Predicting Air Quality using python

Mini Project Report

Submitted in partial fulfillment of the requirements for the award of the Degree of

Bachelor of Technology (B.Tech)

in

COMPUTER SCIENCE AND ENGINEERING

By

S. SAI SRIKARREDDY 15AG1A0556

E. SRIDHAR 15AG1A0511

K. MALLIKARJUN 15AG1A0520

Under the Esteemed Guidance of

Mrs. D. Ashwini

Department of Computer Science and Engineering

ACE ENGINEERING COLLEGE

(NBA Accredited B.Tech Courses: EEE, ECE, CSE)

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad, Telangana)

Ankushapur(V), Ghatkesar(M), R.R.Dist - 501 301

2018 - 2019

ACE ENGINEERING COLLEGE

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project report titled Predicting Air Quality using python is being submitted by S. Saisrikarreddy, bearing 15AG1A0556,K. Mallikarjun, bearing 15AG1A0520 and E. Sridhar, bearing 15AG1A0511, in B.Tech IV I semester *Computer Science & Engineering* to Jawaharlal Nehru Technological University Hyderabad is a record of bonafide work carried out to them.

Ms. D. Ashwini
Internal Guide

Prof. K. JAYA BHARATHI
Head Of Department

DECLARATION

We hereby declare that this project entitled "**Predicting Air Quality using python**" which is being submitted by us in partial fulfillment of the requirement for the award of the degree of B.Tech to the Jawaharlal Nehru Technological University Hyderabad, India either in part or full does not constitute any part of any project submitted by us or any other person to any University/Institute.

S. Saisrikarreddy	**15AG1A0556**
E. Sridhar	**15AG1A0511**
K. Mallikarjun	**15AG1A0520**

ACKNOWLEDGMENTS

I would like to express my gratitude to all the people behind the screen who have helped me transform an idea into a real time application.

I would like to express my heart-felt gratitude to my parents without whom I would not have been privileged to achieve and fulfill my dreams.

A special thanks to our Secretary, **Prof. Y. V. GOPALA KRISHNA MURTHY,** for having founded such an esteemed institution. I am also grateful to our beloved principal, **Dr. B. L. RAJU** for permitting us to carry out this project.

I profoundly thank **Prof K. JAYA BHARATHI**, Head of the Department of Computer Science and Engineering, who has been an excellent guide and also a great source of inspiration to my work.

I extremely thank **Mr. G. SREENIVASULU**, Associate Professor, Project coordinator, who helped us in all the way in fulfilling of all aspects in completion of our Mini-Project.

I am very thankful to my internal guide **Mrs. D. ASHWINI,** Associate Professor who has been an excellent and also given continuous support for the Completion of my project work.

The satisfaction and euphoria that accompany the successful completion of the task would be great, but incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success. In this context, I would like to thank all the other staff members, both teaching and non-teaching, which have extended their timely help and eased my task.

S. Sai Srikarreddy 15AG1A0556

K. Mallikarjun 15AG1A0520

E. Sridhar 15AG1A0511

INDEX

ABSTRACT

Air pollution is a major environmental health problem affecting the developing and the developed countries. Over the past years the development and urbanization in Hyderabad has led to increase in air pollution. The high growth rate of cities induces imbalances in weather patterns, with negative consequences to public health. This has led to study and research in this area. Air pollution occurs when the air contains gases, dust, fumes or odour in harmful amounts. That is, amounts which could cause damage to plants and materials. Air pollutants mainly occur as a result of gaseous discharges from industry and motor vehicles. Industry is another major contributor to air pollution in India. We have seen the trends of various air pollutants like sulphur dioxide(SO_2), nitrogen dioxide (NO_2), particulate matter (PM). By introducing better technology and industrial practices which enable compliance with standards set by Environmental protection agency(EPA). We apply analytical techniques to analyze the existing trends in air pollution in Hyderabad and forecast the year wise progress in coming years about the future. In addition to this, mitigation strategies will be provided.

CHAPTER 1

INTRODUCTION

Air pollution is a gas released in a big enough quantity to harm the health of people or animals, kill plants or stop them growing properly. It also damage or disrupt some other aspect of the environment such as making buildings crumble. The substances that cause air pollution are called pollutants. Pollutants that are pumped into our atmosphere and directly pollute the air are called primary pollutants. Air pollutants mainly occur due to gaseous discharges from industry and motor vehicles. The different types of pollutants are ozone, carbon monoxide, nitrogen oxides, sulphur dioxide, lead, PM_{10}, $PM_{2.5}$. In reality, air pollution occurs when any sort of contaminant is introduced into the atmosphere, thereby disrupting the chemical composition of atmosphere. The most obvious of these contaminants is carbon dioxide, which is frequently cited as the most pervasive “greenhouse gas” in the Earth’s atmosphere.

The effects of air pollution on health are very complex as there are many different sources and their individual effects vary from one to the other. It is not only the ambient air quality in the cities but also the indoor air quality in the rural and the urban areas that are causing concern. In fact in the developing world the highest air pollution exposures occur in the indoor environment. Air pollutants that are inhaled have serious impact on human health affecting the lungs and the respiratory system; they are also taken up by the blood and pumped all-round the body. These pollutants are also deposited on soil, plants, and in the water, further contributing to human exposure.

Air pollution means different problems at different scales. Solving a problem like passive smoking how one person's cigarette smoke can harm other people's health is very different to tackling a problem like global warming, though both involve air pollution and they do have some things in common both problems, for example, require us to think about how our behavior can affect other people in the short and long term and to act more considerately. Generally, air pollution is tackled by a mixture of technological solutions, laws and regulations, and changes in people's behavior.

1.1 Problem Statement:

The population of the world is on the rise and by 2020 is predicted to reach more than 7 billion. The increasing population is bound to lead to a significant rise in the number of vehicles on the road that in turn will lead to higher emissions of harmful particulates into the atmosphere.

The common particulates emitted into the atmosphere are PM10, PM2.5, CO, Nitrogen Oxides (NO+NO2) and Ozone. Inhaling these particulates will affect the normal lung development and lead to respiratory problems such as asthma, heart diseases, etc.

Recent studies have found that 3000-5000 premature deaths occur every year because of inhaling particulate matters present in the atmosphere.

Also, it would be difficult to reduce the pollution levels unless the emissions caused by on road traffic are restricted.

1.2 OVERVIEW

Monitoring and preserving air quality is one of the most essential activities in many industrial and urban areas today. The quality of air is adversely affected due to various forms of pollution caused by transportation, electricity, fuel uses etc. The deposition of harmful gases including carbon dioxide, nitrous oxide, methane etc. is creating a serious threat for the quality of life in smart cities. With increasing air pollution, we need to implement efficient air quality monitoring models which collect information about the concentration of air pollutants and provide assessment of air pollution in each area. Hence, air quality evaluation, monitoring, and prediction has become an important research area. In the past, many environmental researchers have dedicated their research efforts on this subject using conventional approaches.

However, the quality of air is affected by multi-dimensional factors including location, time, and uncertain variables. Recently, many researchers began to use the big data analytics approach for studying, evaluating, and predicting air quality due to the advancements in big data applications and the availability of environmental sensing networks and sensor data.

