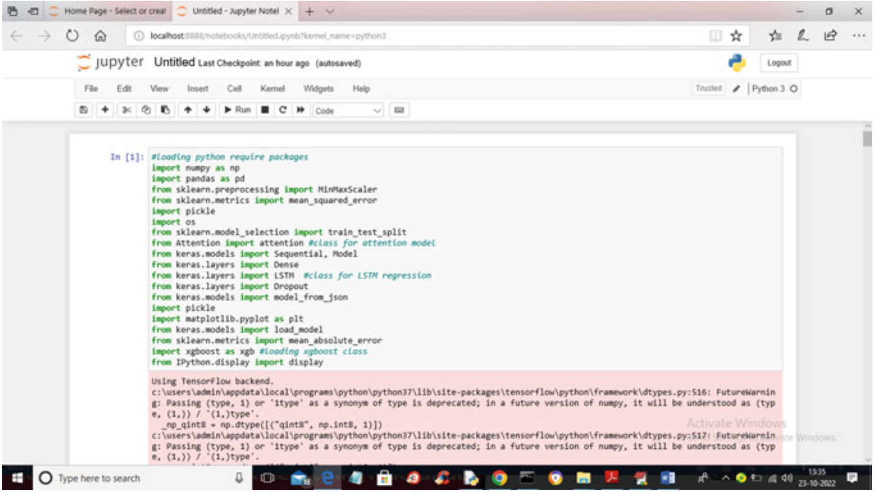
Fig. 2 Flow chart

All three models—single-factor LSTM, multi-factor LSTM with attention, and multi-factor integrated with XGBOOST—have been coded by our team. Below are the code and output screens for all the models we coded in the Jupyter notebook. You can see BLUE colour comments in each screen to learn about code (Figs. 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18 and 19).



```
City,Date,PM2.5,PM10,NO,NO2,NOx,NH3,CO,SO2,O3,Benzene,Toluene,Xylene,AQI,AQI_Bucket
Ahmedabad,2015-01-01,,,0.92,18.22,17.15,,0.92,27.64,133.36,0.0,0.02,0.0,,
Ahmedabad,2015-01-02,,,0.97,15.69,16.46,,0.97,24.55,34.06,3.68,5.5,3.77,,
Ahmedabad,2015-01-03,,,17.4,19.3,29.7,,17.4,29.07,30.7,6.8,16.4,2.25,,
Ahmedabad,2015-01-04,,,1.7,18.48,17.97,,1.7,18.59,36.08,4.43,10.14,1.0,,
Ahmedabad,2015-01-05,,,22.1,21.42,37.76,,22.1,39.33,39.31,7.01,18.89,2.78,,
Ahmedabad,2015-01-06,,,45.41,38.48,81.5,,45.41,45.76,46.51,5.42,10.83,1.93,,
Ahmedabad,2015-01-07,,,112.16,40.62,130.77,,112.16,32.28,33.47,0.0,0.0,0.0,,
Ahmedabad,2015-01-08,,,80.87,36.74,96.75,,80.87,38.54,31.89,0.0,0.0,0.0,,
Ahmedabad,2015-01-09,,,29.16,31.0,48.0,,29.16,58.68,25.75,0.0,0.0,0.0,,
Ahmedabad,2015-01-10,,,,7.04,0.0,,,8.29,4.55,0.0,0.0,0.0,,
Ahmedabad,2015-01-11,,,132.07,55.8,24.53,,132.07,25.03,6.79,0.0,0.0,0.0,,
Ahmedabad,2015-01-12,,,52.04,40.67,90.24,,52.04,51.84,45.89,2.41,0.03,7.88,,
Ahmedabad,2015-01-13,,,48.82,44.2,87.09,,48.82,68.21,35.16,9.45,13.35,12.5,,
Ahmedabad,2015-01-14,,,19.2,27.86,33.05,,19.2,52.65,20.96,2.16,2.26,5.19,,
Ahmedabad,2015-01-15,,,0.6,16.96,16.6,,0.6,28.89,47.63,0.14,0.04,1.35,,
Ahmedabad,2015-01-16,,,1.63,21.72,22.86,,1.63,38.27,46.03,0.35,0.05,2.01,,
Ahmedabad,2015-01-17,,,11.44,24.73,34.75,,11.44,49.5,52.24,0.68,0.0,3.27,,
Ahmedabad,2015-01-18,,,6.1,25.77,29.57,,6.1,48.43,53.49,0.74,0.21,2.75,,
Ahmedabad,2015-01-19,,,2.51,26.88,27.45,,2.51,50.03,49.48,0.26,0.02,2.8,,
Ahmedabad,2015-01-20,,,7.92,26.8,32.4,,7.92,58.87,56.37,0.24,0.01,3.97,,
Ahmedabad,2015-01-21,,,9.52,33.56,39.28,,9.52,106.93,48.75,0.33,0.0,5.65,,
Ahmedabad,2015-01-22,,,9.05,17.51,22.33,,9.05,23.71,42.22,0.0,0.0,4.51,,
Ahmedabad,2015-01-23,,,22.53,27.96,47.79,,22.53,39.19,32.92,0.39,0.0,5.95,,
Ahmedabad,2015-01-24,,,2.03,20.39,21.4,,2.03,40.07,32.49,0.47,0.7,1.54,,
```

