

# **Temporal forecasting of Sensor Measurements in Air quality Monitoring System**

**MINI PROJECT REPORT**

*Submitted by*

**Rangashri V (193002082)  
Sandhya B (193002092)  
Sam DJ (193002090)**

**UEC1605**

**MACHINE LEARNING**



**Department of Electronics and Communication  
Engineering**

**Sri Sivasubramaniya Nadar College of Engineering**

(An Autonomous Institution, Affiliated to Anna University)

**Rajiv Gandhi Salai (OMR), Kalavakkam – 603 110**

**EVEN SEM 2021-2022**

**Sri Sivasubramaniya Nadar College of Engineering**  
**(An Autonomous Institution, Affiliated to Anna University)**

**BONAFIDE CERTIFICATE**

Certified that this mini project titled “**Temporal forecasting of Sensor Measurements in Air quality Monitoring System**” is the bonafide work of “**Rangashri V (193002082), Sam DJ (193002090) and Sandhya B (193002092)**” of VI Semester Electronics and Communication Engineering Branch during Even Semester 2021 – 2022 for UEC1605 Machine Learning

Submitted for examination held on 11/06/2022

**INTERNAL EXAMINER**

## ABSTRACT

This project is a Temporal forecasting of Sensor Measurements in the Air quality Monitoring System. We started with collecting sensor data from the Air Quality Monitoring system. Next comes, data preprocessing and data cleaning. Then various data visualization techniques were performed in order to understand patterns in the dataset. Then the dataset was split into a training and testing set. On the training set N-BEATS gave us the minimal average error percentage. The performance metrics such as MAE (Mean absolute Error) per day and RMSE (Root mean square errors) were applied on the forecasted data in order to predict the accuracy of the forecasted data. We inferred that physical quantities such as Temperature and Relative Humidity have a regular pattern or seasonality which makes our model to easily predict the next 24/12 hour forecasting window. Although the chemical composition of air follows periodicity, there are several exceptions, for example a random holiday can drastically reduce the emission of NO<sub>x</sub> and CO<sub>x</sub> during 8-10 AM which is usually peak travel hours for work.

## **TABLE OF CONTENTS**

<b>CHAPTER NO</b>	<b>TITLE</b>	<b>PAGE NO</b>
1	<b>INTRODUCTION</b>	6
2	<b>LITERATURE SURVEY</b>	7
3	<b>N BEATS ALGORITHM</b>	10
	3.1 Introduction	10
	3.2 Model Architecture	10
	3.3 Model Parameters	13
4	<b>COURSE OF ACTION</b>	15
	4.1 Dataset Description	15
	4.2 Flowchart	16
	4.3 Preprocessing	17
	4.4 Data Visualization	19
5	<b>RESULTS</b>	
	5.1 Introduction	20
	5.2 Inference	25
	5.3 Conclusion	25
	<b>REFERENCES</b>	26

**LIST OF FIGURES**

<b>Figure no</b>	<b>Content</b>	<b>Page no.</b>
3.1	Complete architecture of the neural network	10
3.2	Various Components of time series analysis	12
4.1	Flow Chart	18

**LIST OF TABLES**

<b>Table no</b>	<b>Content</b>	<b>Page no</b>
5.1	Data of pollutant quantity	25

## **CHAPTER 1**

### **INTRODUCTION**

India ranks 5th globally among 98 countries in case of air pollution. Day by day the air pollution becomes a serious concern in India as well as in the overall world.

Proper or accurate prediction or forecast of Air Quality or the concentration level of other Ambient air pollutants such as Sulfur Dioxide, Nitrogen Dioxide, Carbon Monoxide, Particulate Matter having diameter less than  $10\mu$ , Ozone, etc. becomes a necessity because the impact these factors have on human health is huge.

This review focuses on the various techniques used for prediction or modeling of Air Quality Index (AQI) and forecasting of future concentration levels of pollutants. We forecasted these values for 24 hours and got the results.

Thus, with the inferences that we could make, we can study the existing policies to reduce the air pollution and come up with solutions such as carpooling, avoiding stepping out when pollution levels are high etc.

## CHAPTER 2

### LITERATURE SURVEY

This section presents previous work related to our proposed method.

**Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, Yoshua Bengio** “N-Beats: Neural Basis Expansion Analysis For Interpretable Time Series Forecasting”-a deep neural architecture based on backward and forward residual links and a very deep stack of fully-connected layers, tested the proposed architecture on several well-known datasets, including M3, M4 and TOURISM competition datasets containing time series from diverse domains and demonstrated how the proposed architecture can be augmented to provide outputs that are interpretable without considerable loss in accuracy.