One of the prime concerns for researchers is the use of adequate modelling tools that permit interpretation and validation of the data collected from multiple resources regarding air pollution. The aim of this research paper is to investigate various big-data and machine learning based techniques for air quality forecasting in diverse conditions. This paper reviews the published research results relating to air quality evaluation and prediction using methods of artificial intelligence, decision trees, support vector machines, deep learning etc. to name a few. Moreover, the paper classifies and compares the applied big data analytics approaches and big data based prediction models for air quality assessment. Furthermore, the paper also throws light on some of the challenging areas and future research needs.

1.3 OBJECTIVES

The main objectives of this research project are to:

- Trend analysis of environment monitoring data by data mining technique.
- Identify the areas of high concentration of air pollutants in urban and industrial areas.
- Comparison of observed reading with National Ambient Air Quality standards.
- Assessment of the ambient air quality data in the Urban and Industrial areas using Time series models and Deep learning model.
- Recommendation to the decision makers based on data analysis.

1.4 SCOPE

Air quality monitoring is a prospective application domain which is of particular value to our country. Large cities with high concentration of industry, intensive transport networks and high population density are major sources of air pollution. Predicting air quality from multiple sources by using modeling is very complicated. So, air quality models are best used for isolated sources or situations. As per the World Bank report quoted earlier, industrial pollution in India is on the more alarming state than industrial production. Hence, controlling and monitoring air pollution round the clock is a social imperative. This study proves that WSN could be a useful mechanism for this double task. The air quality data generation through air quality monitoring network available today, involves large number of monitoring agencies, personal and equipment for sampling, chemical analysis and data reporting etc. The involvement of several agencies increases the probability of variations and personal biases reflecting on the data. Therefore, the air quality data statistics available today is being recognized to be more indicative rather than absolute and perfect.

1.5 PROPOSED SYSTEM:

A model is designed for future prediction of the air pollution and a suitable technique is implemented to train the model accurately. This simplifies the training, testing and predicting the pollution levels in the air. Also an Air Quality index and the future mitigation strategies are provided.

1.6 SYSTEM REQUIREMENTS

Hardware Requirements:-

RAM: 4 GB

Processor: Intel i3 or above

Hard disk: 20GB

Software Requirements:-

Operating System: Windows 7/8/10

Tools and Environment: Jupyter Notebook

Code Behind: Python and Tensorflow and keras

CHAPTER 2

LITERATURE REVIEW

The population of the world is on the rise and by 2020 is predicted to reach more than 7 billion. The increasing population is bound to lead to a significant rise in the number of vehicles on the road that in turn will lead to higher emissions of harmful particulates into the atmosphere. The common particulates emitted into the atmosphere are PM10, PM2.5, CO, Nitrogen Oxides (NO+NO2) and Ozone. Inhaling these particulates will affect the normal lung development and lead to respiratory problems such as asthma, heart diseases, etc.

Recent studies have found that 3000-5000 premature deaths occur every year because of inhaling particulate matters present in the atmosphere. Also, it would be difficult to reduce the pollution levels unless the emissions caused by on road traffic are restricted. A possible solution to reduce the pollution concentration is by creating awareness to the people on the causes and the harmful effect of the pollutant concentrations.

The technologies available today play a vital role in our day to day life and the dependence on it has greatly increased over the years. Therefore, incorporating the available technologies for creating awareness to the people is one of the possible solutions. One such technology that has been widely used in pollution-related projects is the use of pollution sensors that have the capacity to detect and distinguish each pollution particulate separately.

In recent years, the smart cities initiative is on the rise to mitigate the effect of pollution. This initiative comprises of many projects involved in protecting the city environment and also in reducing the pollution level of the city.

Air Quality Index (AQI)

Air Quality Index (AQI) is an index that provides the public with the level of pollution associated with its health effects. The AQI focuses on the various health

effects that people might experience based on the level and hours of exposure to the pollutant concentration. The AQI values are different from country to country based on the air quality standard of the country. The higher the AQI level greater is the risk of health related problems.

There are five different categories, and each category corresponds to different health concerns.

Good AQI 0-50

In this category the air quality is Satisfactory meaning it does not possess any risk to human health.

Moderate, AQI is 51-100

In this category, the air quality is considered “Acceptable”. However, people sensitive to Ozone may experience certain respiratory problems otherwise, it only possess moderate health concern.

Unhealthy for Sensitive Groups, AQI is 101 – 150

In this category person with lung diseases, elderly people and children are at a greater risk from exposure to Ozone. Also, people suffering from heart and lung diseases possess a greater risk of the presence of particulates in the atmosphere.

Unhealthy, AQI is 151- 200

In this category, every person will start experiencing some adverse effects and members of the sensitive group will be affected even more.

Very Unhealthy, AQI is 201 – 300

In this category, everyone will start experiencing serious health effects.

Hazardous, AQI is 301 – 500

This is the highest possible level, with the entire population suffering from serious health effects.

