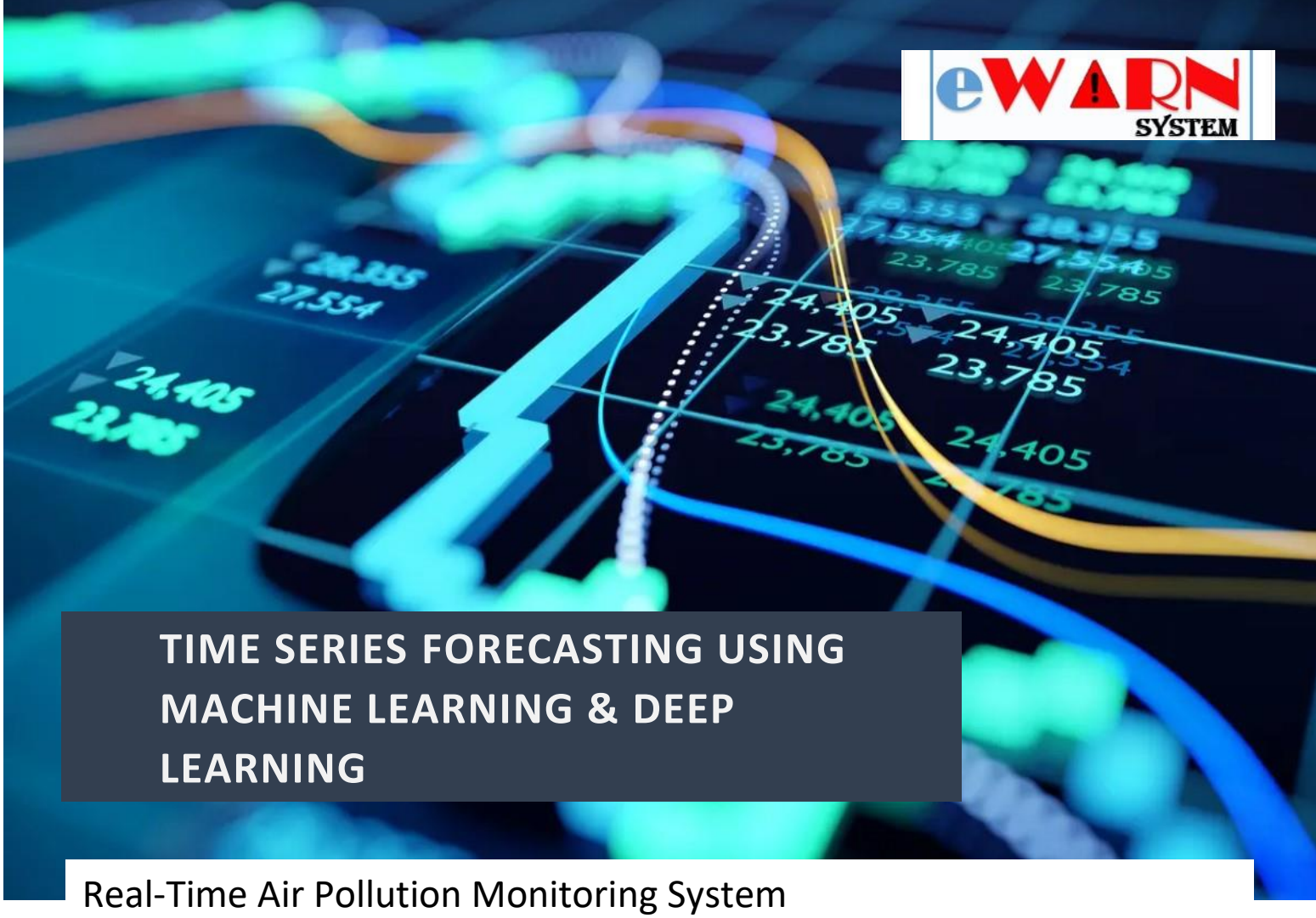


Real-Time Air Pollution Monitoring System

The objective of this project which constantly receives air quality data from the sensing nodes distributed over a vast area. The objective of the project is to develop a real time air pollution monitoring and alerting system. The proposed work consists of a full stack IOT architecture consisting of environmental sensors connected to a Raspberry Pi, gathering data from the physical environment and redirecting it to the cloud ware using HTTP. We use a cloud database which constantly receives air quality data from the sensing nodes distributed over a vast area. Each individual sensor measures multiple parameters such as, CO_2, CO, O_2, O_3, CH_4, wind speed, temperature, humidity and dew-point. Based upon the concentration values, an Air Quality Index (AQI) is calculated following a dynamic time-weighted mean algorithm.

For the described task above, we are going to use some machine learning and deep learning models and for the testing purposes, we may use some error metrics to compare the accuracy of each model that we are going to fit our dataset into.

K A JANUKA SHEHAN FERNANDO

NIT ROURKELA(119CS0120)

Contents

Time Series Forecasting

Introduction

Air pollution consists of chemicals or particles in the air that can harm the health of humans, animals, and plants. It also damages buildings. Pollutants in the air take many forms. They can be gases, solid particles, or liquid droplets. It can affect countries all over the world causing serious health issues and even death. In this project we majorly focus on developing an efficient and an accurate model which can forecast upcoming air quality. Air quality in other words measured with the AQI (Air Quality Index) and for inputs we may use the concentrations of some pollutants such as

- Temperature
- Humidity
- Dew point
- O_2
- CO_2
- CH_4
- SO_2
- NH_4 etc.

We have used time series forecasting approach for predicting the AQI value through predicted future levels of various pollutants and the within a considerable confidence interval.

We may use some statistical machine learning and deep learning models in order to analyze, train and to do predictions from the cleaned and prepared time-stamped dataset.

To ensure a sustainable pollution level, air quality forecasting is a significant step to getting early information about air quality. For instance, this early information can help to improve the traffic control system, which ultimately enhances the air quality. The public can modify their travel plan accordingly to keep themselves safe. Ambulance and patients can change their paths based on the early air pollution information. Air pollution forecasting has a vital role in air quality control management and in developing a sustainable environment.

Acknowledgement

I would like to express my sincere gratitude to ***Prof. Santos Kumar Das***, who allowed me to gain this much of experience by making me a part of the **EWARN** family in the period of June 2022 – August 2022 as an intern. Special thanks to ***Dr. Harshit Srivastava*** for being my mentor throughout this time period and for having patience providing the flexibility in a great extent until the completion of the project.

I am grateful to all my friends who gave me great support over the period of this time. Exceptional thanks go to all the YouTube tutors, Coursera course tutors for all the knowledge I gained throughout this period.

Internship Timeline

1. Project Prerequisites

- During the first the month of June I followed two courses on Coursera (online) in order to gain knowledge on Python, Machine Learning, Deep Learning and Data Analysis.

Machine Learning with Python offered by *IBM*

Introduction to Deep Learning and Neural Networks with Keras by *IBM*

- I followed three guided projects on Coursera about simple machine learning modelling, time series analysis techniques and *fbprophet*.

Breast Cancer Prediction guided project

Predict Future Product Prices using Facebook Prophet guided project

Time Series Data Visualization and Analysis Techniques

2. Research Papers

- During the first three weeks of the month of June 2022, I followed three research papers with the guidance of ***Dr. Harshit Srivastava.***

1. Paper on multivariate time series forecasting with CNN and LSTM
2. Paper on Time series forecasting with SARIMA and Prophet.
3. Paper on Multivariate Time Series Forecasting with Dilated Residual - Convolutional Neural Networks for Urban Air Quality Prediction

3. Project Completion

- Project was completed at the end of the month, July 2022.

Tools and Techniques

- I used Python as the language of my project.
- I used *Jupyter* notebook and google *collab* for the complete project.
- You can download the notebook through the following link.
  See my project

References

- www.youtube.com
- https://www.tableau.com/learn/articles/time-series-forecasting
- https://towardsdatascience.com/an-overview-of-time-series-forecasting-models-a2fa7a358fcb
- https://machinelearningmastery.com/time-series-forecasting-methods-in-python-cheat-sheet/
- https://otexts.com/fpp2/holt-winters.html
- https://ieeexplore.ieee.org/abstract/document/9590113
- https://towardsdatascience.com/4-techniques-to-handle-missing-values-in-time-series-data-c3568589b5a8
- www.analyticsvidhya.com

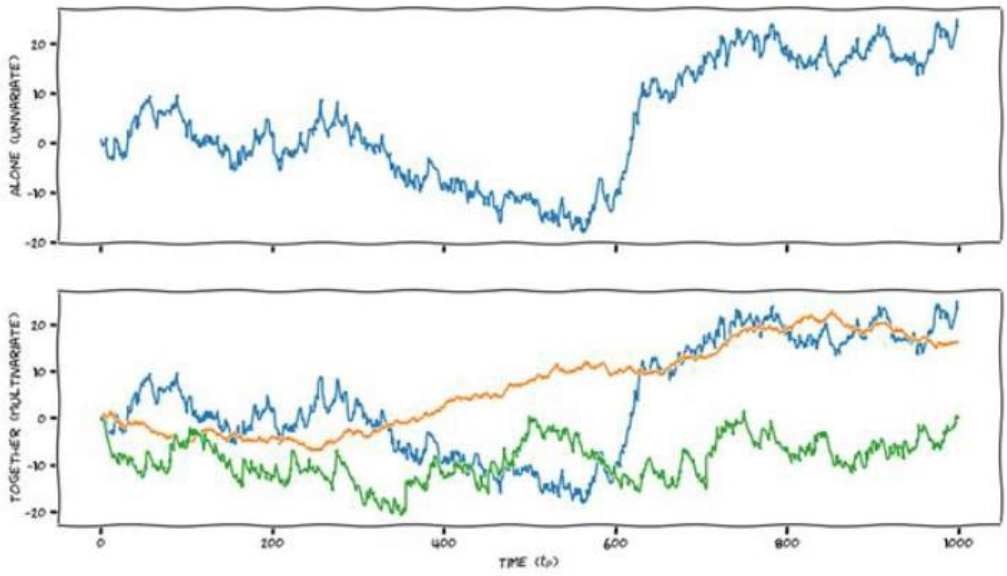
01.Time-Series Analysis

Time series as its name itself indicates is a series of data points that are collected over time. This series of points are collected over a period of definite time, either every minute, every day, every month or every year. Unlike conventional Machine Learning models, in models intended to predict time series, the time variable is vital in this series. This time the order, the observations provides a very useful source of information to be analyzed and used in the prediction process.

And as for the representation of these time series, line charts are normally used where we put the objective variable to predict y on the y-axis on the x-axis, the time variable. The analysis of time series is what allows us to extract information and knowledge in order to later be able to train our models. Time series analysis involves interpreting the data we have and calculating to know the changes that occur over time.

Types of Time-Series Data

1. **Univariate series** - we have a single variable that varies with time. For example, we might want to predict the rainfall along of the different days.
2. **Multi-variant series** - Has more than one variable which are time dependent. Each variable depends not only on its past values, but which also has a certain dependence on the other variables This of Venice. Among the variables, it is used to forecast or predict future values. Using the example above, it may be that we want to predict the humidity that is going to be seen long different days, but we could have more variables, such as the temperature, the rain etc.