Huixiang Liu (et al.2019) has taken the city of Beijing for the study purpose. They have forecasted the Air Quality Index (AQI) for the city Beijing and predicting the concentration of NO<sub>x</sub>.

**Heidar Maleki (et al.2019)** predicted the hourly concentration values for the ambient air pollutants NO<sub>2</sub>, SO<sub>2</sub>, PM<sub>10</sub>, PM<sub>2.5</sub>, CO and O<sub>3</sub> for the stations in Iran

**Nidhi Sharma (et al.2018)** had gone through the detailed data analysis

of air pollutants in New Delhi. By using data analytics Time series Regression forecasting they have predicted the future values of the pollutants mentioned earlier on the of previous records.

**S.Tikhe Shruti (et al.2013)** the research employed two soft computing algorithms Artificial Neural Network (ANN) for the prediction of future concentration levels of air pollutants such as Oxides of Sulfur and Nitrogen and also Respirable Suspended Particulate Matter (RSPM) over the year 2005 to 2011 for the city of Pune.

**Mohamed Shakir and N.Rakesh (2018)** have analysed the proportion of various air pollutants (NO, NO<sub>2</sub>, CO, PM<sub>10</sub> and SO<sub>2</sub>) with respect to the time of the day and the day of the week and estimated the effect of environmental parameters as temperature, wind speed and humidity on the air pollutants. The dataset was collected over the district of Karnataka.

Darts is a Python library for easy manipulation and forecasting of time series. The models can all be used in the same way, using fit() and predict() functions, similar to scikit-learn. The library also makes it easy to backtest models, combine the predictions of several models, and take external data into account. Darts supports both univariate and multivariate time series and models.

Interpretable forecasting with N-Beats - generates a synthetic dataset to demonstrate the network's capabilities. Baseline model predicts future values by repeating the last known value. The key hyperparameter of the NBeats model are the widths. Each denotes the width of each forecasting



block. Predict the validation dataset with `predict()` and calculate the error. Ask PyTorch Forecasting to decompose the prediction into seasonality and trend with `plot_interpretation()`.

## **CHAPTER 3**

### **N-BEATS ALGORITHM**

#### **3.1 INTRODUCTION**

Neural Basis Expansion Analysis for Interpretable Time Series (or) N-BEATS model is a neural network consisting of a sequence of stacks, each of which combine multiple blocks. The blocks connect feedforward networks via forecast and backcast links. A block removes the portion of the signal which it can approximate well and later sets its focus on the residual error, which the preceding blocks could not disentangle.

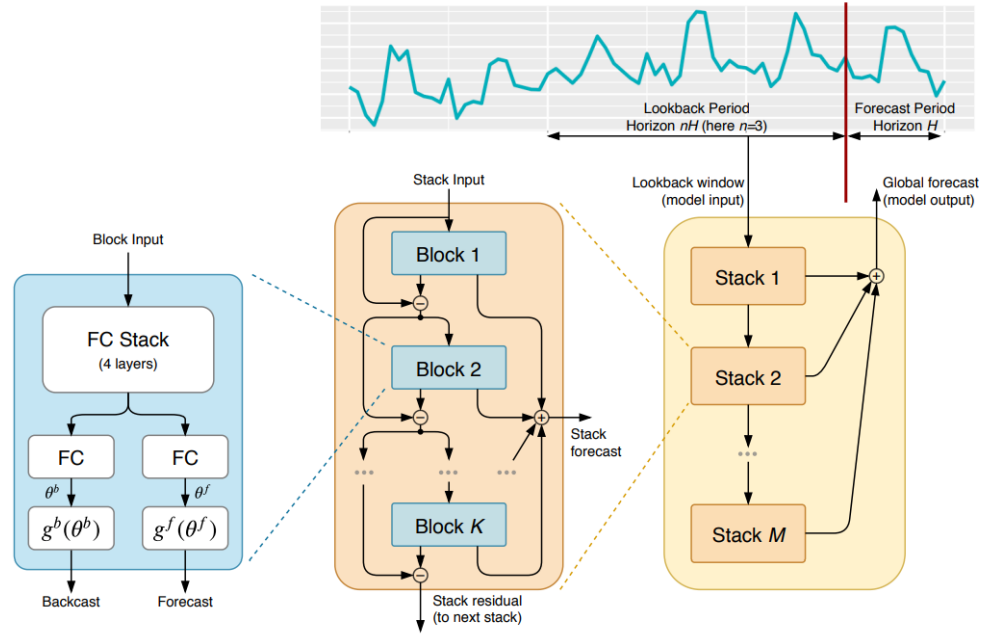
Each block generates a partial forecast, with its focus set on the local characteristics of the time series. The stack aggregates the partial forecasts across the blocks it comprises and then hands the result over to the next stack. The stacks' purpose is to identify non-local patterns along the complete time axis by looking back at the dataset. Finally, the partial forecasts are pieced together to a global forecast at the model level.