Fig. 3 Air quality dataset

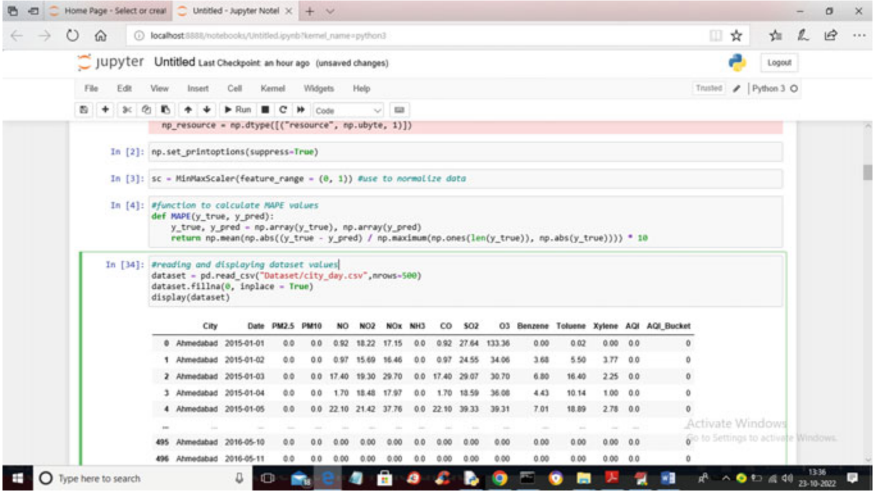


```
In [1]: #Loading python require packages
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
import pickle
import os
from sklearn.model_selection import train_test_split
from Attention import attention #class for attention model
from keras.models import Sequential, Model
from keras.layers import Dense
from keras.layers import LSTM  #class for LSTM regression
from keras.layers import Dropout
from keras.models import model_from_json
import pickle
import matplotlib.pyplot as plt
from keras.models import load_model
from sklearn.metrics import mean_absolute_error
import xgboost as xgb #loading xgboost class
from IPython.display import display
```

```
Using TensorFlow backend.
c:\users\admin\appdata\local\programs\python\python37\lib\site-packages\tensorflow\python\framework\dtypes.py:516: FutureWarnin
g: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (typ
e, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
c:\users\admin\appdata\local\programs\python\python37\lib\site-packages\tensorflow\python\framework\dtypes.py:517: FutureWarnin
g: Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (typ
e, (1,)) / '(1,)type'.
```

Home Page - Select or creat
Untitled - Jupyter Notel
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
13:35
23-10-2022

Fig. 4 Above screen we are loading required Python classes, LSTM, attention and XGBOOST class


Home Page - Select or creat
Untitled - Jupyter Notel
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:36
23-10-2022

```
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

```
In [2]: np.set_printoptions(suppress=True)
```

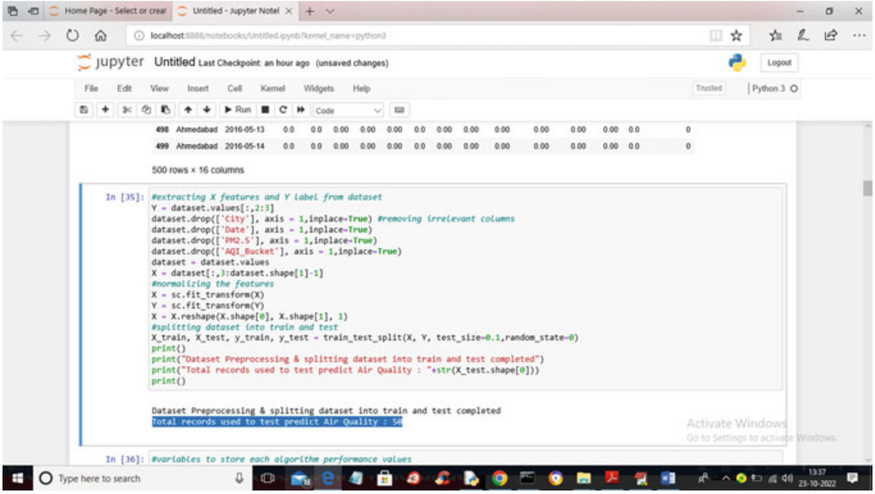
```
In [3]: sc = MinMaxScaler(feature_range = (0, 1)) #use to normalize data
```

```
In [4]: #function to calculate MAPE values
        def MAPE(y_true, y_pred):
            y_true, y_pred = np.array(y_true), np.array(y_pred)
            return np.mean(np.abs((y_true - y_pred) / np.maximum(np.ones(len(y_true)), np.abs(y_true)))) * 10
```

```
In [34]: #reading and displaying dataset values
         dataset = pd.read_csv("Dataset/city_day.csv",nrows=500)
         dataset.fillna(0, inplace = True)
         display(dataset)
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
0	Ahmedabad	2015-01-01	0.0	0.0	0.92	18.22	17.15	0.0	0.92	27.64	133.36	0.00	0.02	0.00	0.0	0
1	Ahmedabad	2015-01-02	0.0	0.0	0.97	15.69	16.46	0.0	0.97	24.55	34.06	3.68	5.50	3.77	0.0	0
2	Ahmedabad	2015-01-03	0.0	0.0	17.40	19.30	29.70	0.0	17.40	29.07	30.70	6.80	16.40	2.25	0.0	0
3	Ahmedabad	2015-01-04	0.0	0.0	1.70	18.48	17.97	0.0	1.70	18.59	36.08	4.43	10.14	1.00	0.0	0
4	Ahmedabad	2015-01-05	0.0	0.0	22.10	21.42	37.76	0.0	22.10	39.33	39.31	7.01	18.89	2.78	0.0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
495	Ahmedabad	2016-05-10	0.0	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.0	0
496	Ahmedabad	2016-05-11	0.0	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.0	0

Fig. 5 Above screen we define function to calculate MAPE and to normalise values and then reading and displaying dataset values


Home Page - Select or creat
Untitled - Jupyter Notel
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:37
23-10-2022

498	Ahmedabad	2016-05-13	0.0	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.0	0
499	Ahmedabad	2016-05-14	0.0	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.0	0