CHAPTER 3

SYSTEM DESIGN

3.1 System Architecture

3.1.1 Flow Diagram

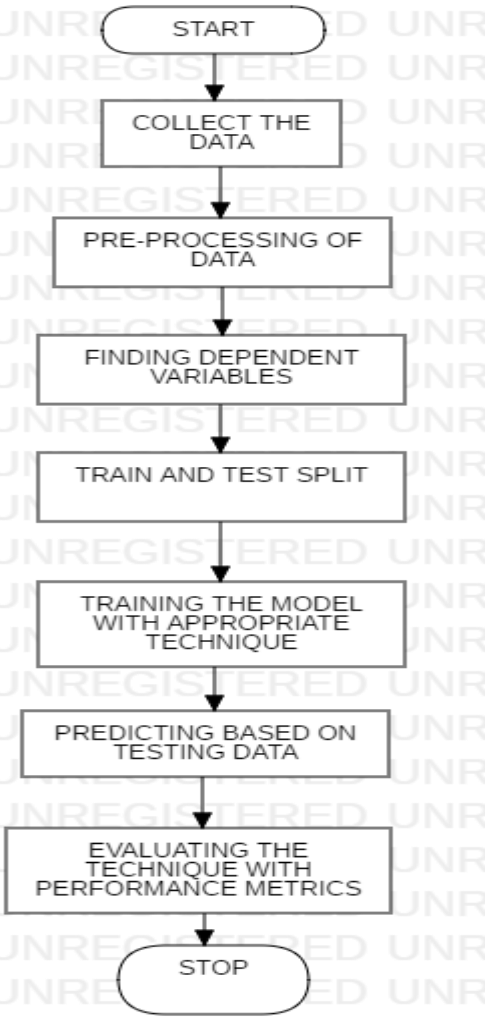


Fig 3.1.1: Flow diagram

3.1.2 Use Case Diagram:

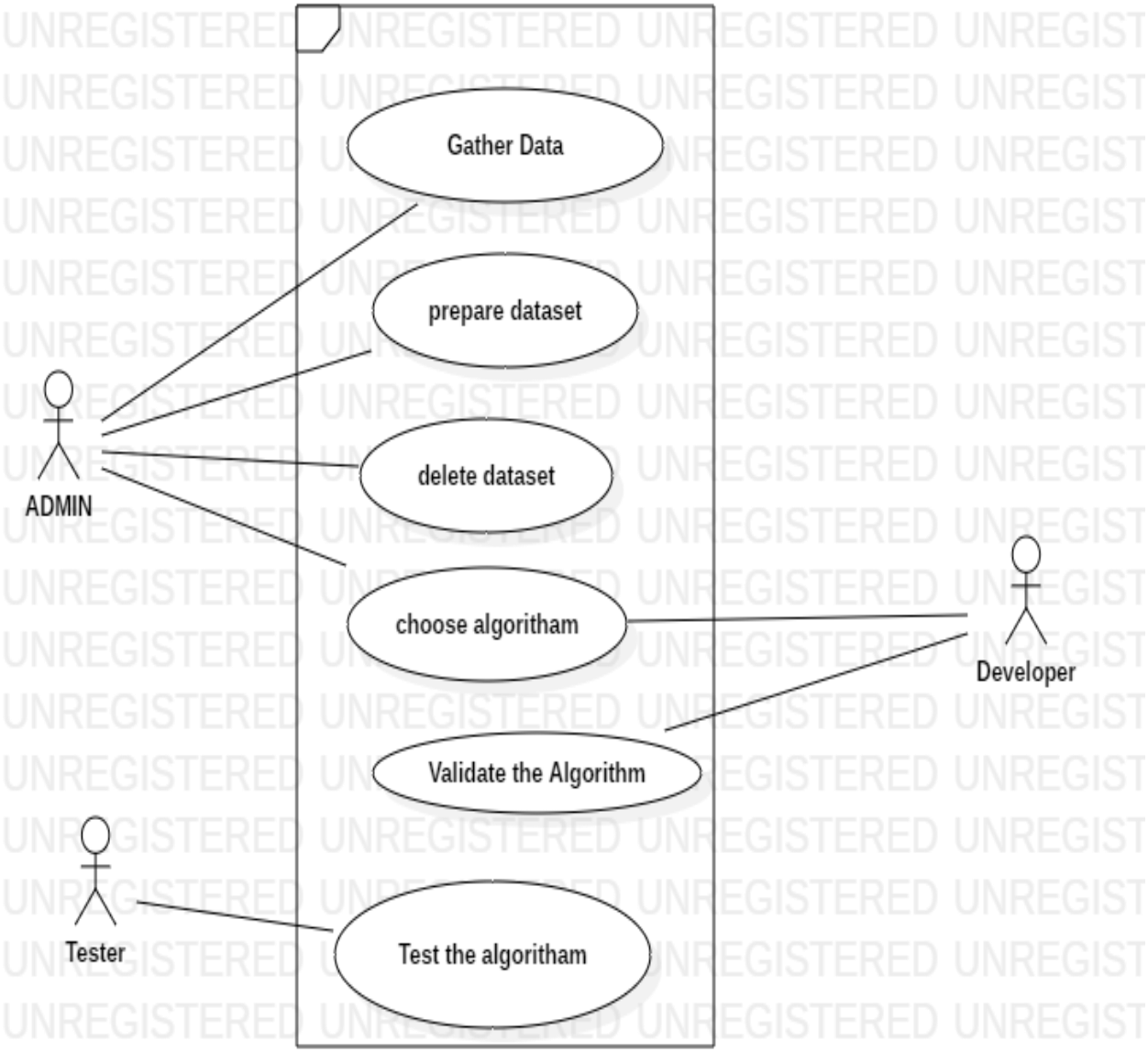


Fig: 3.1.2 Use Case Diagram

Chapter: 4

DATA OVERVIEW

4.1 Data Source:

Datasets are an integral part of the field of machine learning. Major advances in this field can result from advances in learning algorithms (such as deep learning), computer hardware, and, less-intuitively, the availability of high-quality training datasets.[1] High-quality labeled training datasets for supervised and semi-supervised machine learning algorithms are usually difficult and expensive to produce because of the large amount of time needed to label the data. Although they do not need to be labeled, high-quality datasets for unsupervised learning can also be difficult and costly to produce.

Our dataset is taken from data.gov.in which is an government Website.

4.2 Data Overview

Feature:

A feature is one column of the data in your input set. For instance, if you're trying to predict the type of pet someone will choose, your input features might include age, home region, family income, etc. The label is the final choice, such as dog, fish, iguana, rock, etc.

Types of Features

(i) Categorical Data

The values of a categorical variable are selected from a small group of categories. Categorical variables can be further categorized into ordinal and nominal variables.

- An **ordinal** variable is a categorical variable. Observations can take a value that can be logically ordered or ranked. The categories associated with ordinal variables can be ranked higher or lower than another, but do not necessarily establish a numeric difference between each category.

- A **nominal** variable is a categorical variable. Observations can take a value that is not able to be organized in a logical sequence.

(ii) Numerical Data

The values of a numerical variable are numbers. They can be further classified into discrete and continuous variables.

- A **continuous** variable is a numeric variable. Observations can take any value between a certain set of real numbers.

- A **discrete** variable is a numeric variable. Observations can take a value based on a count from a set of distinct whole values. A discrete variable cannot take the value of a fraction between one value and the next closest value.

Dataset Features

1. Stn_code: Represents code of a Place.

2. Date: On which date the readings are taken.

3. State: Telangana.

4. City: Represent cities in Telangana state.

5. Location: The place in city where the readings are taken.

6. Agency: By which board the readings are taken(Andhra Pradesh State Pollution Control Board).

7. Area: They are Industrial Areas, Sensitive Areas and Residential Areas

8. SO2: Sulphur Dioxide

9. NO2: Nitrogen dioxide

10. PM10: Particulate matter

Chapter 5

DATA PRE-PROCESSING

DESCRIPTION

Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues. Data preprocessing prepares raw data for further processing.

Data preprocessing is used database-driven applications such as customer relationship management and rule-based applications (like neural networks).

Data goes through a series of steps during preprocessing

- **Data Cleaning** Data is cleansed through processes such as filling in missing values, smoothing the noisy data, or resolving the inconsistencies in the data.
- **Data Integration** Data with different representations are put together and conflicts within the data are resolved.
- **Data Transformation** Data is normalized, aggregated and generalized.
- **Data Reduction** This step aims to present a reduced representation of the data in a data warehouse.
- **Data Discretization** Involves the reduction of a number of values of a continuous attribute by dividing the range of attribute intervals.

5.1 FINDING AIR QUALITY INDEX (AQI)

An **air quality index** (**AQI**) is a number used by government agencies to communicate to the public how polluted the air currently is or how polluted it is forecast to

become. As the AQI increases, an increasingly large percentage of the population is likely to experience increasingly severe adverse health effects. Different countries have their own air quality indices, corresponding to different national air quality standards. Some of these are the Air Quality Health Index (Canada), the Air Pollution Index (Malaysia), and the Pollutant Standards Index (Singapore), the Air Quality Index by Delhi (INDIA).

As our dataset is of India we take Air quality Index so the table would be:

AQI Category, Pollutants and Health Breakpoints

AQI Category (Range)	PM_{10} (24hr)	$PM_{2.5}$ (24hr)	NO_2 (24hr)	O_3 (8hr)	CO (8hr)	SO_2 (24hr)	NH_3 (24hr)	Pb (24hr)
Good (0–50)	0–50	0–30	0–40	0–50	0–1.0	0–40	0–200	0–0.5
Satisfactory (51–100)	51–100	31–60	41–80	51–100	1.1–2.0	41–80	201–400	0.5–1.0
Moderately polluted (101–200)	101–250	61–90	81–180	101–168	2.1–10	81–380	401–800	1.1–2.0
Poor (201–300)	251–350	91–120	181–280	169–208	10–17	381–800	801–1200	2.1–3.0
Very poor (301–400)	351–430	121–250	281–400	209–748	17–34	801–1600	1200–1800	3.1–3.5
Severe (401–500)	430+	250+	400+	748+	34+	1600+	1800+	3.5+

This is called Normalization in terms of Machine Learning.