#### **3.2 MODEL ARCHITECTURE**

The authors have proposed that N-BEATS works better because of meta-learning. In meta-learning, the learning process can be decomposed into an inner and outer training loop.

The inner training loop focuses on task specific knowledge and the outer loop focuses on across-task knowledge. We can analogize this to

N-BEATS, where  $\theta$  is being learnt inside the blocks and makes use of the parameters that are learnt from the outer loop, where gradient descent trains the weight matrices that  $\theta$  depends on.



**Fig 3.1. Complete architecture of the neural network represented as a stack of blocks**

This basic building block is a multi-layer FC network with RELU nonlinearities. It predicts basis expansion coefficients both forward,  $\theta^f$ , (forecast) and backward,  $\theta^b$ , (backcast). Blocks are organized into stacks using doubly residual stacking principle. A stack may have layers with shared  $g^b$  and  $g^f$ . Forecasts are aggregated in hierarchical fashion. This enables building a very deep neural network with interpretable outputs. The second part consists of the backward  $g^b$  and the forward  $g^f$  basis layers that accept the respective forward  $\theta^f$  and backward  $\theta^b$  expansion coefficients and project them internally on the set of basis functions. The

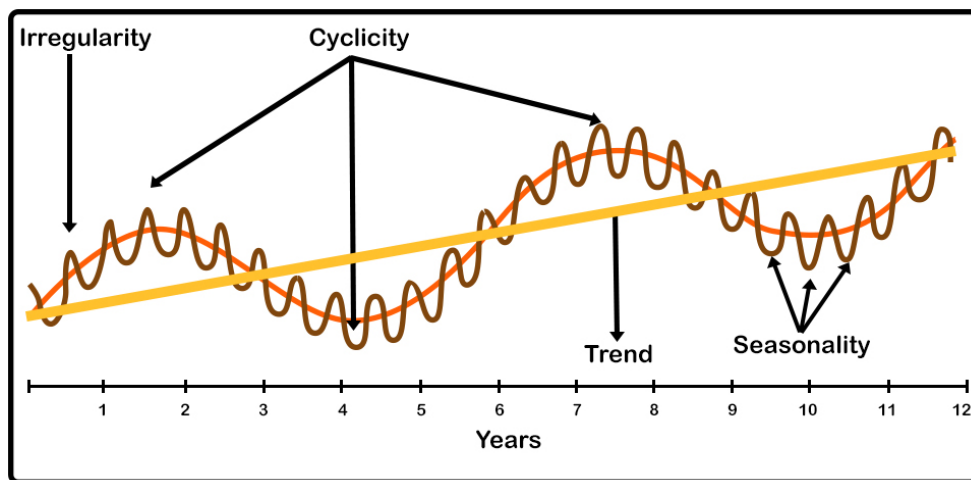
network maps expansion coefficients  $\theta^f$  and  $\theta^b$  to outputs via basis layers, then the block gives out the best estimates, in the form of two outputs, forecast and backcast through the following operations:

$$\hat{\mathbf{y}}_\ell = \sum_{i=1}^{\dim(\theta_\ell^f)} \theta_{\ell,i}^f \mathbf{v}_i^f, \quad \hat{\mathbf{x}}_\ell = \sum_{i=1}^{\dim(\theta_\ell^b)} \theta_{\ell,i}^b \mathbf{v}_i^b.$$

The next block is the residual block, which has two residual branches, one running over backcast prediction of each layer and the other one is running over the forecast branch of each layer. Its operation is described by the following equations:

$$\mathbf{x}_\ell = \mathbf{x}_{\ell-1} - \hat{\mathbf{x}}_{\ell-1}, \quad \hat{\mathbf{y}} = \sum_\ell \hat{\mathbf{y}}_\ell.$$

The trend and seasonality decomposition are designed into the model to make the stack outputs more easily interpretable.



**Fig 3.2. Various components of Time Series Analysis**

A typical characteristic of trend is that most of the time it is a monotonic function, or at least a slowly varying function, whereas seasonality is a regular, cyclical, recurring fluctuation.