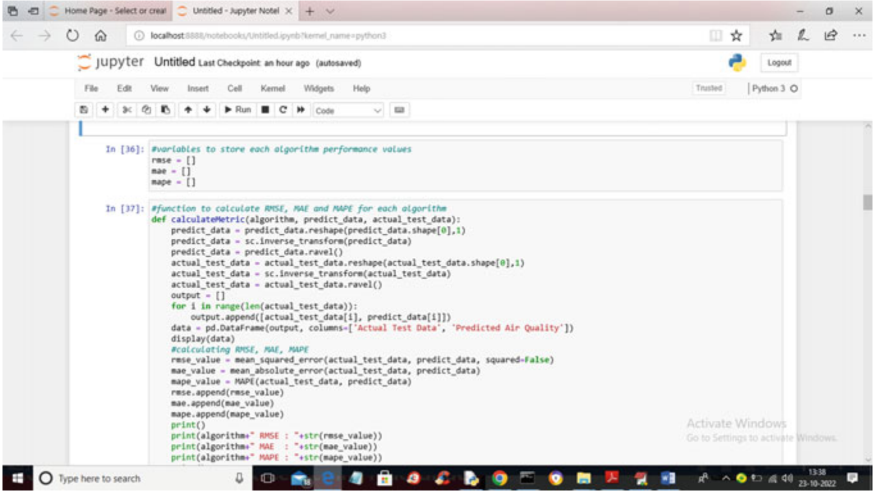
500 rows × 16 columns

```
In [35]: #extracting X features and Y label from dataset
         Y = dataset.values[:,2:3]
         dataset.drop(['City'], axis = 1,inplace=True) #removing irrelevant columns
         dataset.drop(['Date'], axis = 1,inplace=True)
         dataset.drop(['PM2.5'], axis = 1,inplace=True)
         dataset.drop(['AQI_Bucket'], axis = 1,inplace=True)
         dataset = dataset.values
         X = dataset[:,3:dataset.shape[1]-1]
         #normalizing the features
         X = sc.fit_transform(X)
         Y = sc.fit_transform(Y)
         X = X.reshape(X.shape[0], X.shape[1], 1)
         #splitting dataset into train and test
         X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.1,random_state=0)
         print()
         print("Dataset Preprocessing & splitting dataset into train and test completed")
         print("Total records used to test predict Air Quality : "+str(X_test.shape[0]))
         print()
```

```
Dataset Preprocessing & splitting dataset into train and test completed
Total records used to test predict Air Quality : 50
```

```
In [36]: #variables to store each algorithm performance values
```

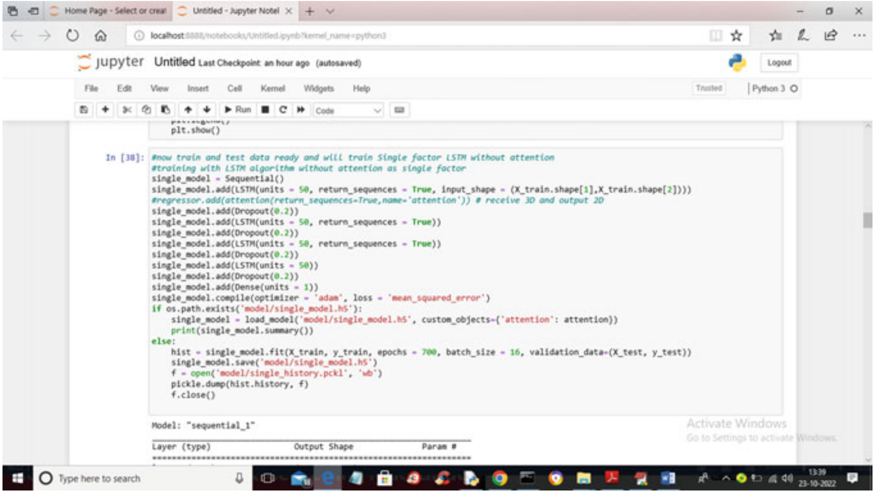
Fig. 6 Above screen we are processing dataset and then removing irrelevant columns and then splitting dataset into train and test and for testing we used 50 values which considers 1 test data per second for next one hour. These test values you can see in blue colour text


Home Page - Select or creat
Untitled - Jupyter Notel
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:38
23-10-2022

```
In [36]: #variables to store each algorithm performance values
         rmse = []
         mae = []
         mape = []

In [37]: #function to calculate RMSE, MAE and MAPE for each algorithm
         def calculateMetric(algorithm, predict_data, actual_test_data):
             predict_data = predict_data.reshape(predict_data.shape[0],1)
             predict_data = sc.inverse_transform(predict_data)
             predict_data = predict_data.ravel()
             actual_test_data = actual_test_data.reshape(actual_test_data.shape[0],1)
             actual_test_data = sc.inverse_transform(actual_test_data)
             actual_test_data = actual_test_data.ravel()
             output = []
             for i in range(len(actual_test_data)):
                 output.append([actual_test_data[i], predict_data[i]])
             data = pd.DataFrame(output, columns=['Actual Test Data', 'Predicted Air Quality'])
             display(data)
             #calculating RMSE, MAE, MAPE
             rmse_value = mean_squared_error(actual_test_data, predict_data, squared=False)
             mae_value = mean_absolute_error(actual_test_data, predict_data)
             mape_value = MAPE(actual_test_data, predict_data)
             rmse.append(rmse_value)
             mae.append(mae_value)
             mape.append(mape_value)
             print()
             print(algorithm+" RMSE : "+str(rmse_value))
             print(algorithm+" MAE  : "+str(mae_value))
             print(algorithm+" MAPE : "+str(mape_value))
```

Fig. 7 Above screen we are defining function to calculate RMSE, MAE, and MAPE values and then plot TEST data air quality and predicted air quality graph