Univariate

Multi-variant

Components of Time-Series Data

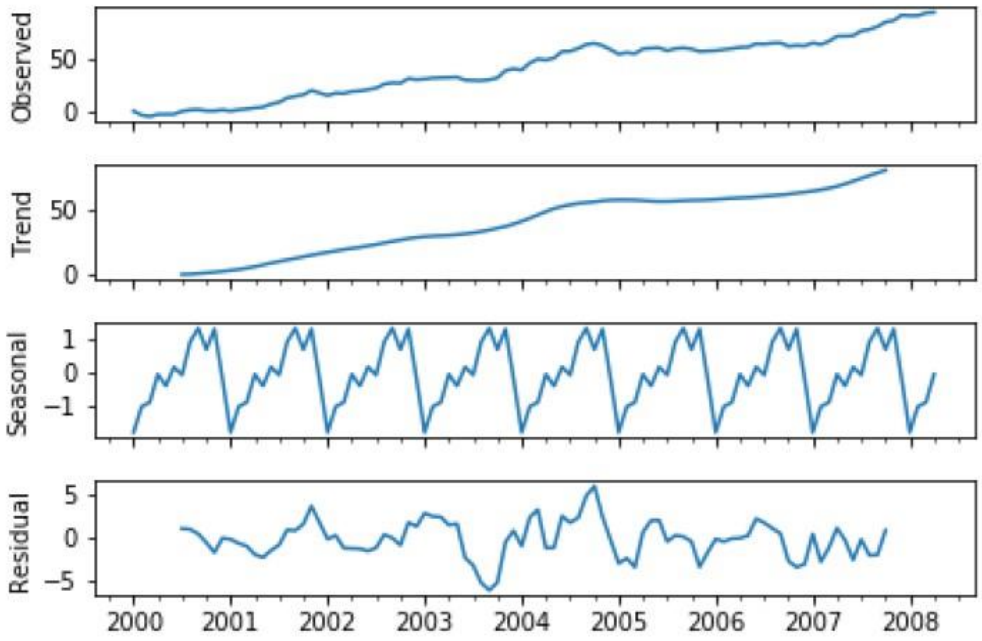
There is a step that is very common in all these analyses, it is the distribution of the time series in its different components. We have three components within a time series,

- Trend
- Seasonality
- Noise

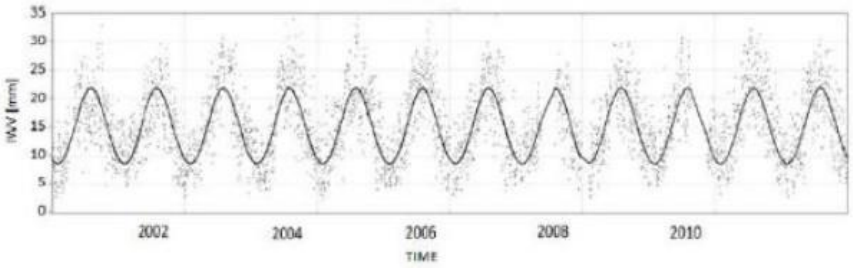
1. **Trend**
   The trend indicates if there is a constant upward or downward movement over a long period of time and as the one that we observe below graph.

2. **Seasonality**
   After we have the component of seasonality that is, if the increase or the decrease in data values is periodic. We would call these patterns that we are seeing as seasonality. A pattern can be identified that repeats every time frame.

3. **Noise(residual)**
   Noise is nothing but the random increase or decrease of the values. The more noise we have in our data set, the worse the model will work.



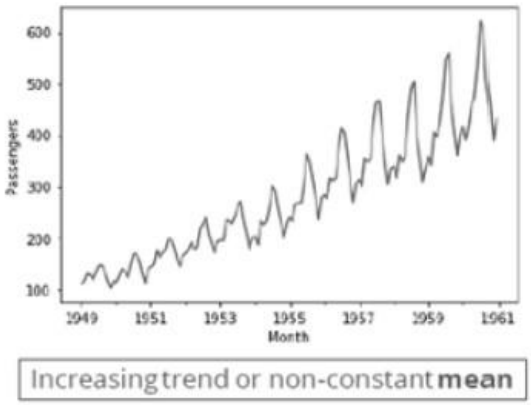
We will talk about the different types of time series derived from the different components that we had described. We will start with the stationary data, which are those that we see in this diagram before applying any statistical model on a time series.



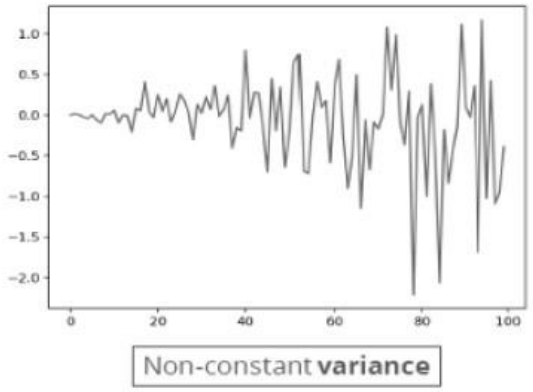
This series should be stationary or at least variant less, over a long time frame. This means that at different time periods you must have a mean, a variance and co-variance. If the mean, the variance, and the covariance of our data is not constant over time, then our data is not stationary and we must make it stationary before any model can be applied predictive. This is necessary because if our data has some regular pattern, then there is a high probability that in a different interval or in the future they have the same behavior and the values can be predicted.

Also mathematical calculation for stationary data is easier compared to the non-stationary data. Here we would have the example of stationary data, where we see that has a constant mean where none is observed trend nor the growing mean. We also see that it has a repeating pattern constantly and that this does not vary over time. The same with the with variance.

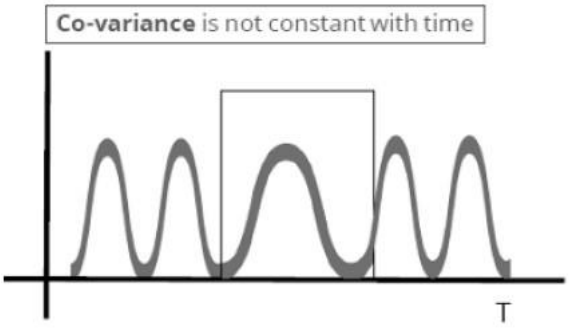
Unlike the stationary data, we can have multiple non-stationary data types. Here we have an example where we have an increasing trend and, therefore, the mean would not be constant.



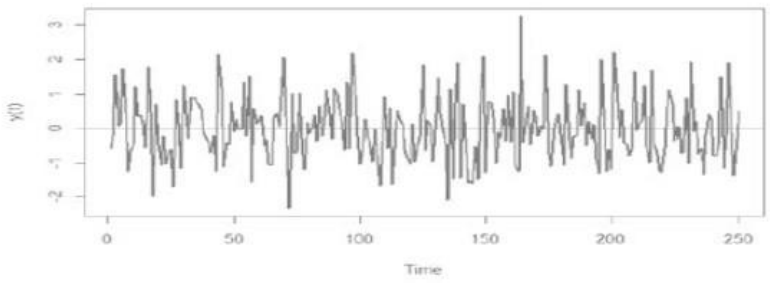
We can also have some non-stationary data because we do not have a constant variance and we can see how initially we have a small variance and then it increases the variance until it becomes wider.



We can also have non-stationary data because we don't have one with constant variance which we can see as at the beginning there is one with variance in a certain interval and then it widens and changes pattern and then returns to the same. This would be an example where covariance is not constant.

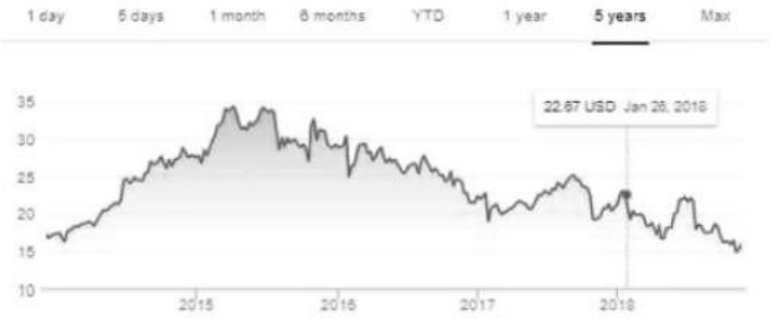


Finally, we have a data set where we cannot get information, which is what we had discussed before as noise. If we look at this graph, we cannot see any kind of pattern or behavior to be repeated over time.

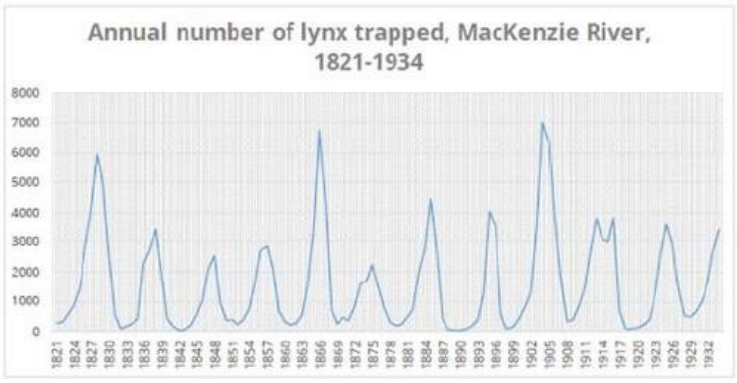


Here are other examples of non-stationary data. We can appreciate in these graphs the component of the trend where we have a rising trend and another chart has a declining trend then a graph with a marked seasonality.

Uptrend

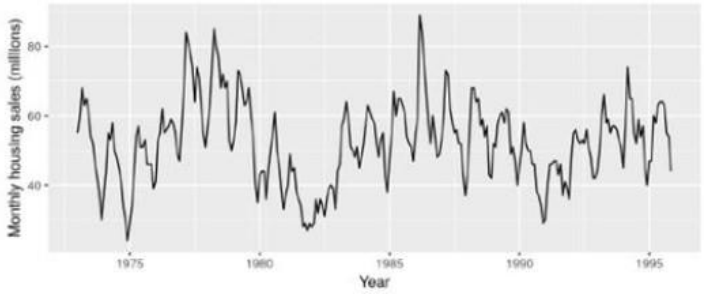


Downtrend



We can see that there is a wide peak followed by three small spikes. Again a wide peak and three other small spikes and the same one, a pattern that repeats itself all the time.

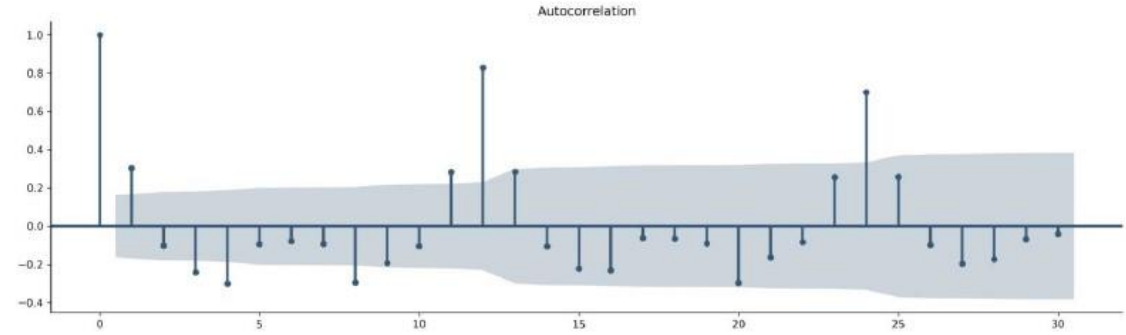
And finally we also have a graph with a cyclical component. This cyclical component would be the fluctuation around the trend line at a random interval.

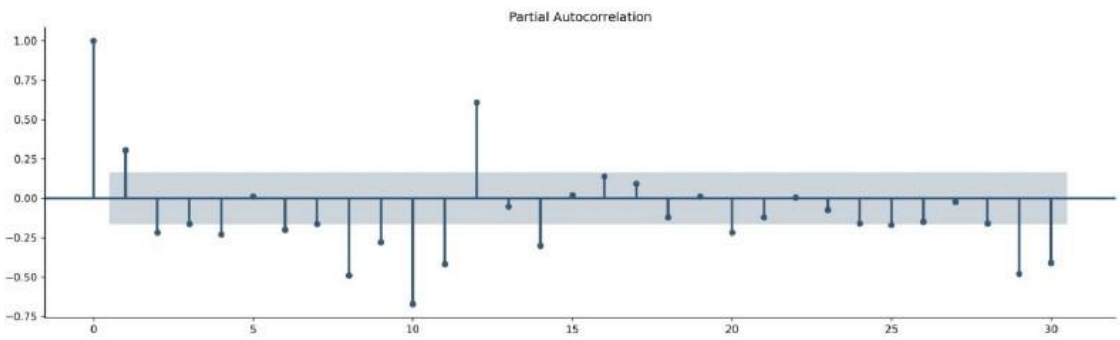


Unlike seasonality in the cyclical component, the pattern in which it occurs over the interval is random. Another component that we must know if we want to train models capable of predicting time series is the concept of self-correlation. Self-correlation can be seen as the measure of internal correlation in a time series.