With a partial forecast of block 1 within stack  $s$  in hand, one can mimic

delthe behaviour of Trend by making  $gb_{s,l}^b$  and  $gf_{s,l}^b$  as a polynomial of small degree  $p$ , i.e. a function that slowly varying across forecast window:

$$\hat{y}_{s,\ell} = \sum_{i=0}^p \theta_{s,\ell,i}^f t^i.$$

Similarly for Seasonality, we constraint  $gb_{s,l}^b$  and  $gf_{s,l}^b$  to belong to a class of periodic functions. Therefore a natural choice for basis to model a periodic function is the Fourier series:

$$\hat{y}_{s,\ell} = \sum_{i=0}^{\lfloor H/2-1 \rfloor} \theta_{s,\ell,i}^f \cos(2\pi it) + \theta_{s,\ell,i+\lfloor H/2 \rfloor}^f \sin(2\pi it),$$

### 3.3 MODEL PARAMETERS

- The sizes of the input and output layers should be adequate to assign a node to each feature in the source data. The Input chunk length should not be smaller than the order of seasonality, or else the learning process will find it more difficult to combine the pieces. For efficient memory usage, set them to a power of 2.
- The number of blocks in a stack (BLOCKS).
- The width of each fully connected layer in each block of a stack: its node count (LWIDTH).
- The batch size defines the number of observations the model will process before it updates its matrix weights. To effectively align it with the memory structure of your system, set it to a power of 2.

Very large batch sizes may mislead the gradient descent in a single direction — the model may dig itself into a suboptimal minimum. Smaller batch sizes will cause the gradient descent to bounce around in different directions and can result in lower accuracy, but they also tend to prevent the model from overfitting. The most frequent recommendation is to choose an initial batch size of 32. Since our dataset has a frequency of 24 daily hours, I set the batch size to the next binary ceiling that can process 24 time steps: 32.

- The epochs tell the model how many training cycles it is supposed to run. During each epoch, the model will process the entire training set, making one forward and one backward pass.
- Allowing for some oversimplification, the product of these hyperparameters defines the tensor size of the model. Large parameter values can make it bump against the memory limit of one's system and will lead to exponentially longer processing times. Whereas small parameter values may turn out to be inadequate to mirror complex patterns in the source data.

## **CHAPTER 4**

### **DATASET ACQUISITION AND WORKFLOW**

#### **4.1 DATASET DESCRIPTION**

This dataset contains the responses of a gas multisensor device. Hourly responses averages are recorded along with gas concentrations references from a certified analyzer. This dataset was taken from Kaggle.

The dataset contains 9357 instances of hourly averaged responses from an array of 5 metal oxide chemical sensors embedded in an Air Quality Chemical Multisensor Device. The device was located on the field in a significantly polluted area, at road level, within an Italian city. Data were recorded from March 2004 to February 2005 (one year) representing the longest freely available recordings of on field deployed air quality chemical sensor devices responses. Ground Truth hourly averaged concentrations for CO, Non-Metallic Hydrocarbons, Benzene, Total Nitrogen Oxides (NO<sub>x</sub>) and Nitrogen Dioxide (NO<sub>2</sub>) and were provided by a co-located reference certified analyzer. Missing values are tagged with -200 values.

#### **4.2 ATTRIBUTE INFORMATION**

The dataset consists of 15 attributes:

1. Date (DD/MM/YYYY)
2. Time (HH.MM.SS)

3. True hourly averaged concentration CO in  $\text{mg/m}^3$  (reference analyzer)
4. PT08.S1 (tin oxide) hourly averaged sensor response
5. True hourly averaged overall Non Metanic HydroCarbons concentration in  $\text{micro g/m}^3$  (reference analyzer)
6. True hourly averaged Benzene concentration in  $\text{microg/m}^3$  (reference analyzer)
7. PT08.S2 (titania) hourly averaged sensor response (nominally NMHC targeted)
8. True hourly averaged NOx concentration in ppb (reference analyzer)
9. PT08.S3 (tungsten oxide) hourly averaged sensor response (nominally NOx targeted)
10. True hourly averaged NO2 concentration in  $\text{microg/m}^3$  (reference analyzer)
11. PT08.S4 (tungsten oxide) hourly averaged sensor response (nominally NO2 targeted)
12. PT08.S5 (indium oxide) hourly averaged sensor response (nominally O3 targeted)
13. Temperature in  $^{\circ}\text{C}$
14. Relative Humidity (%)
15. AH Absolute Humidity

Since the concentration of the same component is measured through multiple sensors, a few of them were dropped for not choking the memory while training.

### 4.3 FLOWCHART



Figure 4.1 describes the flow of the project. First step is the Collection of sensor data from the Air Quality Monitoring system.

Second step is data preprocessing and data cleaning. Data preprocessing is the process of transforming raw data into an understandable format. Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset.

Then various data visualization techniques are performed in order to understand patterns in the dataset.