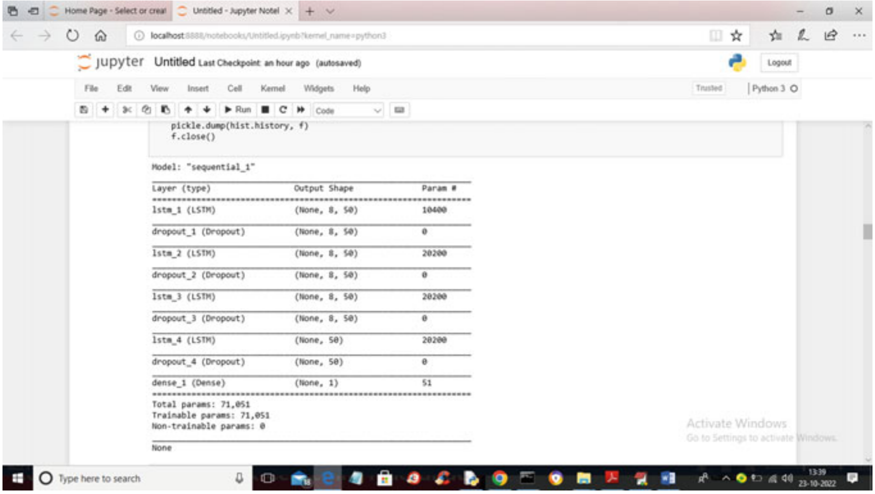

Home Page - Select or creat
Untitled - Jupyter Notel
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:39
23-10-2022

```
    plt.show()

In [38]: #now train and test data ready and will train Single factor LSTM without attention
         #training with LSTM algorithm without attention as single factor
         single_model = Sequential()
         single_model.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1],X_train.shape[2])))
         #regressor.add(attention(return_sequences=True,name='attention')) # receive 3D and output 2D
         single_model.add(Dropout(0.2))
         single_model.add(LSTM(units = 50, return_sequences = True))
         single_model.add(Dropout(0.2))
         single_model.add(LSTM(units = 50, return_sequences = True))
         single_model.add(Dropout(0.2))
         single_model.add(LSTM(units = 50))
         single_model.add(Dropout(0.2))
         single_model.add(Dense(units = 1))
         single_model.compile(optimizer = 'adam', loss = 'mean_squared_error')
         if os.path.exists('model/single_model.h5'):
             single_model = load_model('model/single_model.h5', custom_objects={'attention': attention})
             print(single_model.summary())
         else:
             hist = single_model.fit(X_train, y_train, epochs = 700, batch_size = 16, validation_data=(X_test, y_test))
             single_model.save('model/single_model.h5')
             f = open('model/single_history.pckl', 'wb')
             pickle.dump(hist.history, f)
             f.close()

         Model: "sequential_1"
         _________________________________________________________________
         Layer (type)                 Output Shape              Param #
         =================================================================
```

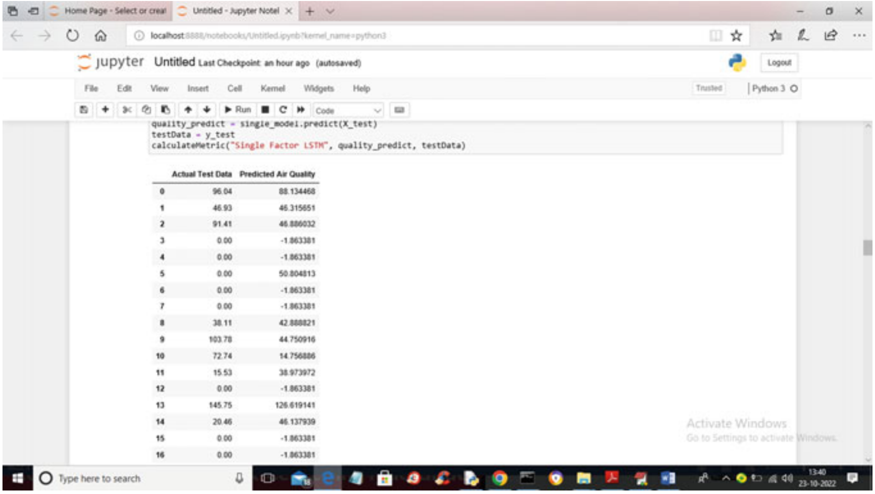
Fig. 8 Above screen we are training single-factor LSTM model and below is the LSTM model summary


Home Page - Select or creat
Untitled - Jupyter Notel
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:39
23-10-2022

```
    pickle.dump(hist.history, f)
    f.close()

Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 8, 50)             10400
_________________________________________________________________
dropout_1 (Dropout)          (None, 8, 50)             0
_________________________________________________________________
lstm_2 (LSTM)                (None, 8, 50)             20200
_________________________________________________________________
dropout_2 (Dropout)          (None, 8, 50)             0
_________________________________________________________________
lstm_3 (LSTM)                (None, 8, 50)             20200
_________________________________________________________________
dropout_3 (Dropout)          (None, 8, 50)             0
_________________________________________________________________
lstm_4 (LSTM)                (None, 50)                20200
_________________________________________________________________
dropout_4 (Dropout)          (None, 50)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 51
=================================================================
Total params: 71,051
Trainable params: 71,051
Non-trainable params: 0
_________________________________________________________________
None
```

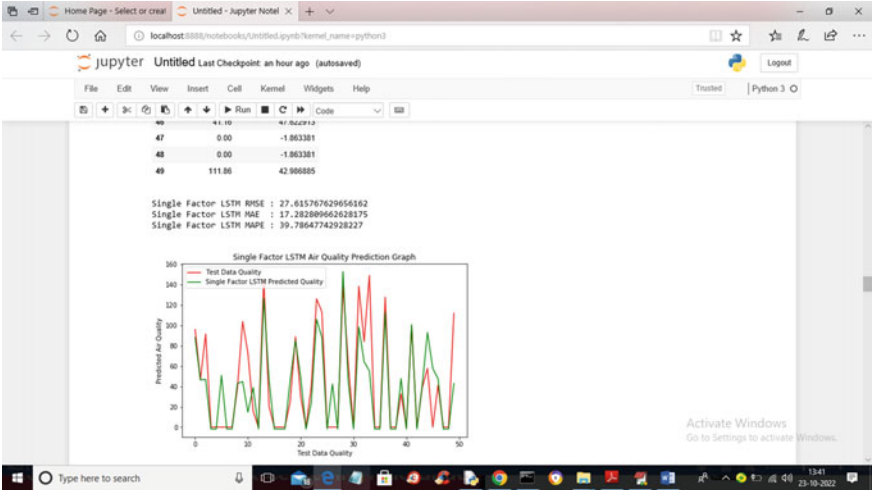
Fig. 9 Above summary you can see that in single-factor LSTM there is no attention layer and in below screen we can see single-factor model prediction output


Home Page - Select or creat
Untitled - Jupyter Notel
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:40
23-10-2022

```
quality_predict = single_model.predict(X_test)
testData = y_test
calculateMetric("Single Factor LSTM", quality_predict, testData)
```