Computing Air Quality Index (AQI)

The air quality index is a piecewise linear function of the pollutant concentration. At the boundary between AQI categories, there is a discontinuous jump of one AQI unit. To convert from concentration to AQI this equation is used.

$$I = \frac{I_{high} - I_{low}}{C_{high} - C_{low}}(C - C_{low}) + I_{low}$$

where:

I = the (Air Quality) index,
C = the pollutant concentration,
C_{low} = the concentration breakpoint that is ≤ C,
C_{high} = the concentration breakpoint that is ≥ C,
I_{low} = the index breakpoint corresponding to C_{low},
I_{high} = the index breakpoint corresponding to C_{high}.

5.2 DETECTION OF OUTLIERS

WHAT ARE OUTLIERS?

Outliers are extreme values that fall a long way outside of the other observations. For example, in a normal distribution, outliers may be values on the tails of the distribution.

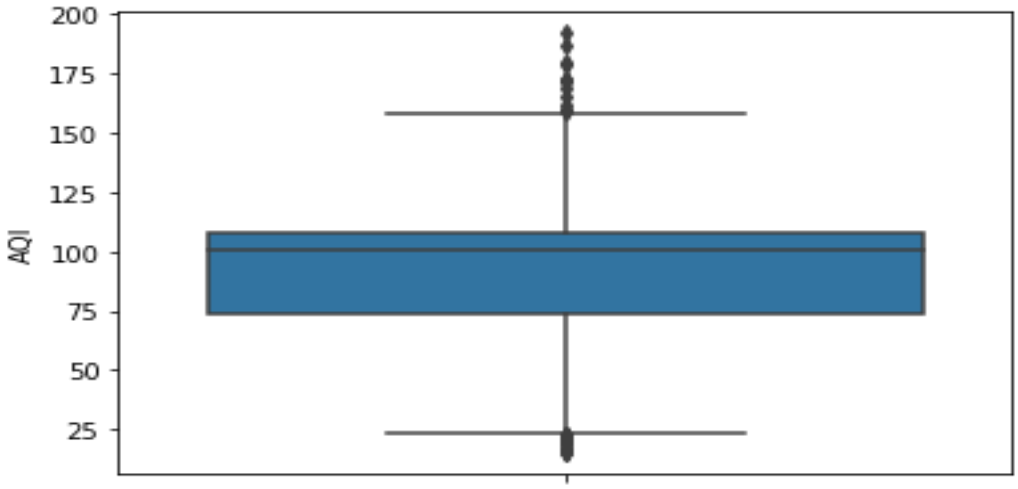
Many machine learning algorithms are sensitive to the range and distribution of attribute values in the input data. Outliers in input data can skew and mislead the training process of machine learning algorithms resulting in longer training times, less accurate models and ultimately poorer results.

We use a Statistical method call Box Plot to detect the outliers.

BOX PLOT

A simple way of representing statistical data on a plot in which a rectangle is drawn to represent the second and third quartiles, usually with a vertical line inside to indicate the median value. The lower and upper quartiles are shown as horizontal lines either side of the rectangle.

The Box plot obtained for our dataset is:



5.3 REMOVAL OF OUTLIERS

We used a technique called percentile so that the values which are not in the range of percentile we have removed them. So, at the end we removed the values less than 100 and greater than 190.

At the end we preserved 51% of data for evaluation.

5.4 DATA PREPERATION

Most machine learning algorithms require data to be formatted in a very specific way, so datasets generally require some amount of preparation before they can yield useful insights. Some datasets have values that are missing, invalid, or otherwise difficult for an algorithm to process. If data is missing, the algorithm can't use it. If data is invalid, it causes the algorithm to produce less accurate or even misleading outcomes. Good data preparation produces clean and well-curetted data that leads to more practical, accurate model outcomes.

In my model as the data is in the form of day wise data. So, we converted the data into monthly wise data i.e., by taking the daily data and making an average of the data monthly. So, the data is limited to monthly wise.

```
In [22]: df=df.set_index('Date').resample('M')["AQI"].mean()
         df.head()

Out[22]: Date
         2014-01-31    112.618168
         2014-02-28    116.115550
         2014-03-31    110.199914
         2014-04-30    112.884587
         2014-05-31    112.169417
         Freq: M, Name: AQI, dtype: float64
```

Then after that the date is converted to YYYY/MM/DD format. So, it is easily usable in Date format.

In [24]:
```
# Arranging the Date format
data=df.reset_index(level=0, inplace=False)
data = data[np.isfinite(data['AQI'])]
data=data[data.Date != '1970-01-31']
data = data.reset_index(drop=True)
data.head()
```

Out[24]:

	Date	AQI
0	2014-01-31	112.618168
1	2014-02-28	116.115550
2	2014-03-31	110.199914
3	2014-04-30	112.884587
4	2014-05-31	112.169417

CHAPTER 6

IMPLEMENTATION AND RESULTS

6.1 DESCRIPTION

The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from problem domain to the solution domain. The design of a system is perhaps the most critical factor affecting the quality of the software, and has a major impact on the later phases, particularly testing and maintenance. The output of this phase is the design document. This document is similar to a blue print or plan for the solution, and is used later during implementation, testing and maintenance.

The design activity is often divided into two separate phase-system design and detailed design. System design, which is sometimes also called top-level design, aims to identify the modules that should be in the system, the specifications of these modules, and how they interact with each other to produce the desired results. At the end of system design all the major data structures, file formats, output formats, as well as the major modules in the system and their specifications are decided.

During detailed design the internal logic of each of the modules specified in system design is decided. During this phase further details of the data structures and algorithmic design of each of the modules is specified. The logic of a module is usually specified in a high-level design description language, which is independent of the target language in which the software will eventually be implemented. In system design the focus is on identifying the modules, whereas during detailed design the focus is on designing the logic for each of the modules.

During the design phase, often two separate documents are produced. One for the system design and one for the detailed design. Together, these documents completely specify the design of the system. That is they specify the different modules in the system and internal logic of each of the modules.

A design methodology is a systematic approach to creating a design by application of set of techniques and guidelines. Most methodologies focus on system design. The two basic principles used in any design methodology are problem partitioning and abstraction. A large system cannot be handled as a whole, and so for design it is partitioned into smaller systems. Abstraction is a concept related to problem partitioning. When partitioning is used during design, the design activity focuses on one part of the system at a time. Since the part being designed interacts with other parts of the system, a clear understanding of the interaction is essential for properly designing the part. For this, abstraction is used. An abstraction of a system or a part defines the overall behavior of the system at an abstract level without giving the internal details.

While working with the part of a system, a designer needs to understand only the abstractions of the other parts with which the part being designed interacts. The use of abstraction allows the designer to practice the "divide and conquer" technique effectively by focusing one part at a time, without worrying about the details of other parts.

Like every other phase, the design phase ends with verification of the design. If the design is not specified in some executable language, the verification has to be done by evaluating the design documents. One way of doing this is thorough reviews. Typically, at least two design reviews are held-one for the system design and one for the detailed and one for the detailed design.