Then the dataset is split into training and testing set in the ratio of 9:1. On the training set various temporal forecasting algorithms like ARIMA, SARIMA, N-BEATS algorithms are applied, out of which N-BEATS gave us the minimal average error percentage.

Performance metrics such as MAE (Mean absolute Error) per day and RMSE (Root mean square errors) are applied on the forecasted data in order to predict the accuracy of the forecasted data.

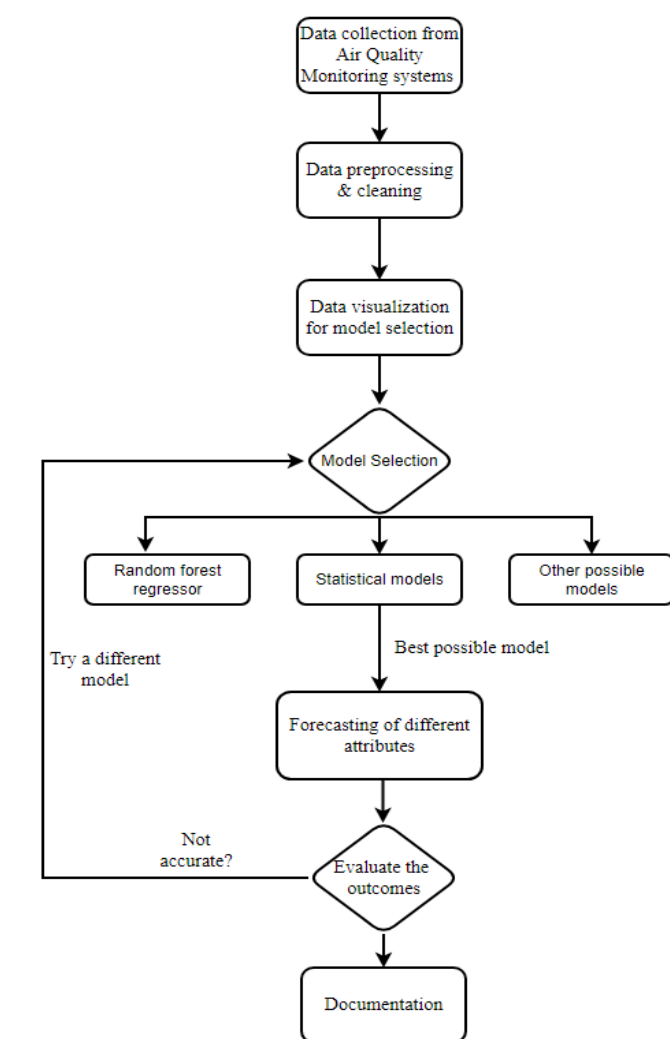
This process of training with different algorithms and testing using various Performance metrics is done until an accurate model has been identified.

#### **4.4 DATASET PREPROCESSING**

For unknown reasons, the sensors instead of returning a nan or NULL value, chose to fill the dataset with -200 whenever it fails to take a reading. This abrupt value is also illogical since the concentration of

chemical compounds can never be negative.

Therefore, using Pandas, we trace out all negative values and replace them with the mode of that particular chemical. Moreover, the decimal points of the values are replaced with a comma, this can be also replaced by an appropriate data cleaning methodology.



**Figure 4.1: Flowchart**

## 4.5 PERFORMANCE METRICS

The performance metrics used in this project are Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE).

**Mean absolute percentage error:**

$$M = \frac{1}{n} \sum_{t=1}^n \left| \frac{A_t - F_t}{A_t} \right|$$

$n$  is the number of fitted points,

$A_t$  is the actual value,

$F_t$  is the forecast value.

$\Sigma$  is summation notation (the absolute value is summed for every forecasted point in time)

**Root Mean Square Error/Deviation:**

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^N (x_i - \hat{x}_i)^2}{N}}$$