The autocorrelation represents the degree of similarity between a given time series and a delayed version of itself at successive time intervals. To know the autocorrelation, there are two types of graphs that are used very often, which would be the

- Autocorrelation
- Partial Autocorrelation





These graphics allow us to identify which statistical model is the one that best fits our data series and also how to configure our hyper parameters. In order to be able to interpret these graphs and be able to associate the interpretation of these graphs with the best model for a given time series.

There is a method called Box Jenkins method. Through this method, we will see how it allows us to interpret these graphs and, based on to the results, knowing which model we should use in that time series.

02.Time-Series Forecasting

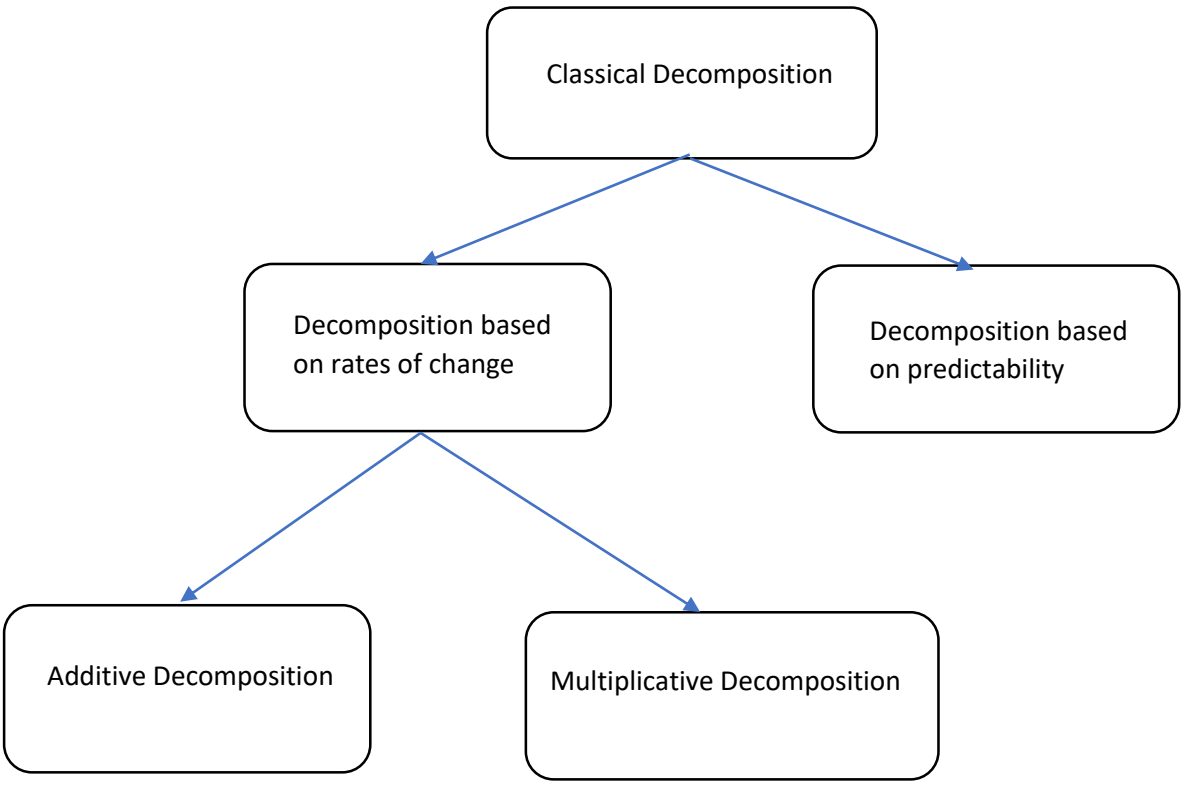
Time series forecasting is nothing but predicting future events by analyzing past trends, based on the assumption that future trends will hold similar to historical trends. Forecasting involves using models fit on historical data to predict future values. Forecasting involves using models fit on historical data to predict future values. Prediction problems that involve a time component require time series forecasting, which provides a data-driven approach to effective and efficient planning.

Applications of Time-Series forecasting

- Power generation plants may use to decide future power demands.
- Predicting weather and climate
- Companies use to schedule staff of coming week.
- Estimate the inventory requirements to stock inventory to meet demand.
- Supply and demand forecasting to optimize fleet management and other aspects of the supply chain.
- Predicting infection rates to optimize disease control and outbreak programs.
- Predicting customer ratings through to forecasting product sales.

Time Series Decomposition

Time series data can exhibit variety of patterns, so it is often helpful to split time series into components, each representing underlying pattern category. When we decompose a time series into components, we think of a time series as comprising three components: a trend component, a seasonal component, and residual component or noise. Classical decomposition is one of the most popular types of time series decomposition.



In additive time series, the three components (trend, seasonality and residuals) add together to make the time series. An additive model is used when the variations around the trend do not vary with level of the time series. In a multiplicative time-series, the three components multiply together to make the time series. A multiplicative model is appropriate if the trend is proportional to the level of the time series.

Basic Concepts:

Trend – The increasing or decreasing linear behavior of the series over time. A trend can be increasing (up), decreasing (down), or horizontal (stationary).

Seasonality – The patterns or cycles of behavior that repeat themselves over time.

ETS decomposition – ETS decomposition is used to separate different components of a time series. The term ETS stands for error, trend, and seasonality.

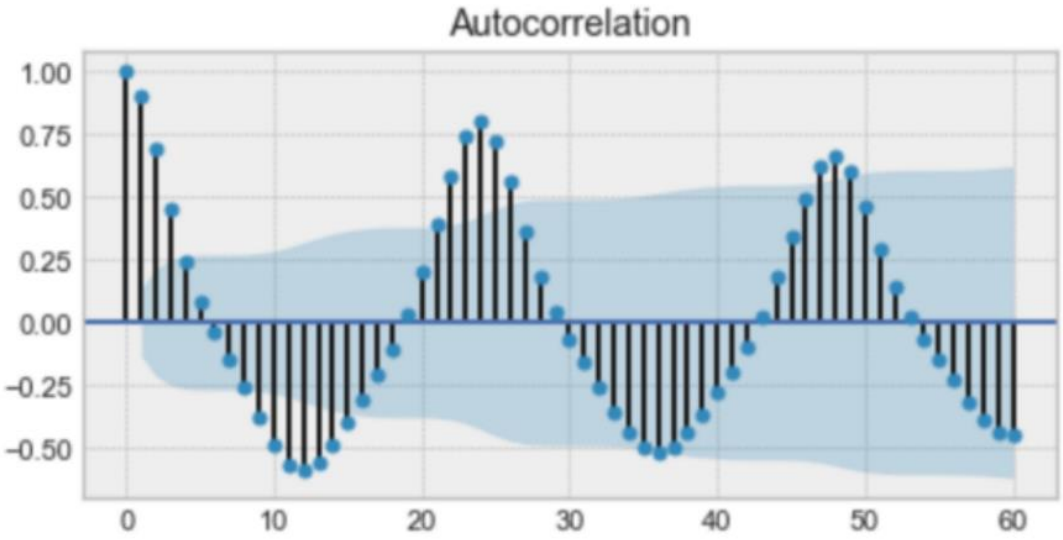
Stationarity – Shows the mean value of the series that remains constant over the time period. If the past effects accumulate and the values increase towards infinity, then stationarity does not hold.

Differentiation – Differentiation is used to make the series stationary and to control for autocorrelations.

Dependence – Refers to the association of two observations of the same variable in previous periods of time.

Noise – The variability in the observations that the model cannot explain.

Autocorrelation – Autocorrelation is the similarity between observations as a function of the time lag between them.



Above is an example of an autocorrelation plot. Looking closely, it is seen that the first value and the value 24 have a high autocorrelation. Similarly, observations 12 and 36 are highly correlated. This means that we will find a very similar value every 24 units of time.

Notice how the graph looks like a sine function. This is a hint of seasonality.

03.Data Exploration

Time Series data points are collected over a period of definite time, either every minute, every day, every month or every year. The crucial part of the data exploration is that a time series data should be continuous over the period of time where our data resides. So that's why we cannot simply remove null valued rows in order to have a clean dataset! Keeping that point in mind, let us explore the dataset we will use.

The dataset that is used in this project is **Air Quality Data in India (2015 - 2020)**.
Find it here: https://www.kaggle.com/datasets/rohanrao/air-quality-data-in-india

The dataset includes air quality data and AQI (Air Quality Index) at hourly and daily level of various stations across multiple cities in India. For the purpose of the project, we may go with the hourly data collection of air quality of multiple cities in India.

Step 01: Importing relevant libraries and packages

- We may use pandas, NumPy libraries for statistical and mathematical purposes.
- Matplotlib and seaborn are used for the visualization purposes.
- Datetime library is used.

Step 02: Importing the dataset

- We may import the dataset and we explore it for some special features like existing columns, number of rows, number of missing values etc.

```
[133] df.head(10)
```

	City	Datetime	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
0	Ahmedabad	2015-01-01 01:00:00	NaN	NaN	1.00	40.01	36.37	NaN	1.00	122.07	NaN	0.0	0.0	0.0	NaN	NaN
1	Ahmedabad	2015-01-01 02:00:00	NaN	NaN	0.02	27.75	19.73	NaN	0.02	85.90	NaN	0.0	0.0	0.0	NaN	NaN
2	Ahmedabad	2015-01-01 03:00:00	NaN	NaN	0.08	19.32	11.08	NaN	0.08	52.83	NaN	0.0	0.0	0.0	NaN	NaN
3	Ahmedabad	2015-01-01 04:00:00	NaN	NaN	0.30	16.45	9.20	NaN	0.30	39.53	153.58	0.0	0.0	0.0	NaN	NaN
4	Ahmedabad	2015-01-01 05:00:00	NaN	NaN	0.12	14.90	7.85	NaN	0.12	32.63	NaN	0.0	0.0	0.0	NaN	NaN
5	Ahmedabad	2015-01-01 06:00:00	NaN	NaN	0.33	15.95	10.82	NaN	0.33	29.87	64.25	0.0	0.0	0.0	NaN	NaN
6	Ahmedabad	2015-01-01 07:00:00	NaN	NaN	0.45	15.94	12.47	NaN	0.45	27.41	191.96	0.0	0.0	0.0	NaN	NaN
7	Ahmedabad	2015-01-01 08:00:00	NaN	NaN	1.03	16.66	16.48	NaN	1.03	20.92	177.21	0.0	0.0	0.0	NaN	NaN
8	Ahmedabad	2015-01-01 09:00:00	NaN	NaN	1.47	16.25	18.02	NaN	1.47	16.45	122.08	0.0	0.0	0.0	NaN	NaN
9	Ahmedabad	2015-01-01 10:00:00	NaN	NaN	2.05	13.78	16.08	NaN	2.05	15.14	NaN	0.0	0.0	0.0	NaN	NaN