6.2 TRAIN TEST SPLIT

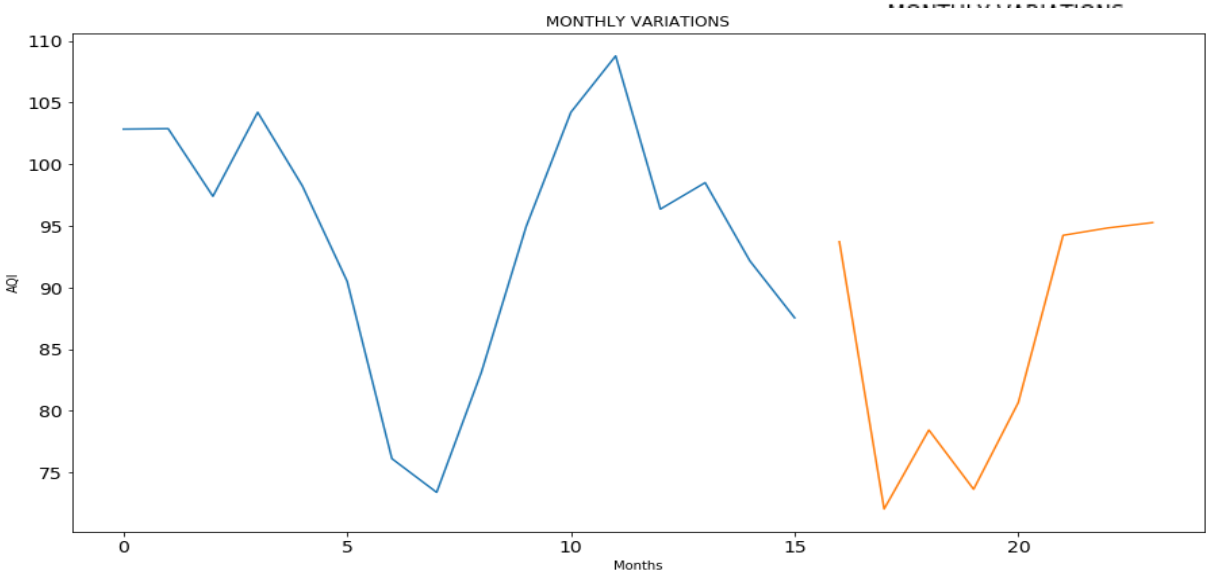
As I said before, the data we use is usually split into training data and test data. The training set contains a known output and the model learns on this data in order to be generalized to other data later on. We have the test dataset (or subset) in order to test our model’s prediction on this subset.

As it is Time series data we cannot split the data randomly we need to split according to time. Train data contain 70% of Data and Test contain 30% of data.

```
In [26]: n = df.shape[0]
         train_size = 0.70

         features_dataframe = data.sort_values('Date')
         train = data.iloc[:int(n * train_size)]
         test = data.iloc[int(n * train_size):]
```

```
In [27]: train.AQI.plot(figsize=(15,8), title= 'MONTHLY VARIATIONS', fontsize=14)
         test.AQI.plot(figsize=(15,8), title= 'MONTHLY VARIATIONS', fontsize=14)
         plt.xlabel("Months")
         plt.ylabel("AQI")
         plt.show()
```



The Blue line is training data.
The Orange line is test data.

MACHINE LEARNING MODELS

For classification and regression problem, there are different choices of Machine Learning Models each of which can be viewed as a black box that solve the same problem. However, each model come from a different algorithm approaches and will perform differently under different data set. The best way is to use cross-validation to determine which model performs best on test data.

Here I'll try to provide a high level summary of its underlying algorithmic approach and hopefully can give a sense of whether it will be a good fit for your particular problem.

We use different models for prediction as we need to measure the models, we need a baseline model.

6.3 BASELINE MODEL

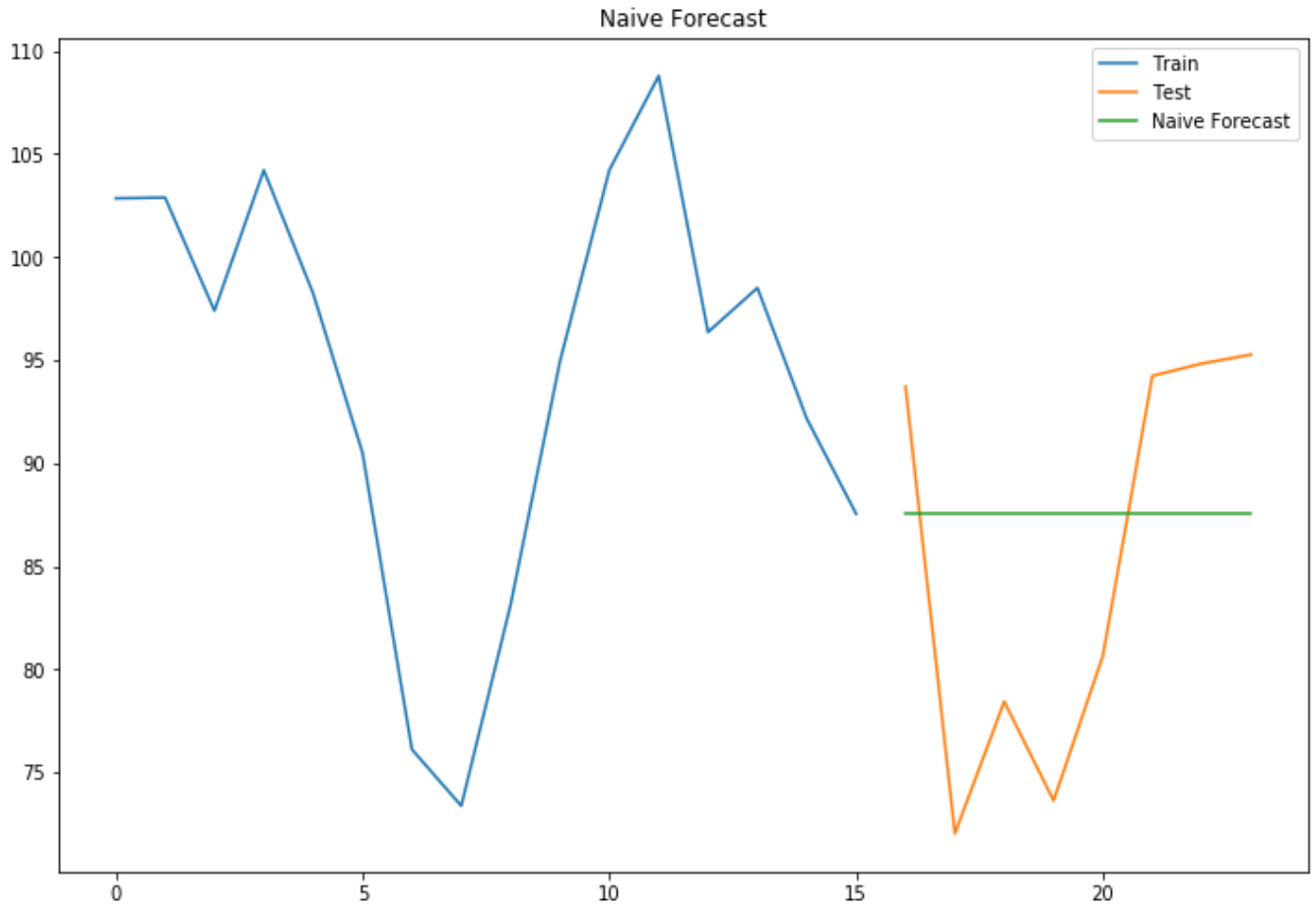
A baseline is a method that uses heuristics, simple summary statistics, randomness, or machine learning to create predictions for a dataset. You can use these predictions to measure the baseline's performance (e.g., accuracy)-- this metric will then become what you compare any other machine learning algorithm against.