$i$  = variable  $i$

$N$  = number of non-missing data points

$x_i$  = actual observations time series

$\hat{x}_i$  = estimated time series

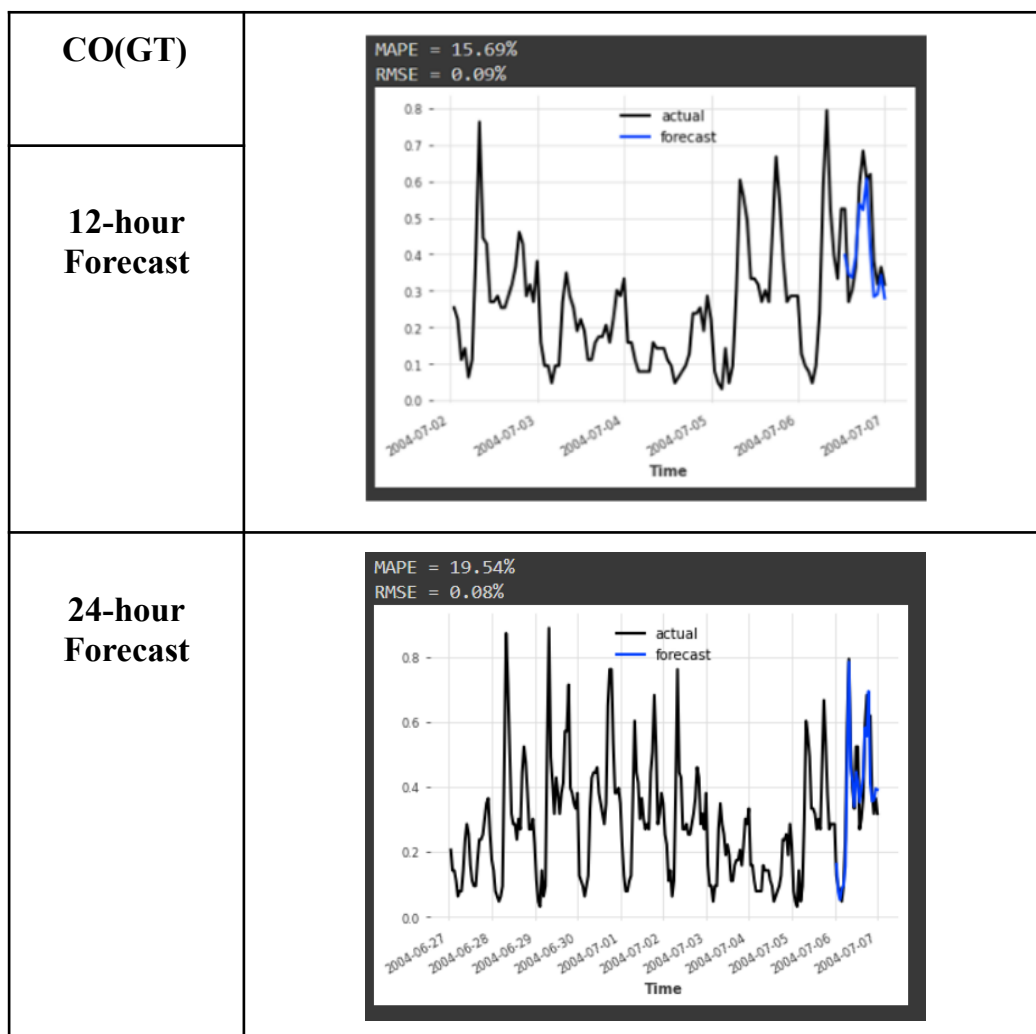
## CHAPTER 5

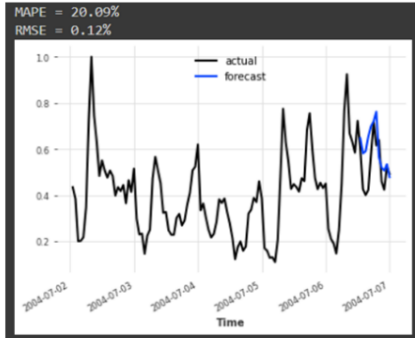
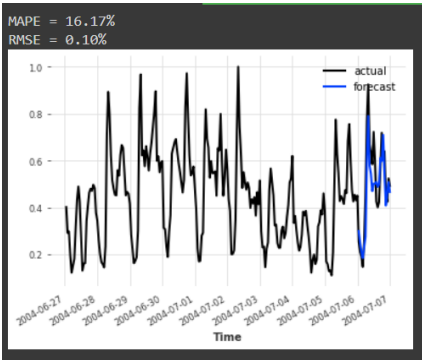
### RESULTS AND INFERENCE

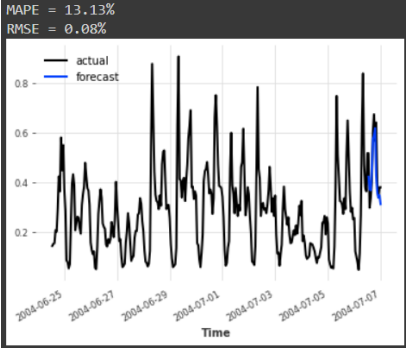
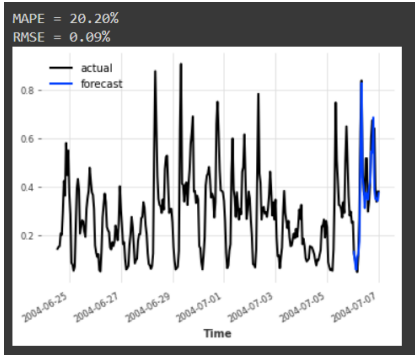
#### 5.1 INTRODUCTION