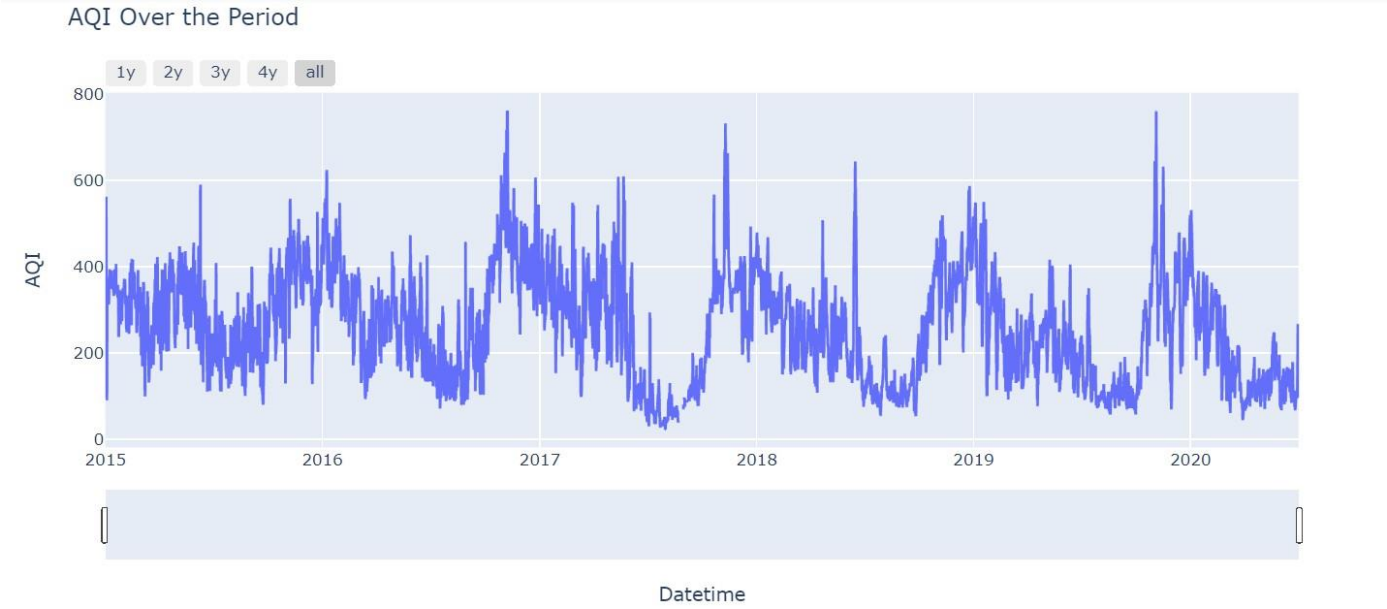
Step 03: Analyzing the data set

- We can first recognize that the 'City' column includes several cities of India as follows:
  Ahmedabad, Aizawl, Amaravati, Amritsar, Bengaluru, Bhopal, Brajrajnagar, Chandigarh, Chennai, Coimbatore, Delhi, Ernakulam, Gurugram, Guwahati, Hyderabad, Jaipur, Jorapokhar, Kochi, Kolkata, Lucknow, Mumbai, Patna, Shillong, Talcher, Thiruvananthapuram, Visakhapatnam
- As we need a time series data set for a continuous period of time, we may choose one city for further steps. The city can be chosen by comparing the total number of null values present in each city and we may pick the least. We can identify 'Delhi' as a city which has the least null values comparing to the other cities. So we move forward with Delhi.
- The data set is spread over a vast period of time : **2015-01-01 to 2020-07-01**

```
[12] print('Time period start: {}\nTime period end: {}'.format(dfDelhi.Datetime.min(),dfDelhi.Datetime.max()))

     Time period start: 2015-01-01 01:00:00
     Time period end: 2020-07-01 00:00:00
```

- We may convert the type of 'Datetime' column from object type to date_time, and we may use 'Datetime' column as our index for easy manipulation.
- We may plot and explore the dataset over the whole period of time; where we can observe any pattern, correlation and missing values.



- The next most important task is to fill the null values. We cannot simply drop null rows as we lose our series data. So we need to follow some special methods of filling null values of a time series dataset.

Step 04: Impute missing values

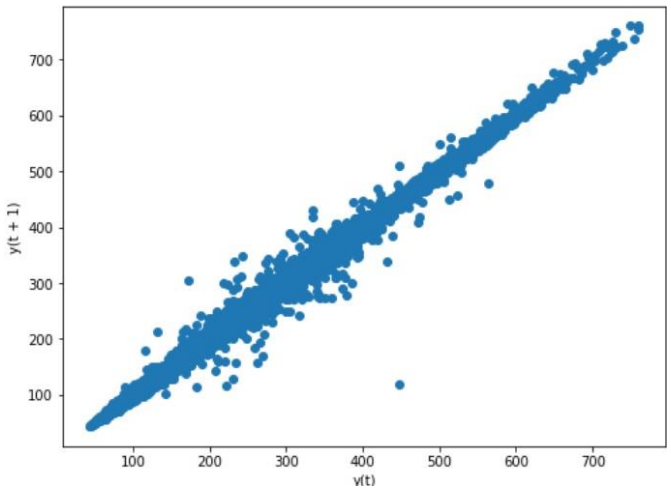
- The real-world data often contain missing values. All types of the dataset including time-series data have the problem with missing values. The cause of missing values can be data corruption or failure to record data at any given time.
- Time Series models work with the complete data and therefore they require to impute the missing values prior to the modeling or actual time series analysis. The missing values in the time series dataset can be handled using two broad techniques:
    1. Drop the record with missing values
    2. Impute the missing information
- Dropping the missing value is however an inappropriate solution, as we may lose the correlation of adjacent observation. Estimating or imputing the missing values can be an excellent approach to dealing with the missing values.
- For this project, we may use 4 different techniques to impute missing information.
    1. Last observation carried forward (forward filling)
    2. Next observation carried backward (backward filling)
    3. Rolling statistical mean
    4. Interpolation
- We may use each of the techniques for a small portion of our dataset using 'AQI' values and we can observe the effectiveness of each method for a given set of null values.

Datetime	AQI	FFILL	BFILL	ROLL2	ROLL7	InterPol
2015-07-04 15:00:00	280.0	280.0	280.0	280.0	280.0	280.0
2015-07-04 16:00:00	280.0	280.0	280.0	280.0	280.0	280.0
2015-07-04 17:00:00	324.0	324.0	324.0	302.0	295.0	324.0
2015-07-04 18:00:00	324.0	324.0	324.0	324.0	302.0	324.0
2015-07-04 19:00:00	321.0	321.0	321.0	322.5	306.0	321.0
2015-07-04 20:00:00	309.0	309.0	309.0	315.0	306.0	309.0
2015-07-04 21:00:00	222.0	222.0	222.0	265.5	294.0	222.0
2015-07-04 22:00:00	NaN	222.0	246.0	222.0	297.0	225.0
2015-07-04 23:00:00	NaN	222.0	246.0	NaN	300.0	229.0
2015-07-05 00:00:00	NaN	222.0	246.0	NaN	294.0	232.0
2015-07-05 01:00:00	NaN	222.0	246.0	NaN	284.0	236.0
2015-07-05 02:00:00	NaN	222.0	246.0	NaN	266.0	239.0
2015-07-05 03:00:00	NaN	222.0	246.0	NaN	222.0	243.0
2015-07-05 04:00:00	246.0	246.0	246.0	246.0	246.0	246.0
2015-07-05 05:00:00	245.0	245.0	245.0	245.5	246.0	245.0
2015-07-05 06:00:00	241.0	241.0	241.0	243.0	244.0	241.0
2015-07-05 07:00:00	239.0	239.0	239.0	240.0	243.0	239.0
2015-07-05 08:00:00	242.0	242.0	242.0	240.5	243.0	242.0
2015-07-05 09:00:00	251.0	251.0	251.0	246.5	244.0	251.0

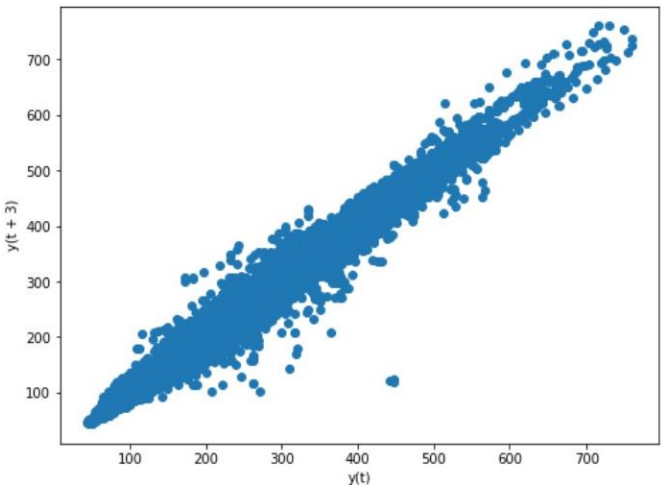
- According to the situation, we may use any of these methods to fill null values of our records.
- As we are more especially focus on the AQI values, we may impute the null values of it accordingly. Then we simply use the 'interpolation' method for the other columns. Finally we may come up with a pure dataset with no any null value.

Step 05: Explore more on 'AQI' values

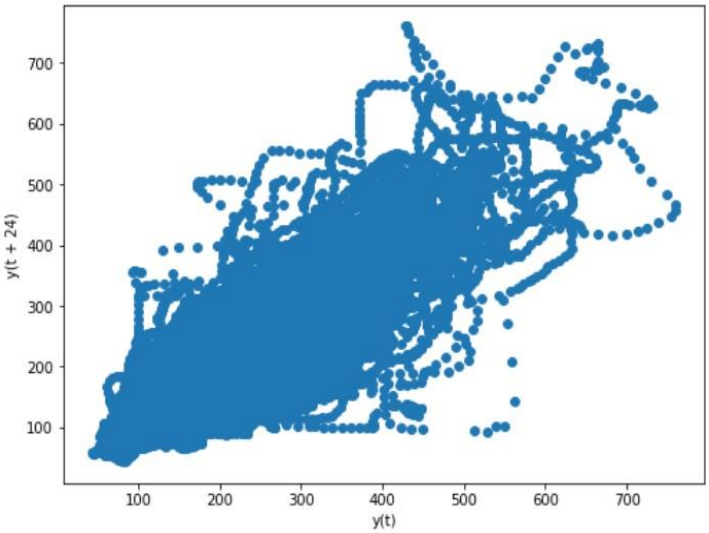
- As we mentioned before, we may use the most suitable technique for imputations of the 'AQI' column.
- The first step is to analyze the autocorrelation fact of 'AQI' values.
- With the help of an autocorrelation plot and several lag plots, we may have an idea of how the current data point is related to previous or future data points of the same feature.



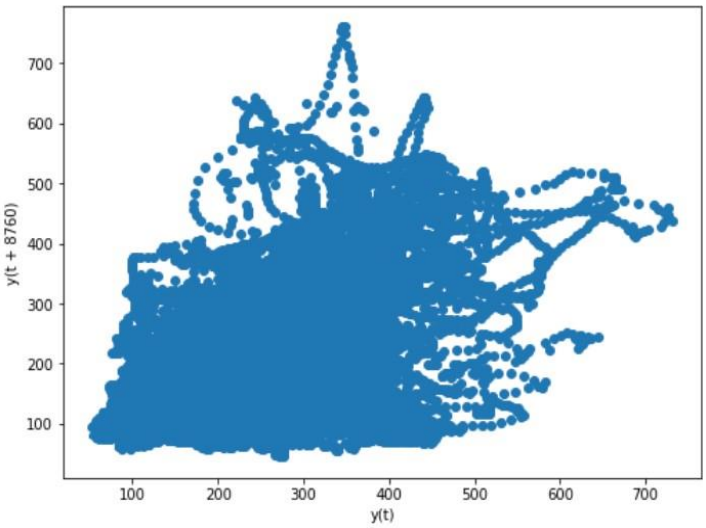
Lag=1: How data points are related to the data points around 1 hour



Lag=2: How data points are related to the data points around 2 hours



Lag=24: How data points are related to the data points around 1 day



Lag=8760: How data points are related to the data points around 1 year

- We can observe that there is a significant relationship between hourly data but more than that, the linear relationship seems to be vanished.
- There is no any significant relationship among yearly data too.
- So we can move forward with **Forward filling, Backward filling, Rolling mean or Interpolation** methods for AQI values. Filling simply with monthly average or past year values will probably not work.

Step 06: Impute missing values of other columns

- We may simply follow the method, interpolation to impute the null values.

More on techniques that can be used to impute missing values in a time series dataset:

1. **Last Observation Carried Forward**
   LOCF is a simple but elegant hack where the previous non-missing values are carried or copied forward and replaced with the missing values.
   ffill() function from Pandas.DataFrame function can be used to impute the missing value with the previous value.