For Base line model we used Naïve Approach.

Naïve approach

Estimating technique in which the last period's actuals are used as this period's forecast, without adjusting them or attempting to establish causal factors. It is used only for comparison with the forecasts generated by the better (sophisticated) techniques.

In [28]:
```
dd= np.asarray(train.AQI)
y_hat = test.copy()
y_hat['naive'] = dd[len(dd)-1]
plt.figure(figsize=(12,8))
plt.plot(train.index, train['AQI'], label='Train')
plt.plot(test.index,test['AQI'], label='Test')
plt.plot(y_hat.index,y_hat['naive'], label='Naive Forecast')
plt.legend(loc='best')
plt.title("Naive Forecast")
plt.show()
```



The Blue line is train data.
The Orange line is test data.
The Green line is predicted data.

6.4 STATISTICAL MODELS

A statistical model is a mathematical model that embodies a set of statistical assumptions concerning the generation of some sample data and similar data from a larger population. A statistical model represents, often in considerably idealized form, the data-generating process.

The assumptions embodied by a statistical model describe a set of probability distributions, some of which are assumed to adequately approximate the distribution from which a particular data set is sampled. The probability distributions inherent in statistical

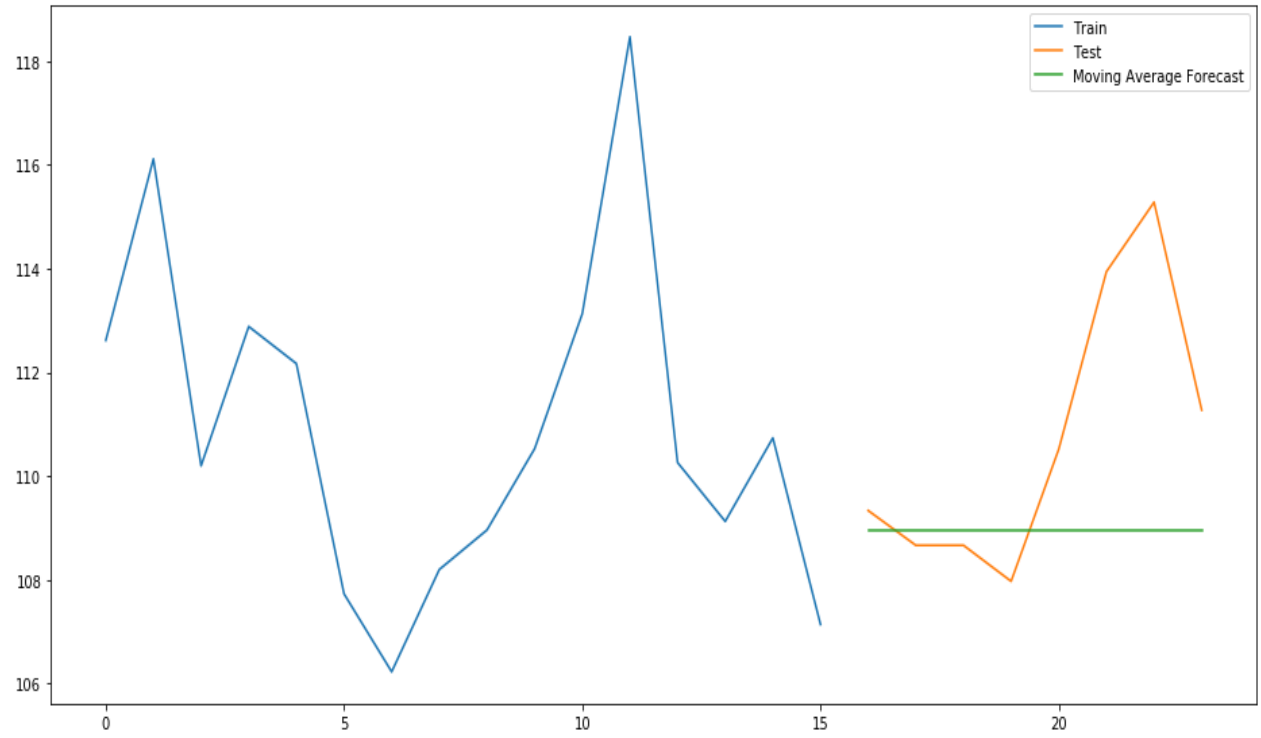
models are what distinguish statistical models from other, non-statistical, mathematical models.

SIMPLE MOVING AVERAGE

A simple moving average (SMA)is an arithmetic moving average calculated by adding recent closing prices and then dividing that by the number of time periods in the calculation average. A simple, or arithmetic, moving average that is calculated by adding the closing price of the security for a number of time periods and then dividing this total by that same number of periods. Short-term averages respond quickly to changes in the price of the underlying, while long-term averages are slow to react.

$$SMA = \frac{A_1 + A_2 + \ldots + A_n}{n}$$

In [41]:
```
y_hat_avg = test.copy()
y_hat_avg['moving_avg_forecast'] = train['AQI'].rolling(2).mean().iloc[-1]
plt.figure(figsize=(16,8))
plt.plot(train['AQI'], label='Train')
plt.plot(test['AQI'], label='Test')
plt.plot(y_hat_avg['moving_avg_forecast'], label='Moving Average Forecast')
plt.legend(loc='best')
plt.show()
```



The Blue line is train data.
The Orange line is test data.
The Green line is predicted data.

6.5 DEEP LEARNING MODELS

Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms. Learning can be supervised, semi-supervised or unsupervised.

Deep learning architectures such as deep neural networks, deep belief networks and recurrent neural networks have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases superior to human experts.

LONG SHORT TERM DEPENDENCIES

Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. They were introduced

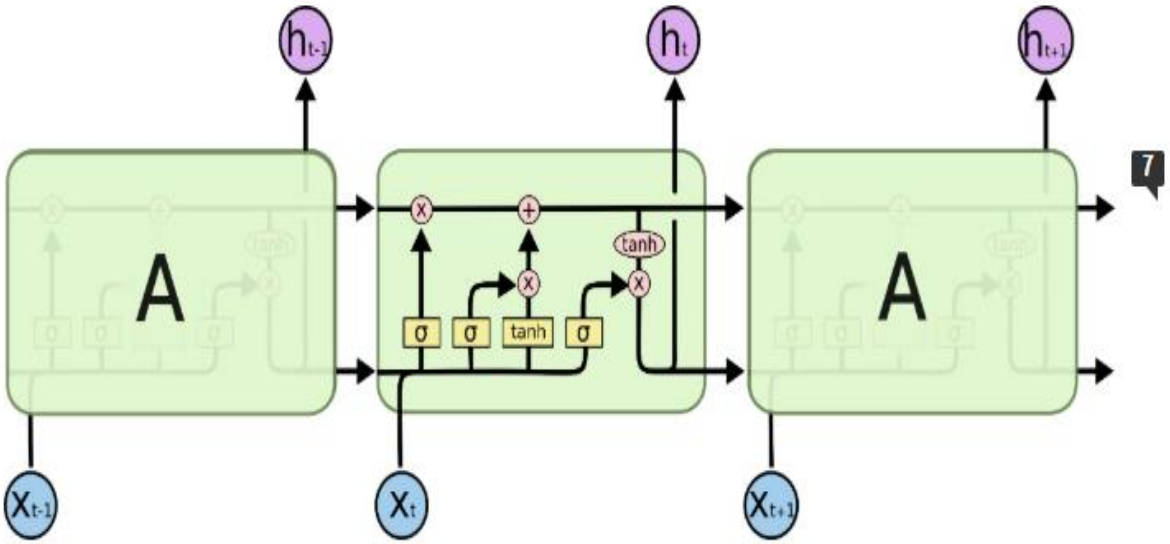
by Hochreiter & Schmidhuber (1997), and were refined and popularized by many people in following work.[1] They work tremendously well on a large variety of problems, and are now widely used.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