	Actual Test Data	Predicted Air Quality
0	96.04	88.134468
1	46.93	46.315651
2	91.41	46.886032
3	0.00	-1.863381
4	0.00	-1.863381
5	0.00	50.804813
6	0.00	-1.863381
7	0.00	-1.863381
8	38.11	42.888821
9	103.78	44.750916
10	72.74	14.756886
11	15.53	38.973972
12	0.00	-1.863381
13	145.75	126.619141
14	20.46	46.137939
15	0.00	-1.863381
16	0.00	-1.863381

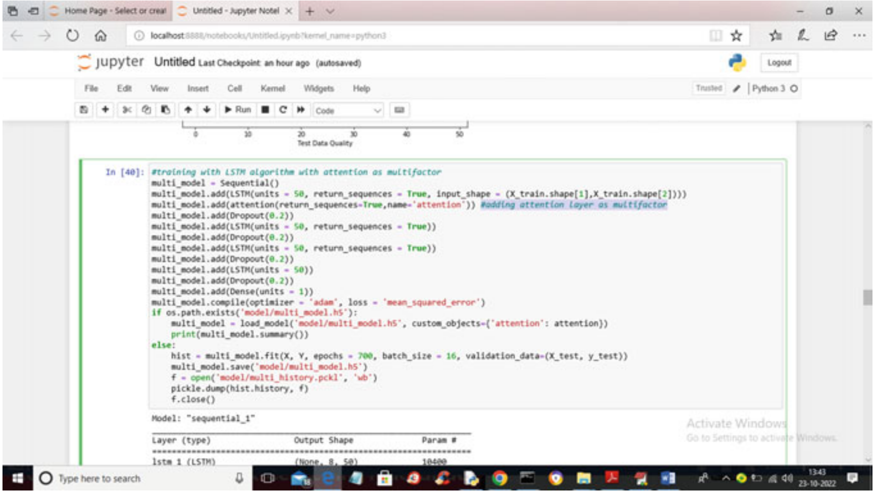
Fig. 10 Above screen we can see single-factor test data quality and predicted air quality for next 50 records


Home Page - Select or creat
Untitled - Jupyter Notel
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Single Factor LSTM Air Quality Prediction Graph
Test Data Quality
Single Factor LSTM Predicted Quality
Predicted Air Quality
160
140
120
100
80
60
40
20
0
0
10
20
30
40
50
Test Data Quality
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:41
23-10-2022

```
46      41.10     47.622913
47       0.00     -1.863381
48       0.00     -1.863381
49     111.86     42.986885

Single Factor LSTM RMSE : 27.615767629656162
Single Factor LSTM MAE  : 17.282809662628175
Single Factor LSTM MAPE : 39.78647742928227
```

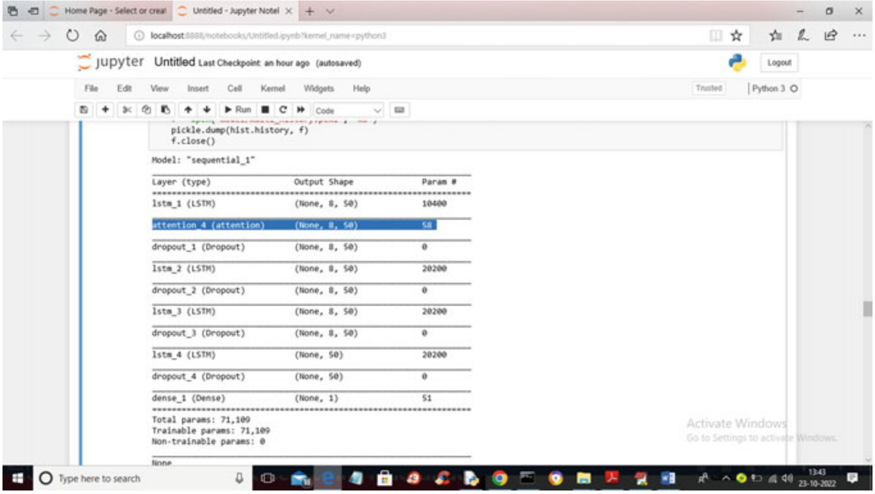
Fig. 11 Above screen with single factor we got RMSE as 27 and MAE as 17, and MAPE as 39 and then in graph x-axis represents 50 min and y-axis represents air quality values, red line represents TEST data actual values, and green line represents predicted air quality and we can see both lines are overlapping with little difference. In below screen we can see LSTM with attention


Home Page - Select or creat
Untitled - Jupyter Notel
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
0
10
20
30
40
50
Test Data Quality
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:43
23-10-2022

```
In [40]: #training with LSTM algorithm with attention as multifactor
         multi_model = Sequential()
         multi_model.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1],X_train.shape[2])))
         multi_model.add(attention(return_sequences=True,name='attention')) #adding attention layer as multifactor
         multi_model.add(Dropout(0.2))
         multi_model.add(LSTM(units = 50, return_sequences = True))
         multi_model.add(Dropout(0.2))
         multi_model.add(LSTM(units = 50, return_sequences = True))
         multi_model.add(Dropout(0.2))
         multi_model.add(LSTM(units = 50))
         multi_model.add(Dropout(0.2))
         multi_model.add(Dense(units = 1))
         multi_model.compile(optimizer = 'adam', loss = 'mean_squared_error')
         if os.path.exists('model/multi_model.h5'):
             multi_model = load_model('model/multi_model.h5', custom_objects={'attention': attention})
             print(multi_model.summary())
         else:
             hist = multi_model.fit(X, Y, epochs = 700, batch_size = 16, validation_data=(X_test, y_test))
             multi_model.save('model/multi_model.h5')
             f = open('model/multi_history.pckl', 'wb')
             pickle.dump(hist.history, f)
             f.close()

         Model: "sequential_1"
         _________________________________________________________________
         Layer (type)                 Output Shape              Param #
         =================================================================
         lstm_1 (LSTM)                (None, 8, 50)             10400
```