2. **Next Observation Carried Backward**
   NOCB is another simple technique where the next non-missing values are copied and replaced with the previous missing values.
   bfill() function from the Pandas.DataFrame function can be used to impute the missing value with the previous value.

3. **Rolling Statistical**
   Statistical techniques can be used to impute missing values by aggregating the previous non-missing values.

4. **Interpolation**
   Interpolation techniques estimate the missing values by assuming a relationship within a range of data points. In rolling statistical techniques where only the previous values were considered to impute missing values, interpolation technique estimates using past and future known data points.

We have various interpolation methods:

Linear: Assumes linear relationship b/w range of data points.
Spline: Estimates values that minimize overall curvature, thus obtaining a smooth surface passing through the input points.
Time: Estimates missing values by focusing more on nearby points than far away points.

Stationarity & Seasonality:

What is stationarity?

There should be

- Constant mean
- Constant variance
- No seasonality over time.

How to check stationarity?

1. **ACF and PACF plots** – If the time series is stationary, the ACF/PACF plots will show a rapid decrease in correlation after a small amount of lag between points.
2. **Provide moving statistics** – We can plot the moving average or moving variance and see if it varies over time. The moving average/variance is for any moment "t", is the average/variance of the last year, that is, the last 12 months.
3. **Augmented Dickey-Fuller test**: This is one of the statistical tests to check stationarity. Here the null hypothesis is that the TS is not stationary. The test results comprise a test statistic and some critical values for different confidence levels. If the "test statistic" is less than the "critical value", we can reject the null hypothesis and say that the series is stationary.

How to make time series stationary?

There are 2 main reasons behind the non-stationarity of a TS:

1. **Trend** - moving average over time. For example, in this case we saw that, on average, the number of passengers was growing over time.
2. **Seasonality** – Variations in specific time periods. For example, people may have a tendency to buy cars in a particular month due to salary increases or festivals.

We can apply transformations that penalize higher values more than smaller ones.

1. Log scale transformation
2. Exponential transformation
3. Square root transformation

Techniques to remove Trend: Smoothing

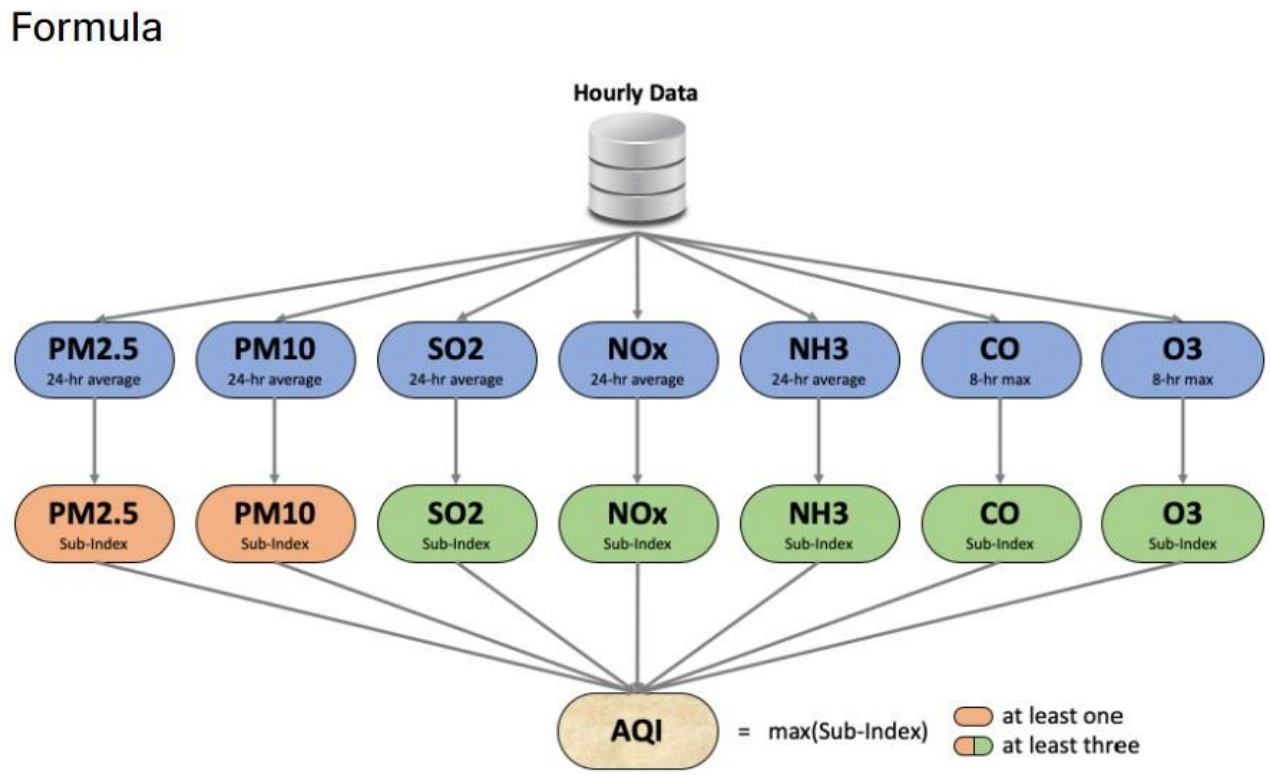
Smoothing is taking moving averages over time windows.

1. Moving Average
   - We take an average of "k" consecutive values depending on the time series frequency.
   - Here we can take the average of the last month, the last 24*30 values.
   - A drawback of this particular approach is that the time period must be strictly defined.
2. Exponentially Weighted Moving Average
   - To overcome the problem of choosing a window defined in the moving average, we can use the exponential weighted moving average.
   - We take a "weighted moving average" where the most recent values are given a higher weight.
   - There can be many techniques for assigning weights. A popular one is the exponentially weighted moving average where weights are assigned to all previous values with a decay factor.

3. Differentiation
   - In this technique, we take the difference between the observation at a particular time with that of the previous time.
4. Decomposition

More on AQI value calculation:

- The AQI calculation uses 7 measures: PM2.5, PM10, SO2, NOx, NH3, CO and O3.
- For PM2.5, PM10, SO2, NOx and NH3 the average value in last 24-hrs is used with the condition of having at least 16 values.
- For CO and O3 the maximum value in last 8-hrs is used.
- PM2.5 is measured in µg/m^3 (micrograms per cubic meter of air)
- PM10 is measured in µg/m^3 (micrograms per cubic meter of air)
- SO_2 is measured in µg/m^3 (micrograms per cubic meter of air)
- NOx is measured in ppb (parts per billion)
- NH_3 is measured in µg/m^3 (micrograms per cubic meter of air)
- CO is measured in mg/m^3 (milligrams per cubic meter of air)
- O_3 is measured in µg/m^3 (micrograms per cubic meter of air)
- The final AQI is the maximum Sub-Index among the available sub-indices with the condition that at least one of PM2.5 and PM10 should be available and at least three out of the seven should be available.



- There is no theoretical upper value of AQI but it's rare to find values over 1000.
- The pre-defined buckets of AQI are as follows:

Good (0–50)	Minimal Impact	Poor (201–300)	Breathing discomfort to people on prolonged exposure
Satisfactory (51–100)	Minor breathing discomfort to sensitive people	Very Poor (301–400)	Respiratory illness to the people on prolonged exposure
Moderate (101–200)	Breathing discomfort to the people with lung, heart disease, children and older adults	Severe (>401)	Respiratory effects even on healthy people

04.Methods of Time-Series Forecasting

Most of the models described under this topic were developed in R and Python. Forecasting can be considered as a subset of supervised regression problems but some specific tools are necessary due to the temporal nature of observations.

A time series is usually modeled through a stochastic process Y(t), i.e. a sequence of random variables. In a forecasting setting, we find ourselves at time t and we are interested in estimating Y(t+h), using only information available at time t.

A stochastic process is any process describing the evolution in time of a random phenomenon. From a mathematical point of view, the theory of stochastic processes was settled around 1950. Since then, stochastic processes have become a common tool for mathematicians, physicists, engineers, and the field of application of this theory ranges from the modeling of stock pricing to a rational option pricing theory to differential geometry.

Time series regression:

Regression models are among the most common types of time series analysis and forecasting techniques. Regression models describe a mathematical relationship between the forecasted variable and a single predictor variable. The most well-known regression model is the linear model. However, nonlinear regression models are extremely popular. Multiple regression models describe a relationship between a forecasted variable and several predictor variables. Understanding the regression models is the basis for understanding more sophisticated time series forecasting methods.

Some of the most relevant models:

Autoregressive (AR) – An autoregressive (AR) model predicts future behavior based on past behavior. It is used to forecast when there is some correlation between the values of a time series and the values that precede and follow them.

Exponential Smoothing (ES) – Exponential smoothing is a time series forecasting method for univariate data that can be extended to support data with a systematic trend or seasonal component.

Autoregressive Integrated Moving Average (ARIMA) – Autoregressive Integrated Moving Average, ARIMA, models are among the most widely used approaches to time series forecasting. Actually, it is a class of models that 'explains' a given time series based on its own past values.

Seasonal Autoregressive Integrated Moving Average (SARIMA) – Seasonal Autoregressive Integrated Moving Average (SARIMA) models extend the basic ARIMA models and allow the incorporation of seasonal patterns.

Prophet – Prophet, which was released by Facebook's Core Data Science team, is an open source library developed by Facebook and designed for automated forecasting of univariate time series data.

LSTM – Long Short-Term Memory (LSTM) is a type of recurrent neural network that can learn the order dependency between elements in a sequence. It is often used to solve time series forecasting problems.

GRU - *GRU architecture is similar to LSTM as they both have gates that control the flow of data within the unit and they are dissimilar as GRU has fewer cells allowing less computation and easier implementation.*

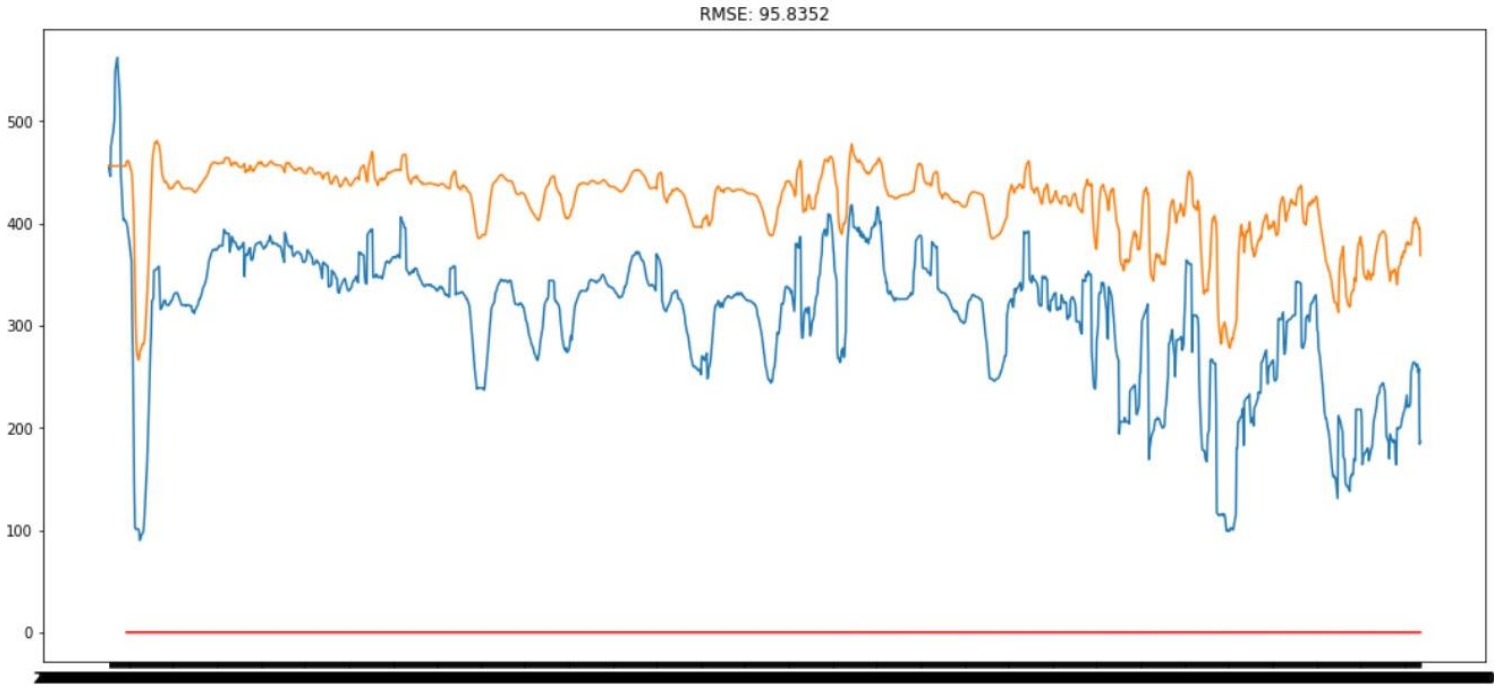
○ AutoRegressive Model (AR)

An autoregression model is a linear regression model that uses lagged variables as input variables.
It models the next step in the sequence as a linear function of the observations in the previous time steps.
We could calculate the linear regression model manually using the *LinearRegession* class in *scikit-learn* and manually specify the lag input variables to use.
Alternately, the *statsmodels* library provides an *autoregression* model where you must specify an appropriate lag value and trains a linear regression model. It is provided in the *AutoReg* class.