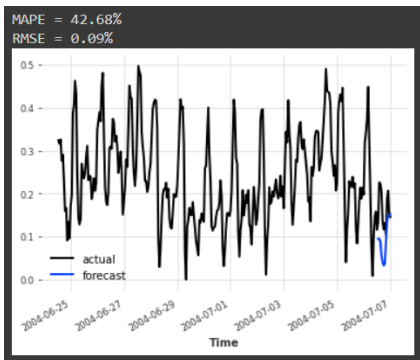
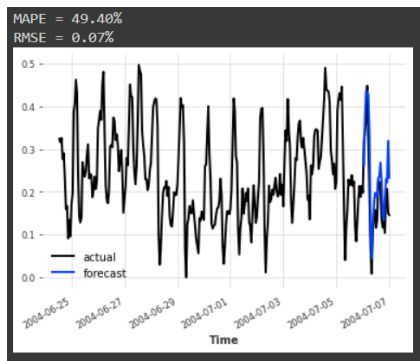
With the N-BEATS algorithm trained with 100 epochs, a 12-hour and a 24-hour forecast window was put to test, each varying in their corresponding training sets.

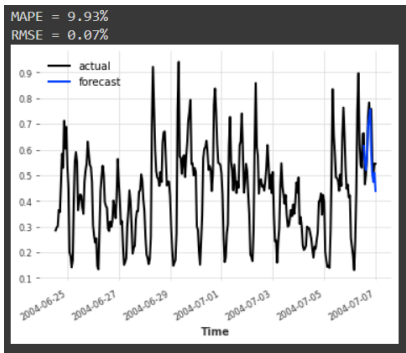
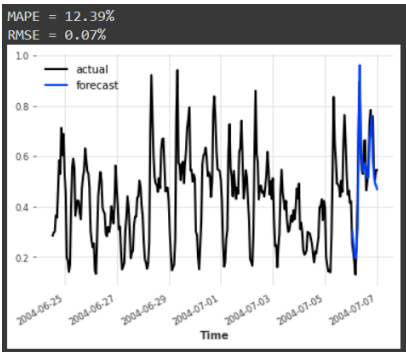
Both the forecasted values along with the existing test values were plotted. The measured errors are also shown in the Fig 5.1

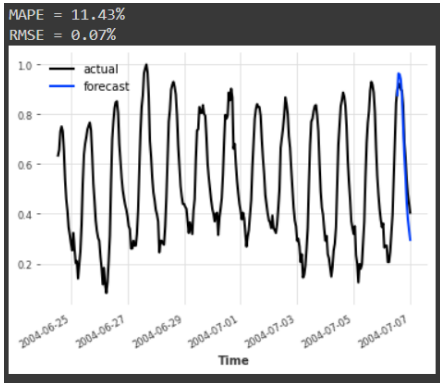
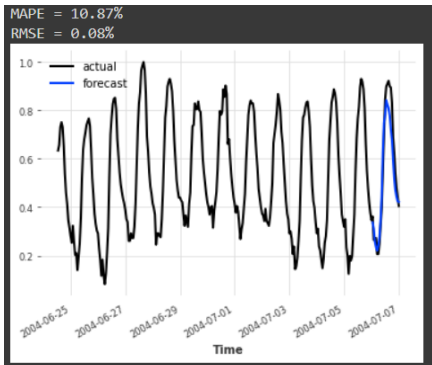


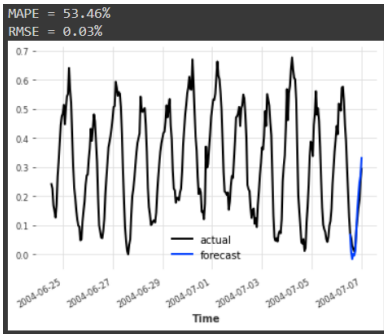
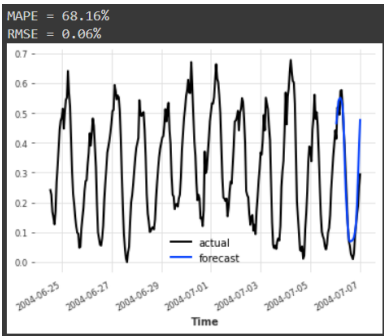
<b>PT08.S1 (CO)</b>	
<b>12-hour Forecast</b>	
<b>24-hour Forecast</b>	