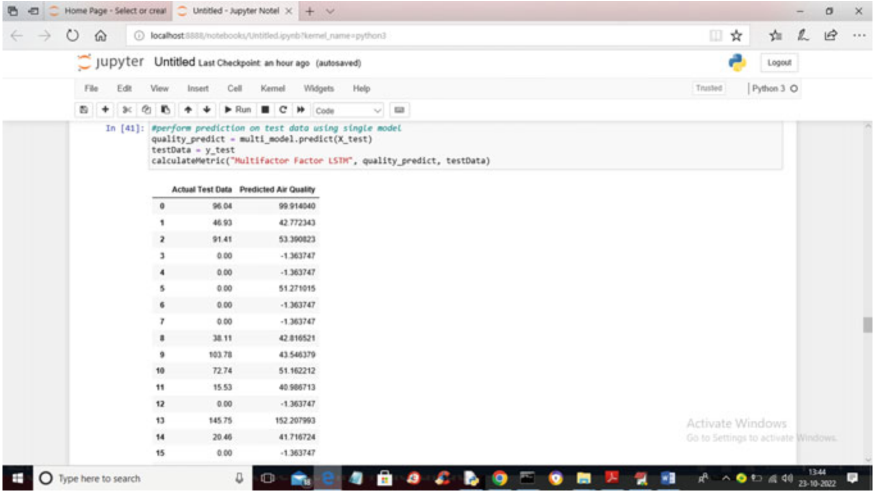
Fig. 12 Above screen we are defining multi-model by combining LSTM and attention layer and below is the multi-model summary



```
    pickle.dump(hist.history, f)
    f.close()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 8, 50)             10400
_________________________________________________________________
attention_4 (attention)      (None, 8, 50)             58
_________________________________________________________________
dropout_1 (Dropout)          (None, 8, 50)             0
_________________________________________________________________
lstm_2 (LSTM)                (None, 8, 50)             20200
_________________________________________________________________
dropout_2 (Dropout)          (None, 8, 50)             0
_________________________________________________________________
lstm_3 (LSTM)                (None, 8, 50)             20200
_________________________________________________________________
dropout_3 (Dropout)          (None, 8, 50)             0
_________________________________________________________________
lstm_4 (LSTM)                (None, 50)                20200
_________________________________________________________________
dropout_4 (Dropout)          (None, 50)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 51
=================================================================
Total params: 71,109
Trainable params: 71,109
Non-trainable params: 0
_________________________________________________________________
None
```

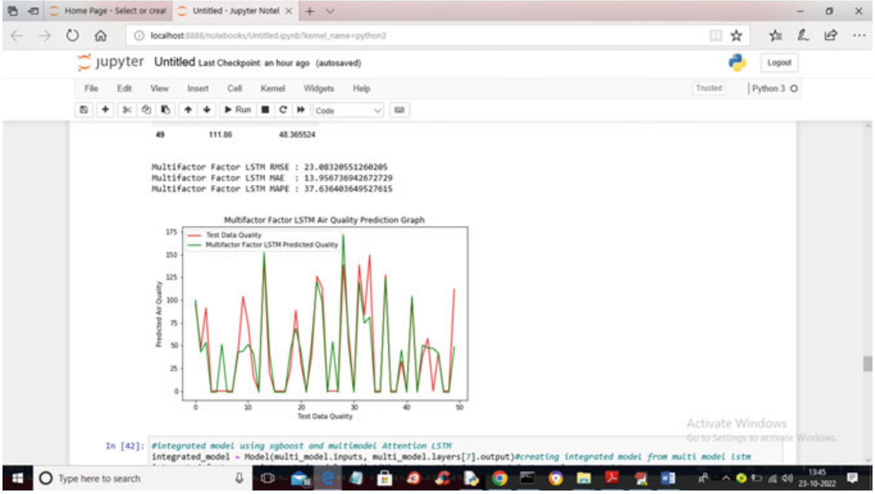
Fig. 13 Above screen LSTM is combined with attention and below is the multi-model predicted output



```
In [41]: #perform prediction on test data using single model
         quality_predict = multi_model.predict(X_test)
         testData = y_test
         calculateMetric("Multifactor Factor LSTM", quality_predict, testData)
```

	Actual Test Data	Predicted Air Quality
0	96.04	99.914040
1	46.93	42.772343
2	91.41	53.390823
3	0.00	-1.363747
4	0.00	-1.363747
5	0.00	51.271015
6	0.00	-1.363747
7	0.00	-1.363747
8	38.11	42.816521
9	103.78	43.546379
10	72.74	51.162212
11	15.53	40.986713
12	0.00	-1.363747
13	145.75	152.207993
14	20.46	41.716724
15	0.00	-1.363747

Fig. 14 Above screen we can see test data and multi-model predicted air quality for next 50 s



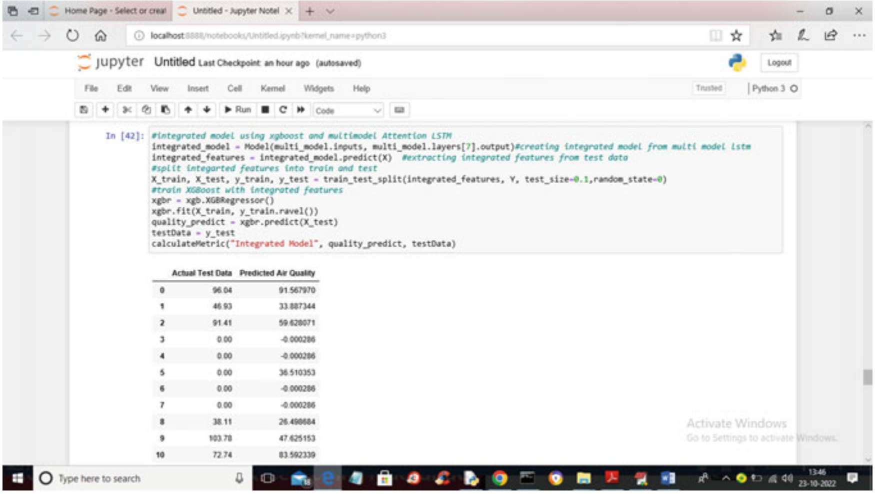
```
49        111.86        48.365524

Multifactor Factor LSTM RMSE : 23.08320551260205
Multifactor Factor LSTM MAE  : 13.956736942672729
Multifactor Factor LSTM MAPE : 37.636403649527615
```

jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Multifactor Factor LSTM Air Quality Prediction Graph
Test Data Quality
Multifactor Factor LSTM Predicted Quality
Predicted Air Quality
Test Data Quality
0
25
50
75
100
125
150
175
0
10
20
30
40
50
Type here to search
13:45
23-10-2022

```
In [42]: #integrated model using xgboost and multimodel Attention LSTM
         integrated_model = Model(multi_model.inputs, multi_model.layers[7].output)#creating integrated model from multi model lstm
```