Number of AR (autoregressive) terms (p): p is the parameter associated with the autoregressive aspect of the model, which incorporates past values, that is, lags of the dependent variable. For example, if p is 5, the predictors of x(t) will be x(t-1) …….x(t-5).

$$y_{t+1} - \mu = \beta(y_t - \mu) + \epsilon_{t+1}$$

The forecast horizon is the length of time into the future for which forecasts are to be prepared. These generally vary from short-term forecasting horizons (less than three months) to long-term horizons (more than two years).



The blue plot represents the AQI values and the orange plot represents the predicted values for the time period **2015-01-01 16:00:00 : 2015-03-20 00:00:00**

RMSE value is calculated as 95.83.

Error Metric Table of the Model:

	r2_score	mean_absolute_error	median_absolute_error	mse	msle	mape	rmse
0	0.382734	76.515461	66.904993	9184.376556	0.162777	37.222587	95.835153

○ ARIMA

In an ARIMA model there are 3 parameters that are used to help model the main aspects of a time series: seasonality, trend, and noise. These parameters are called p, d and q.

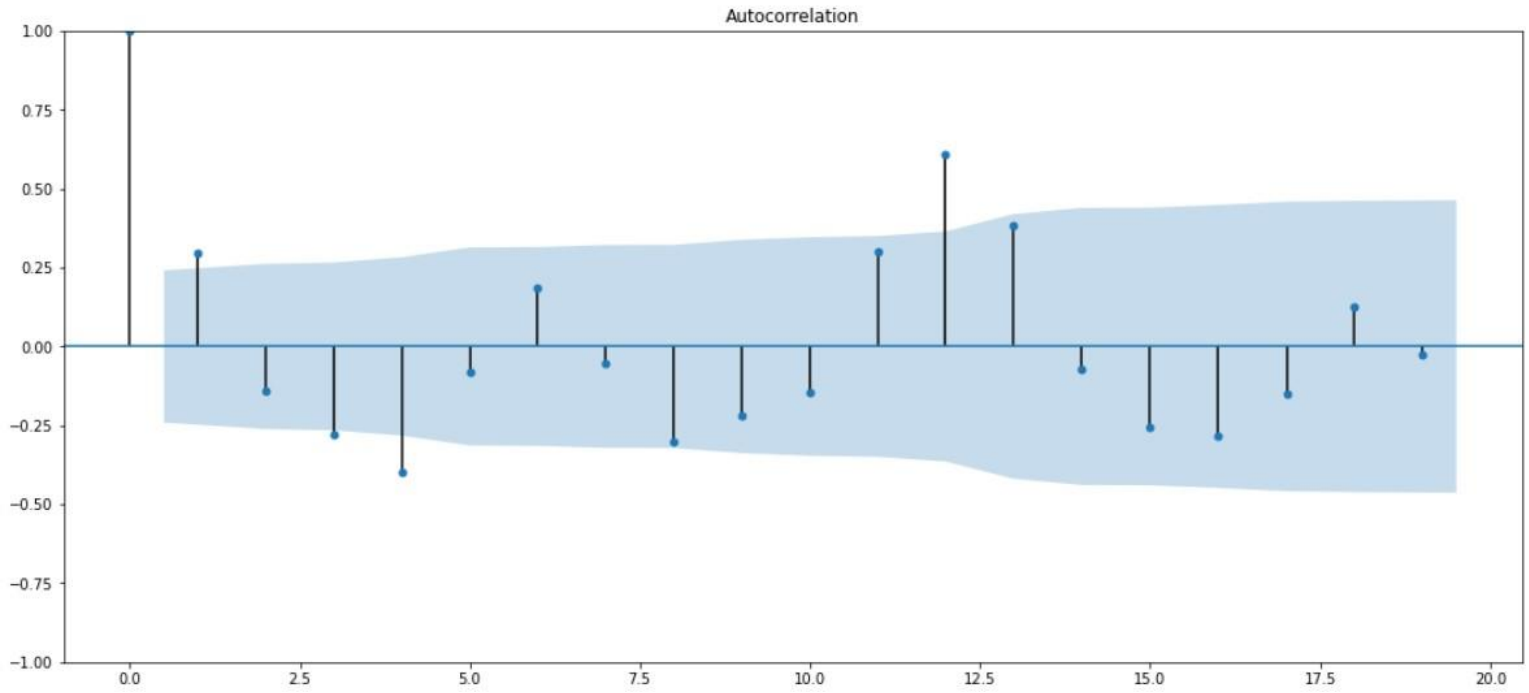
- **Number of AR (Auto-Regressive) terms (p):** p is the parameter associated with the auto-regressive aspect of the model, which incorporates past values, that is, lags of the dependent variable. For example, if p is 5, the predictors of x(t) will be x(t-1)… .x(t-5).
- **Number of differences (d)**: d is the parameter associated with the integrated part of the model, which affects the amount of differentiation to apply to a time series.
- **Number of MA (Moving Average) terms (q)**: q is the size of the partial moving average window of the model, i.e. lagged forecast errors in the forecast equation. For example, if q is 5, the predictors for x (t) will be e (t-1)… .e (t-5) where e(i) is the difference between the moving average at i^{th} instant and the real value.

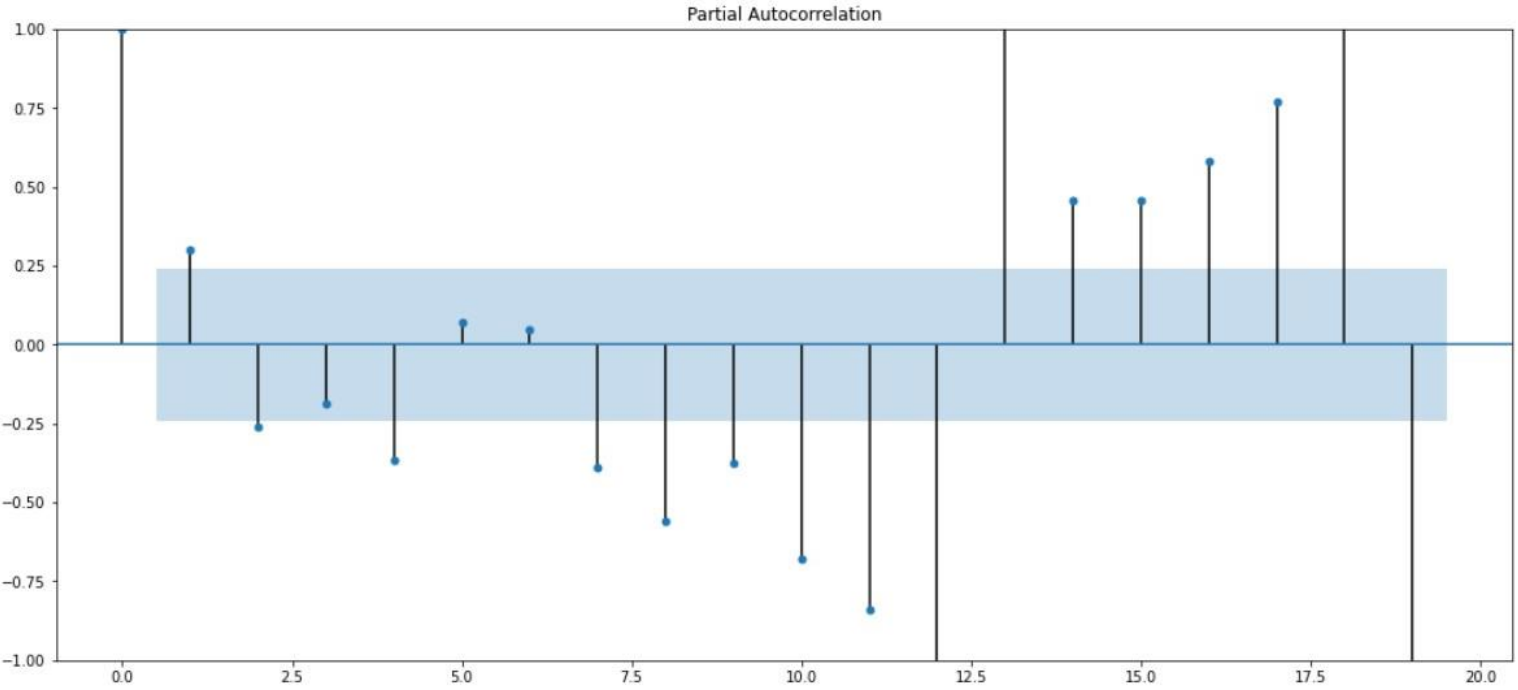
EDA observations on the time series:

- Non-stationarity implies that at least one level of differentiation (d) is required in ARIMA
- The next step is to select the lag values for the Autoregression (AR) and Moving Average (MA) parameters, p and q respectively, using PACF, ACF plots

  Note: One problem with ARIMA is that it does not support seasonal data. That is a time series with a repeating cycle. ARIMA expects data that is not seasonal or has the seasonal component removed, e.g. Seasonally adjusted using methods such as seasonal differentiation.

We will move forward with monthly mean since the hourly dataset is highly stationary.