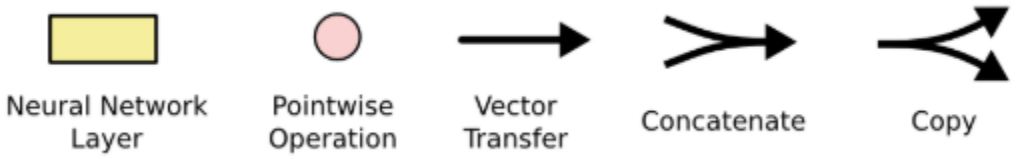
All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.

LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.

The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to just flow along it unchanged.

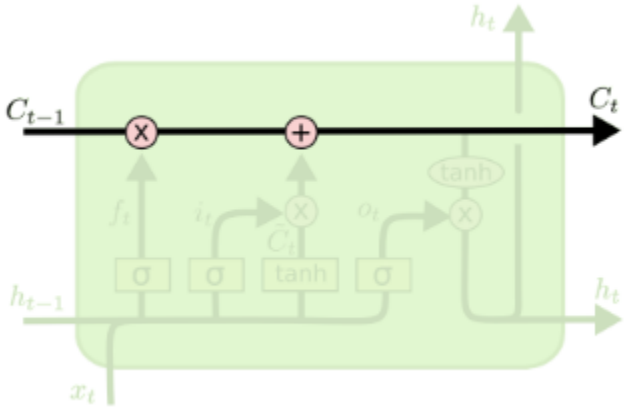


Don’t worry about the details of what’s going on. We’ll walk through the LSTM diagram step by step later. For now, let’s just try to get comfortable with the notation we’ll be using.



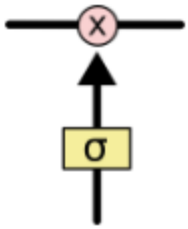
In the above diagram, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent point wise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denote its content being copied and the copies going to different locations.

The key to LSTMs is the cell state, the horizontal line running through the top of the diagram.



The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates.

Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a point wise multiplication operation.



The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means “let nothing through,” while a value of one means “let everything through!”

An LSTM has three of these gates, to protect and control the cell state.

In [75]:
```
df=data[['AQI']]
```

In [76]:
```
dataframe = df
dataset = dataframe.values
dataset = dataset.astype('float32')
```

In [77]:
```
scaler = MinMaxScaler(feature_range=(0, 1))
dataset = scaler.fit_transform(dataset)
```

In [78]:
```
# split into train and test sets
train_size = int(len(dataset) * 0.70)
test_size = len(dataset) - train_size
train, test = dataset[0:train_size,:], dataset[train_size:len(dataset),:]
print(len(train), len(test))
```

```
16 8
```

In [79]:
```
# convert an array of values into a dataset matrix
def create_dataset(dataset, look_back=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-look_back-1):
        a = dataset[i:(i+look_back), 0]
        dataX.append(a)
        dataY.append(dataset[i + look_back, 0])
    return numpy.array(dataX), numpy.array(dataY)
```

In [79]:
```
# convert an array of values into a dataset matrix
def create_dataset(dataset, look_back=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-look_back-1):
        a = dataset[i:(i+look_back), 0]
        dataX.append(a)
        dataY.append(dataset[i + look_back, 0])
    return numpy.array(dataX), numpy.array(dataY)
```

In [80]:
```
# reshape into X=t and Y=t+1
look_back = 1
trainX, trainY = create_dataset(train, look_back)
testX, testY = create_dataset(test, look_back)
```

In [81]:
```
# reshape input to be [samples, time steps, features]
trainX = numpy.reshape(trainX, (trainX.shape[0], 1, trainX.shape[1]))
testX = numpy.reshape(testX, (testX.shape[0], 1, testX.shape[1]))
```

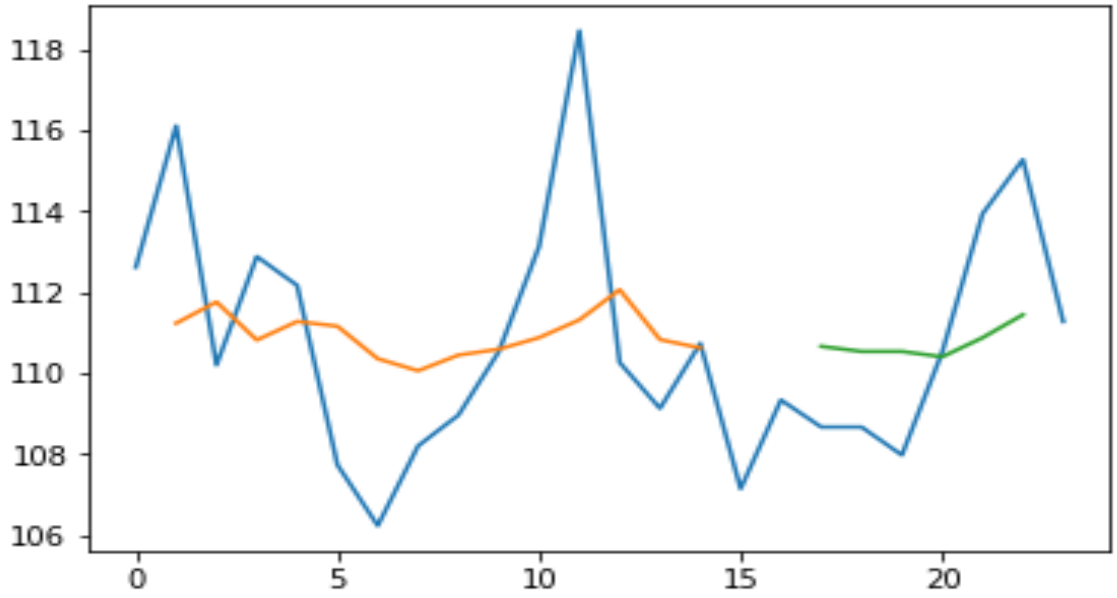
In [82]:
```
# create and fit the LSTM network
model = Sequential()
model.add(LSTM(4, input_shape=(1, look_back)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
history=model.fit(trainX, trainY, epochs=30, batch_size=1, verbose=2,validation_data=(testX, testY))
```

Test Loss by the LSTM:

Train on 14 samples, validate on 6 samples
Epoch 1/30
 - 4s - loss: 0.2047 - val_loss: 0.1707
Epoch 2/30
 - 0s - loss: 0.1900 - val_loss: 0.1578
Epoch 3/30
 - 0s - loss: 0.1771 - val_loss: 0.1455
Epoch 4/30
 - 0s - loss: 0.1634 - val_loss: 0.1349
Epoch 5/30
 - 0s - loss: 0.1522 - val_loss: 0.1248
Epoch 6/30
 - 0s - loss: 0.1411 - val_loss: 0.1156

```
Epoch 7/30
 - 0s - loss: 0.1319 - val_loss: 0.1061
Epoch 8/30
 - 0s - loss: 0.1222 - val_loss: 0.0982
Epoch 9/30
 - 0s - loss: 0.1147 - val_loss: 0.0901
Epoch 10/30
 - 0s - loss: 0.1066 - val_loss: 0.0836
Epoch 11/30
 - 0s - loss: 0.0999 - val_loss: 0.0780
Epoch 12/30
 - 0s - loss: 0.0943 - val_loss: 0.0724
Epoch 13/30
 - 0s - loss: 0.0888 - val_loss: 0.0678
Epoch 14/30
 - 0s - loss: 0.0845 - val_loss: 0.0633
Epoch 15/30
 - 0s - loss: 0.0803 - val_loss: 0.0594
Epoch 16/30
 - 0s - loss: 0.0778 - val_loss: 0.0556
Epoch 17/30
 - 0s - loss: 0.0737 - val_loss: 0.0532
Epoch 18/30
 - 0s - loss: 0.0716 - val_loss: 0.0508
Epoch 19/30
 - 0s - loss: 0.0693 - val_loss: 0.0489
Epoch 20/30
 - 0s - loss: 0.0678 - val_loss: 0.0471
Epoch 21/30
 - 0s - loss: 0.0664 - val_loss: 0.0456
Epoch 22/30
 - 0s - loss: 0.0650 - val_loss: 0.0447
Epoch 23/30
 - 0s - loss: 0.0646 - val_loss: 0.0434
Epoch 24/30
 - 0s - loss: 0.0635 - val_loss: 0.0432
Epoch 25/30
 - 0s - loss: 0.0630 - val_loss: 0.0422
Epoch 26/30
 - 0s - loss: 0.0625 - val_loss: 0.0415
Epoch 27/30
 - 0s - loss: 0.0619 - val_loss: 0.0410
Epoch 28/30
```

- 0s - loss: 0.0616 - val_loss: 0.0406
Epoch 29/30
- 0s - loss: 0.0613 - val_loss: 0.0403
Epoch 30/30
- 0s - loss: 0.0613 - val_loss: 0.0400



The Blue line is train data.
The Orange line is test data.
The Green line is predicted data.

CHAPTER 7
TESTING

DESCRIPTION

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.1 PERFORMANCE METRIC

The next important question while evaluating the performance of a machine learning model is *what dataset should be used to evaluate model performance*. The machine learning model cannot be simply tested using the training set, because the output will be prejudiced, because the process of training the machine learning model has already tuned the predicted outcome to the training dataset. Therefore in order to estimate the generalization error, the model is required to test a dataset which it hasn't seen yet; giving birth to the term testing dataset.

Therefore for the purpose of testing the model, we would require a labeled dataset. This can be achieved by splitting the training dataset into training dataset and testing dataset. This can be achieved by various techniques such as, k-fold cross validation, jackknife resampling and bootstrapping. Techniques like A/B testing are used to measure performance of machine learning models in production against response from real user interaction.

In a regression task, the model learns to predict numeric scores.

An example is predicting the price of a stock on future days given past price history and other information about the company and the market.

This section will deal with two ways of measuring regression performance:

MEAN ABSOLUTE ERROR (MAE)

In statistics, mean absolute error (MAE) is a measure of difference between two continuous variables. Assume *X* and *Y* are variables of paired observations that express the same phenomenon. Examples of *Y* versus *X* include comparisons of predicted versus observed, subsequent time versus initial time, and one technique of measurement versus an alternative technique of measurement. Consider a scatter plot of *n* points, where point *i* has coordinates (x_i, y_i)... Mean Absolute Error (MAE) is the average vertical distance between each point and the identity line. MAE is also the average horizontal distance between each point and the identity line.

The Mean Absolute Error is given by:

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

7.2 TEST OUTPUTS

7.1.1 BASELINE MODEL

TEST OUTPUT

Date	Actual	Predicted
2015-07-31 00:00:00	108.66666666666666	107.14166666666668
2015-08-31 00:00:00	107.97348484848486	107.14166666666668
2015-09-30 00:00:00	110.51999999999998	107.14166666666668
2015-10-31 00:00:00	113.94157088122603	107.14166666666668
2015-11-30 00:00:00	115.28183520599252	107.14166666666668
2015-12-31 00:00:00	111.27491408934704	107.14166666666668

TEST ERROR

```
In [29]: from sklearn.metrics import mean_absolute_error
         mae=mean_absolute_error(test.AQI,  y_hat.naive)
         print("Mean_absolute_error:",mae)

         Mean_absolute_error: 3.5660317028976483
```

7.1.2 SIMPLE MOVING AVERAGE

TEST OUTPUT

Date	Actual	Predicted
2015-07-31 00:00:00	108.66666666666666	108.666666
2015-08-31 00:00:00	107.97348484848486	108.315339
2015-09-30 00:00:00	110.51999999999998	109.244508
2015-10-31 00:00:00	113.94157088122603	112.229469
2015-11-30 00:00:00	115.28183520599252	114.619962
2015-12-31 00:00:00	111.27491408934704	113.275545

TEST ERROR

```
In [42]: mae=mean_absolute_error(test.AQI, y_hat_avg.moving_avg_forecast)
         print("Mean_absolute_percentage_error:",mae)

         Mean_absolute_percentage_error: 2.1461802276185526
```

7.1.3 LONG SHORT TERM MEMORY

TEST OUTPUT

Date	Actual	Predicted
2015-07-31 00:00:00	108.66666666666666	[110.66127]
2015-08-31 00:00:00	107.97348484848486	[110.535805]
2015-09-30 00:00:00	110.51999999999998	[110.535805]
2015-10-31 00:00:00	113.94157088122603	[110.403244]
2015-11-30 00:00:00	115.28183520599252	[110.876236]
2015-12-31 00:00:00	111.27491408934704	[111.44397]

TEST ERROR

In [53]:
```
# make predictions
trainPredict = model.predict(trainX)
testPredict = model.predict(testX)
# invert predictions
trainPredict = scaler.inverse_transform(trainPredict)
trainY = scaler.inverse_transform([trainY])
testPredict = scaler.inverse_transform(testPredict)
testY = scaler.inverse_transform([testY])
# calculate mean absolute error
trainScore1 = math.sqrt(mean_absolute_error(trainY[0], trainPredict[:,0]))
print('Train Score: %.2f MAE'%(trainScore1))
testScore = math.sqrt(mean_absolute_error(testY[0], testPredict[:,0]))
print('Test Score: %.2f MAE' % (testScore))
```

```
Train Score: 1.54 MAE
Test Score: 1.50 MAE
```

CHAPTER 8

CONCLUSION

Finally we conclude that Prediction of Air Pollution is an Area were the Future pollution is predicted based on the past data. Here in this application the user is uses the predicted values so that future mitigation strategies can be implemented to reduce the pollution. This application can be extended to a web page were the user can interact with the data.

The future enhancement of this project is to take the server by which we can utilize more places so that user can use the data to predict in different locations more efficiently even from the small places.

REFERENCES

1. https://data.gov.in/
2. https://scikit-learn.org/stable/
3. https://en.wikipedia.org/wiki/Air_quality_index
4. https://machinelearningmastery.com/time-series-forecasting-methods-in-python-cheat-sheet/
5. http://colah.github.io/posts/2015-08-Understanding-LSTMs/
6. https://keras.io/