<b>PT08.S2 (NMHC)</b>	
<b>12-hour Forecast</b>	
<b>24-hour Forecast</b>	

<b>PT08.S3 (NO<sub>x</sub>)</b>	
<b>12-hour Forecast</b>	
<b>24-hour Forecast</b>	

<b>C6H6(GT)</b>	
<b>12-hour Forecast</b>	
<b>24-hour Forecast</b>	

<b>Temperature</b>	 <p>MAPE = 11.43% RMSE = 0.07%</p>
<b>12-hour Forecast</b>	
<b>24-hour Forecast</b>	 <p>MAPE = 10.87% RMSE = 0.08%</p>

<b>Relative Humidity</b>	 <p>MAPE = 53.46% RMSE = 0.03%</p>
<b>12-hour Forecast</b>	
<b>24-hour Forecast</b>	 <p>MAPE = 68.16% RMSE = 0.06%</p>

<b>Pollutant/Physical Quantity</b>	<b>MAPE (12 hours)</b>	<b>MAPE (24 hours)</b>
CO(GT)	15.69%	19.54%
PT08.S1(CO)	20.09%	16.17%
PT08.S2 (NMHC)	13.13%	20.20%
PT08.S3 (NO <sub>x</sub> )	42.68%	42.40%
C <sub>6</sub> H <sub>6</sub> (GT)	9.93%	12.39%
Temperature	11.43%	10.87%
Relative Humidity	53.46%	68.16%

**Table 5.1- Data for pollutant quantity after 12 and 24 hours.**

## **5.2 INFERENCE:**

Physical quantities such as Temperature and Relative Humidity have a regular pattern or seasonality which makes our model to easily predict the next 24/12 hour forecasting window. Although the chemical composition of air follows periodicity, there are several exceptions, for example a random holiday can drastically reduce the emission of NO<sub>x</sub> and CO<sub>x</sub> during 8-10 AM which is usually peak travel hours for work.

## **5.3 CONCLUSION:**

Therefore, we are able to forecast the chemical composition and physical quantities such as Temperature successfully. The one limitation is that no such data is available for any Indian city. In future, we plan on collecting real data from our location so that it would be relevant, useful in India.



## REFERENCES

- [1] Archontoula Chaloulakou, Georgios Grivas, Nikolas Spyrellis, “Neural Network and Multiple Regression Models for PM10 Prediction in Athens: A Comparative Assessment”, Journal of the Air & Waste Management Association, 2012.
  
- [2] Boris N. Oreshkin, Dmitri Carpo, Nicolas Chapados, Yoshua Bengio : “N-Beats: Neural Basis Expansion Analysis For Interpretable Time Series Forecasting”, 1905.10437v4, ICLR 2020
  
- [3] Dixian Zhu, Changjie Cai, Tianbao Yang, Xun Zhou, “A Machine Learning Approach for Air Quality Prediction: Model Regularization and Optimization”, February 2018, Big Data and Cognitive Computing
  
- [4] Heidar Malek, Armin Sorooshian, Gholamreza Goudarzi, Zeynab Baboli, Yaser Tahmasebi Birgani, Mojtaba Rahmati, “Air pollution prediction by using an artificial neural network model”, Clean Technologies and Environmental Policy, (2019) 21:1341–1352
  
- [5] Huixiang Liu, Qing Li, Dongbing Yu, Yu Gu, “Air Quality Index and Air Pollutant Concentration Prediction Based on Machine Learning Algorithms”, Applied Sciences, ISSN 2076-3417; CODEN: ASPCC7, 2019, 9, 4069; doi:10.3390/app9194069
  
- [6] Mohamed Shakir, N. Rakesh, “Investigation on Air Pollutant Data Sets using Data Mining Tool”, IEEE Xplore Part Number:CFP18OZV-ART; ISBN:978-1- 5386-1442-6

[7] Nidhi Sharma , ShwetaTaneja , VaishaliSagar , Arshita Bhatt, “Forecasting air pollution load in Delhi using data analysis tools”, ScienceDirect, 132 (2018) 1077– 1085.

[8] S Makridakis, E Spiliotis, and V Assimakopoulos: “Statistical and machine learning forecasting methods: Concerns and ways forward”, PLoS ONE, 13(3), 2018.

[9] N-Beats, DARTS library Documentation:  
<https://unit8co.github.io/darts/index.html>

[10] Interpretable forecasting with N-Beats, PyTorch-Forecasting:  
<https://pytorchforecasting.readthedocs.io/en/stable/tutorials/ar.html>

[11] Air Quality Dataset:  
<https://www.kaggle.com/datasets/fedesoriano/air-quality-data-set?resource=download>