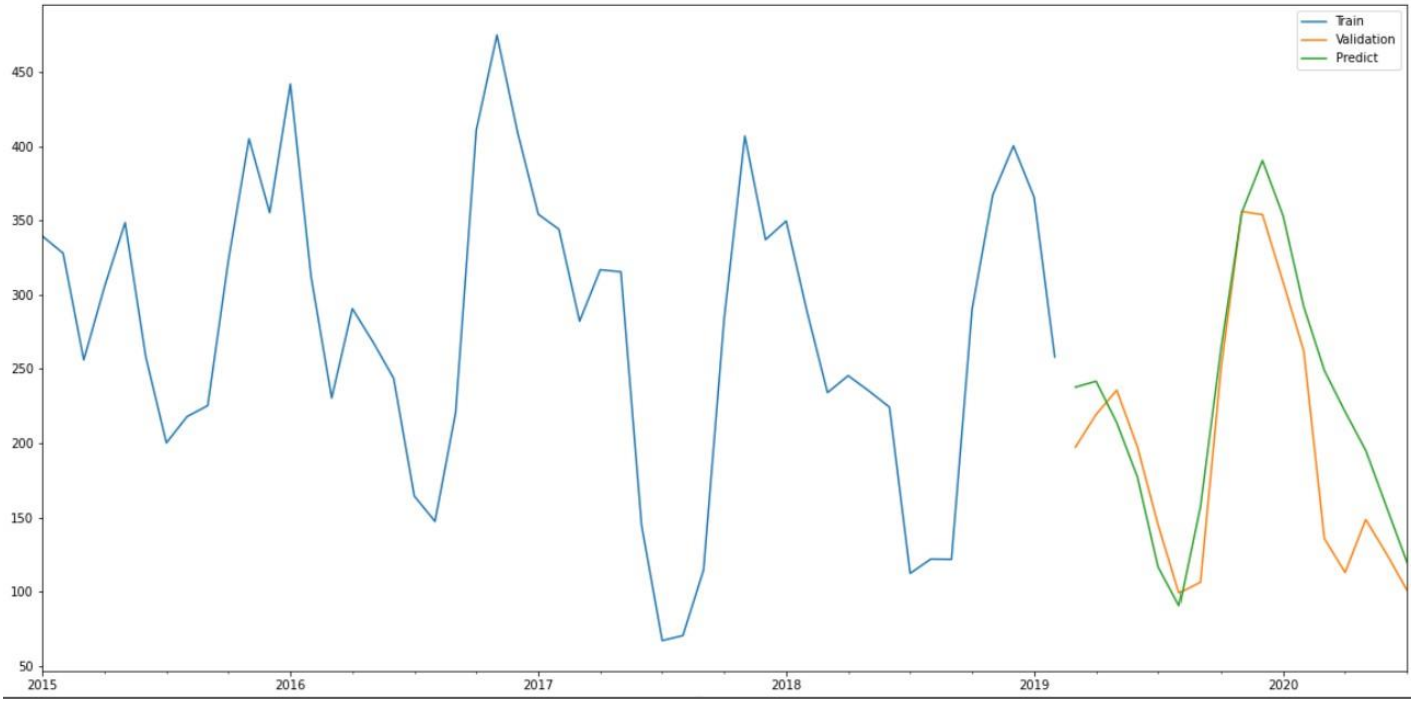




Since the autocorrelation and partial autocorrelation graphs provide significant information on P and Q values, for the dataset observed p,d,q values:

- p – 7
- d – 1
- q – 2

The comparison between predicted values and tested values is given by the below graph.



Accuracy Metrics:

	r2_score	mean_absolute_error	median_absolute_error	mse	msle	mape	rmse
0	0.674028	37.412375	29.82587	2289.46742	0.0748	24.820433	47.848379

○ SARIMA

The seasonal autoregressive integrated moving average, SARIMA or seasonal ARIMA, is an extension of ARIMA that explicitly supports univariate time series data with a seasonal component.

Adds three new hyperparameters specify **autoregression** (AR), **differencing** (I), and **moving average** (MA) for the seasonal component of the series, as well as an additional parameter for the period of seasonality.

Trend items:

There are three trend items that require setup. They are the same as the ARIMA model, specifically:

- p: trend autoregression order.
- d: trend difference order.
- q: Trend moving average order.

Season Elements:

There are four seasonal elements that are not part of ARIMA that need to be configured; are:

- P: seasonal autoregressive order.
- D: seasonal difference order.
- Q: Seasonal moving average order.

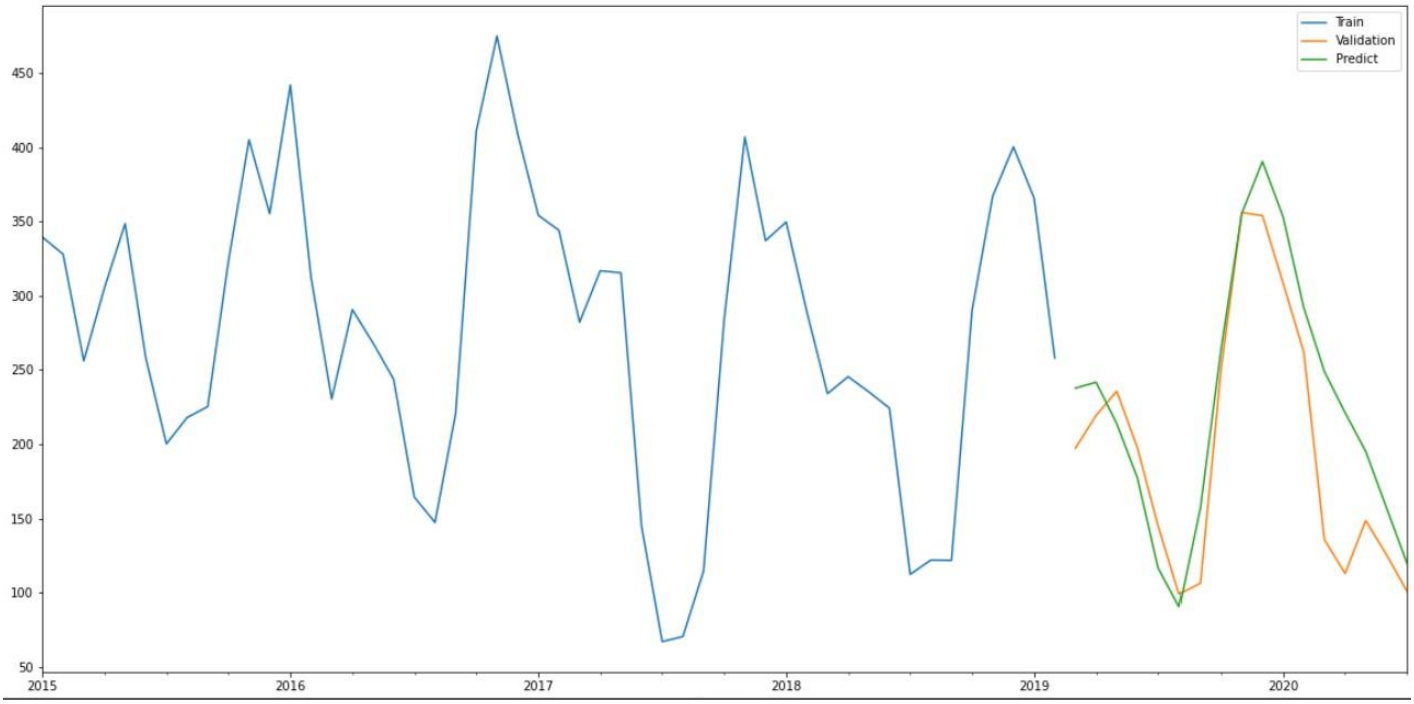
m: the number of time steps for a single seasonal period. For example, an S of 12 for monthly data suggests an annual seasonal cycle.

Sarima Notation: SARIMA (p, d, q) (P, D, Q, m)

Since the autocorrelation and partial autocorrelation graphs provide significant information on P and Q values, for the dataset observed p,d,q values:

- p – 7
- d – 1
- q – 2
- P – 1
- D – 1
- Q – 1
- m - 12

The comparison between predicted values and tested values is given by the below graph.



Accuracy Metrics:

	r2_score	mean_absolute_error	median_absolute_error	mse	msle	mape	rmse
0	0.652215	41.730916	28.73892	2442.668494	0.145709	26.761564	49.42336

○ Prophet

Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

Prophet is an open source software released by Facebook's Core Data Science team. So, Prophet is the Facebook's open source tool for making time-series predictions.

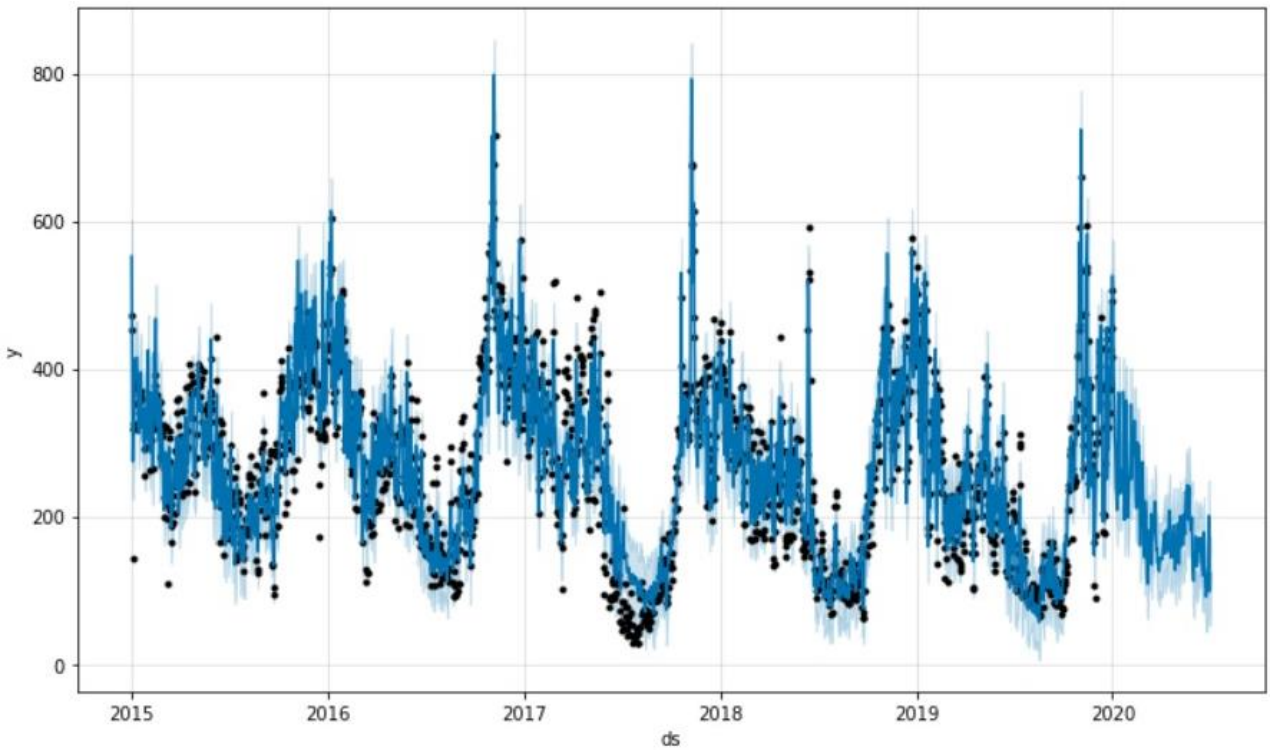
Prophet decomposes time series data into trend, seasonality and holiday effect.

- **Trend:** models non-periodic changes in the time series data.
- **Seasonality**: caused due to periodic changes like daily, weekly, or yearly seasonality.
- **Holiday effect:** which occurs on irregular schedules over a day or a period of days.

- Error terms are what not explained by the *model.h*

*** Daily mean values of each feature is used to reduce complexity.

- By default, 'ds' and 'y' are the independent and dependent columns(we should rename) in the dataset to be trained a prophet model. Other features also can be used for the model by the *'add_regressor'* method.