Fig. 15 Above screen with multi-model we got RMSE as 23 and MAE as 13 and MAPE as 37, and we can see both test data and predicted air quality in graph. Above model RMSE, MAE, and MAPE is lesser than single model. In below screen showing integrated model with XGBOOST

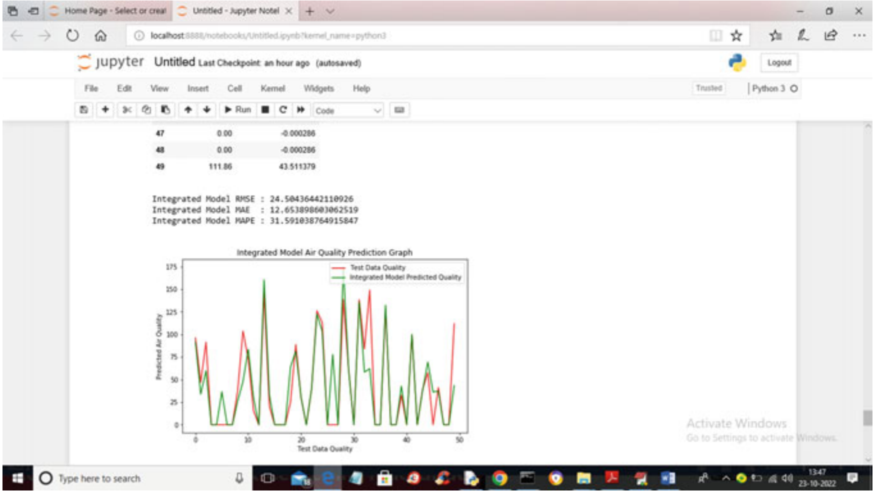


```
In [42]: #integrated model using xgboost and multimodel Attention LSTM
         integrated_model = Model(multi_model.inputs, multi_model.layers[7].output)#creating integrated model from multi model lstm
         integrated_features = integrated_model.predict(X)  #extracting integrated features from test data
         #split integarted features into train and test
         X_train, X_test, y_train, y_test = train_test_split(integrated_features, Y, test_size=0.1,random_state=0)
         #train XGBoost with integrated features
         xgbr = xgb.XGBRegressor()
         xgbr.fit(X_train, y_train.ravel())
         quality_predict = xgbr.predict(X_test)
         testData = y_test
         calculateMetric("Integrated Model", quality_predict, testData)
```

	Actual Test Data	Predicted Air Quality
0	96.04	91.567970
1	46.93	33.887344
2	91.41	59.628071
3	0.00	-0.000286
4	0.00	-0.000286
5	0.00	36.510353
6	0.00	-0.000286
7	0.00	-0.000286
8	38.11	26.498684
9	103.78	47.625153
10	72.74	83.592339

jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search
13:46
23-10-2022

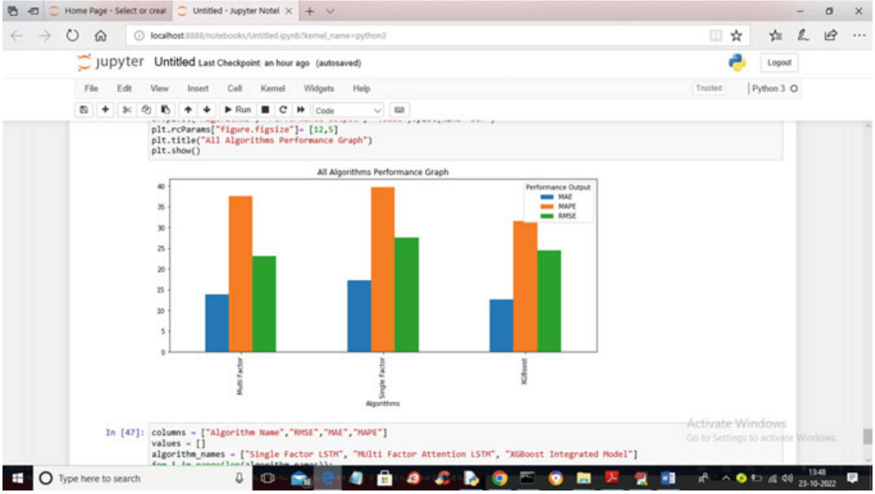
Fig. 16 Above screen showing result of integrated XGBOOST



```
47        0.00       -0.000286
48        0.00       -0.000286
49      111.86       43.511379

Integrated Model RMSE : 24.50436442110926
Integrated Model MAE  : 12.653898603062519
Integrated Model MAPE : 31.591038764915847
```

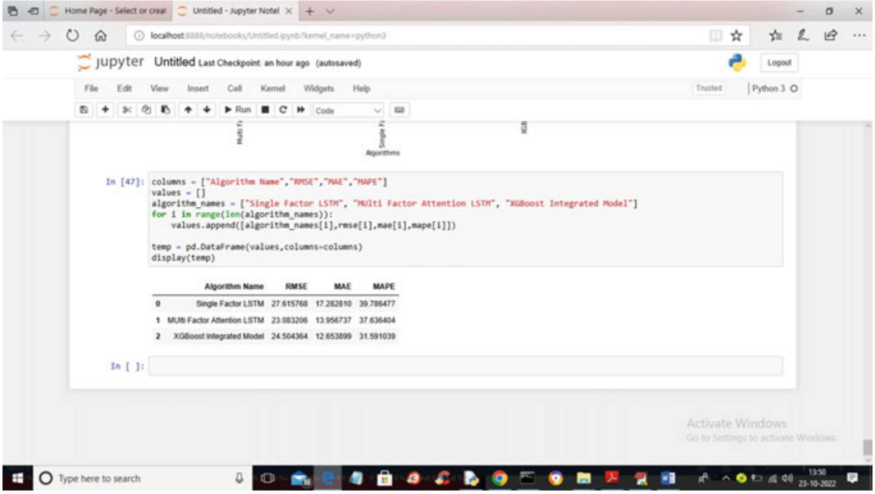
Fig. 17 Above screen with integrated model we got RMSE as 24 and MAE as 12, and MAPE as 31 and we can see predicted and actual test values in graph. In integrated model we got RMSE as high but MAE and MAPE is lesser than single and multi-model. In below screen showing graph of all algorithms



```
plt.rcParams["figure.figsize"]= [12,5]
plt.title("All Algorithms Performance Graph")
plt.show()
```

```
In [47]: columns = ["Algorithm Name","RMSE","MAE","MAPE"]
         values = []
         algorithm_names = ["Single Factor LSTM", "MUlti Factor Attention LSTM", "XGBoost Integrated Model"]
```

Fig. 18 Represents values where each different colour bar represents different metric such as RMSE, MAE, and MAPE and in above graph we can see integrated XGBOOST got less MAE and MAPE compared with all other algorithms and same output we can see in below tabular format


Home Page - Select or creat
Untitled - Jupyter Notel
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Multi Fa
Single Fa
XGB
Algorithms
Activate Windows
Go to Settings to activate Windows.
Type here to search
13:50
23-10-2022

```
In [47]: columns = ["Algorithm Name","RMSE","MAE","MAPE"]
         values = []
         algorithm_names = ["Single Factor LSTM", "MUlti Factor Attention LSTM", "XGBoost Integrated Model"]
         for i in range(len(algorithm_names)):
             values.append([algorithm_names[i],rmse[i],mae[i],mape[i]])

         temp = pd.DataFrame(values,columns=columns)
         display(temp)
```

```
               Algorithm Name       RMSE        MAE       MAPE
0          Single Factor LSTM  27.615768  17.282810  39.786477
1  MUlti Factor Attention LSTM  23.083206  13.956737  37.636404
2    XGBoost Integrated Model  24.504364  12.653899  31.591039
```

```
In [ ]:
```

Fig. 19 Above table we can see metric values of all algorithms and integrated XGBOOST got better result with low error rate

8 Conclusion

We suggested a prediction model based on integrated dual LSTM model method to increase the precision of air quality data prediction. The integrated model's realisation procedure and impact can be summed up as follows. The air quality characteristics in the model are taken into consideration, together with meteorological information and data from surrounding stations. The method tree is then used to integrate the two models. First, single-factor models for each factor in the temporal dimension were made. To obtain the forecasted outcomes, the temporal dimension's attributes are employed. The projected value and weight of each leaf node are put together to provide the ideal expected value. Since the technique outlined in this study is based on analysing the experimental data using five evaluation indicators, it can result in predictions that are more accurate.

In order to improve the accuracy of by integrating the advantages of various models, the integrated dual LSTM model technique will be expanded in the next phase of the study. Although our model's outputs have very low probability, we have also found certain prediction results with outlier values. The examination of this sort of outlier value is one of the concerns that has to be addressed in the feature scope.

References

1. Petr H, Vladimir O (2013) Prediction of air quality indices by neural networks and fuzzy inference systems. Commun Comput Inf Sci 383:302–312. https://doi.org/10.1007/978-3-642-41013-0_31
2. Kang Z, Qu Z (2017) Application of BP neural network optimized by genetic simulated annealing algorithm to prediction of air quality index in Lanzhou. In: Proc. IEEE Comput. Intell. Appl. (ICCIA), Sep. 2017, pp. 155–160. https://doi.org/10.1109/CIAPP.2017.8167199
3. Wang X, Wang B (2019) 'Research on prediction of environmental aerosol and PM2.5 based on artificial neural network.' Neural Comput Appl 31(12):8217–8227. https://doi.org/10.1007/s00521-018-3861-y
4. T. S. Rajput and N. Sharma, "Multivariate regression analysis of air quality index for Hyderabad city: Forecasting model with hourly frequency," Int. J. Appl. Res., vol. 3, no. 8, pp. 443–447, 2017. Accessed: Mar. 20, 2021. [Online]. Available: https://www.allresearchjournal.com/archives/2017/vol3iss ue8/PartG/3–8- 78–443.pdf
5. Mahajan S, Liu H-M, Tsai T-C, Chen L-J (2018) 'Improving the accuracy and efficiency of PM2.5 forecast service using cluster-based hybrid neural network model.' IEEE Access 6:19193–19204. https://doi.org/10.1109/ACCESS.2018.2820164
6. Li R, Dong Y, Zhu Z, Li C, Yang H (2019) 'A dynamic evaluation framework for ambient air pollution monitoring.' Appl Math Model 65:52–71. https://doi.org/10.1016/j.apm.2018.07.052
7. Liu B, Yan S, Li J, Qu G, Li Y, Lang J, Gu R (2019) 'A sequenceto-sequence air quality predictor based on the n-step recurrent prediction.' IEEE Access 7:43331–43345. https://doi.org/10.1109/ACCESS.2019.2908081
8. Gu K, Qiao J, Lin W (2018) 'Recurrent air quality predictor based on meteorology- and pollution-related factors.' IEEE Trans. Ind. Informat. 14(9):3946–3955. https://doi.org/10.1109/TII.2018.2793950
9. Benhaddi M, Ouarzazi J (2021) 'Multivariate time series forecasting with dilated residual convolutional neural networks for urban air quality prediction.' Arabian J. Sci. Eng. 46(4):3423–3442. https://doi.org/10.1007/s13369-020-05109-x
10. Song X, Huang J, Song D (2019) Air quality prediction based on LSTM-Kalman model. In: Proceedings IEEE 8th Joint Int. Inf. Technol. Artif. Intell. Conf. (ITAIC), Chongqing, China, May 2019, 695–699. https://doi.org/10.1109/ITAIC.2019.8785751