- The dark blue line is the **yhat** values.
- Light blue area is the lower and upper confident interval bounds.
- Black points are the original data points

Performance Analysis:

	horizon	mse	rmse	mae	mape	mdape	coverage
0	36 days 12:00:00	1880.645139	43.366406	35.415572	0.344638	0.111796	0.708791
1	37 days 00:00:00	1845.734643	42.962014	35.224042	0.335461	0.111796	0.710623
2	37 days 12:00:00	1832.040896	42.802347	34.988011	0.336111	0.111796	0.714286
3	38 days 00:00:00	1878.341992	43.339843	35.466140	0.339048	0.112671	0.705128
4	38 days 12:00:00	1861.002088	43.139333	35.231189	0.339957	0.112671	0.708791
...	...	...	...	...	...	...	...
653	363 days 00:00:00	4328.024902	65.787726	50.603582	0.207479	0.153245	0.595238
654	363 days 12:00:00	4330.577708	65.807125	50.650866	0.210658	0.153245	0.593407
655	364 days 00:00:00	4291.372762	65.508570	50.269651	0.206639	0.149095	0.595238
656	364 days 12:00:00	4286.136949	65.468595	50.227422	0.209043	0.149095	0.593407
657	365 days 00:00:00	4235.592231	65.081428	49.721956	0.205454	0.147708	0.600733

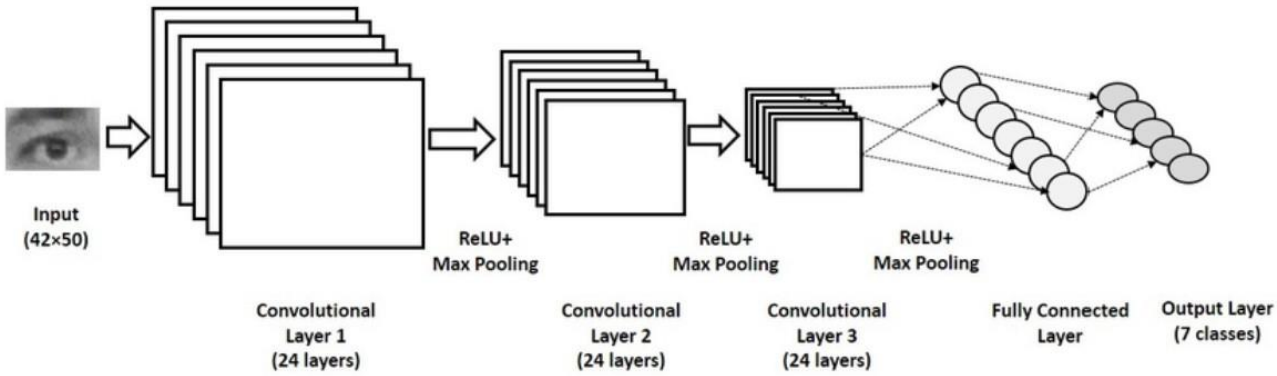
○ 1-D CNN

A convolutional neural network (CNN) is a type of artificial neural network used in image recognition and processing that is specifically designed to process pixel data. CNN is suitable for forecasting time-series because it offers dilated convolutions, in which filters can be used to compute dilations between cells. The size of the space between each cell allows the neural network to understand better the relationships between the different observations in the time-series.

There are various steps to achieve this CNN algorithm, they are as follows,

1. Convolution layer
2. Activation function (ReLu layer)
3. Pooling

4. Flattening
5. Full connection

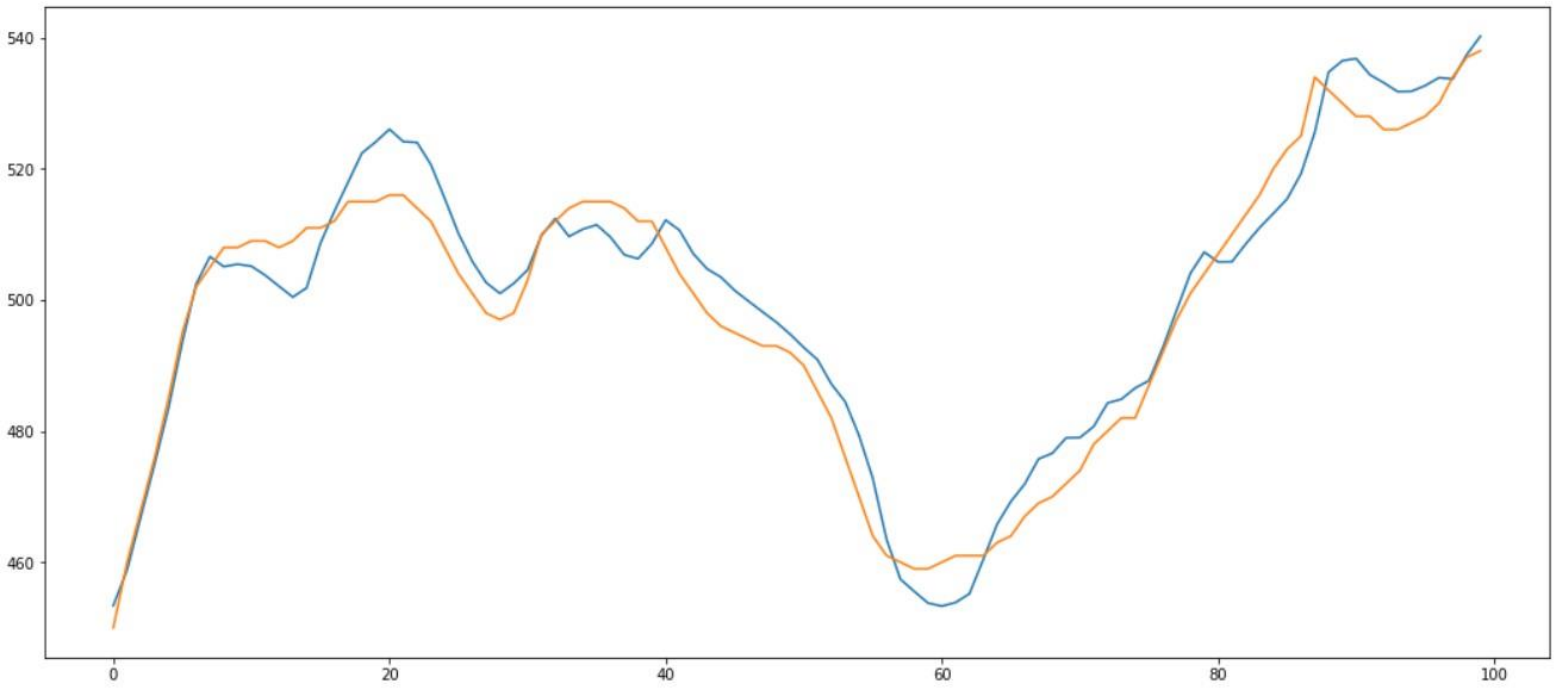


Model Specifications

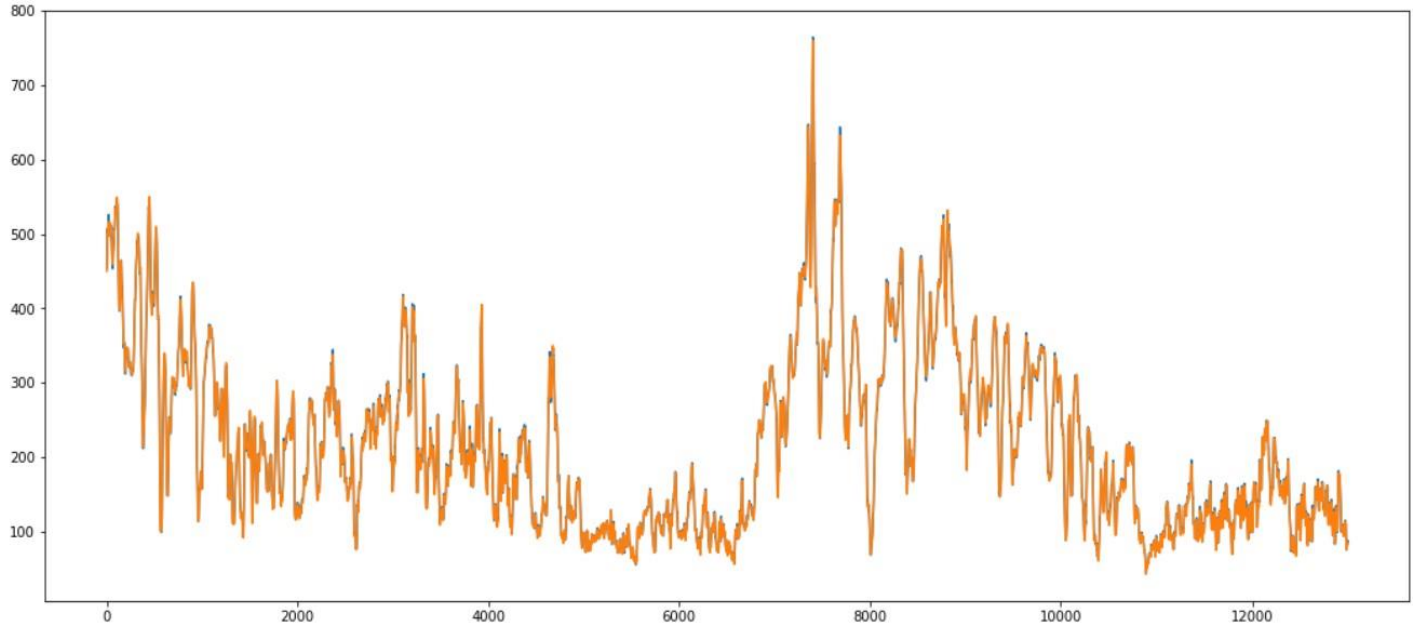
For the AQI value prediction, LSTM has been used as a multivariate time series model where, all the attributes including time has been used to train the model.

Number of layers in total: 4
Convolution Layer: (0, 64) as output dimension with kernel size=2
Epochs: 10

Model Accuracy



Actuals vs Predicted AQI for first 500 test cases



Actuals vs Predicted AQI for 13000 test cases

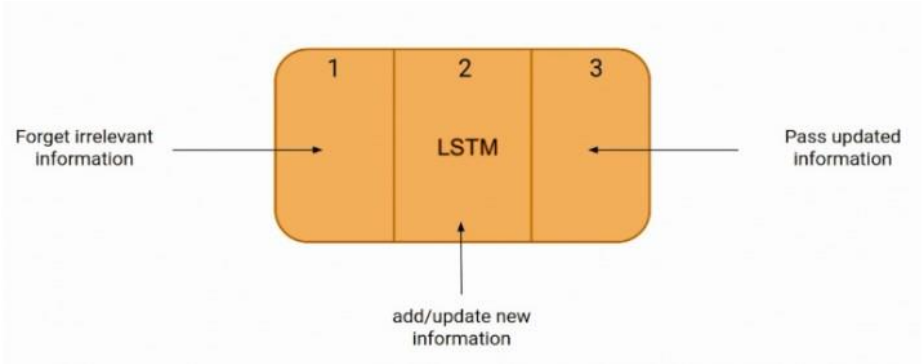
	r2_score	mean_absolute_error	median_absolute_error	mse	msle	mape	rmse
0	0.998989	2.667251	2.031097	13.26637	0.000457	1.499465	3.642303

○ LSTM

Long Short-Term Memory Network is an advanced RNN, a sequential network, that allows information to persist. It is capable of handling the vanishing gradient problem faced by RNN. A recurrent neural network is also known as RNN is used for persistent memory. Similarly, RNNs work, they remember the previous information and use it for processing the current input. The shortcoming of RNN is, they cannot remember Long term dependencies due to vanishing gradient. LSTMs are explicitly designed to avoid long-term dependency problems.

LSTM Architecture

the internal functioning of the LSTM network,

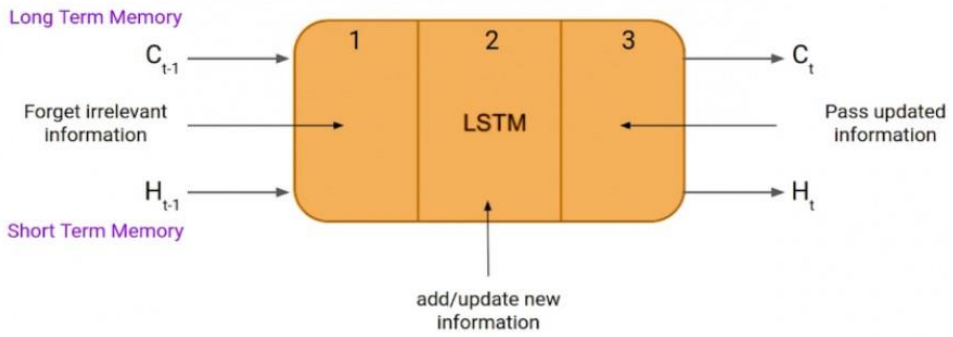


The first part chooses whether the information coming from the previous timestamp is to be remembered or is irrelevant and can be forgotten. In the second part, the cell tries to learn new information from the input to this cell. At last, in the third part, the cell passes the updated information from the current timestamp to the next timestamp.

These three parts of an LSTM cell are known as gates. The first part is called Forget gate, the second part is known as the Input gate and the last one is the Output gate.

Just like a simple RNN, an LSTM also has a hidden state where H(t-1) represents the hidden state of the previous time--stamp and H(t) is the hidden state of the current timestamp. In addition to that LSTM also have a cell state represented by C(t-1) and C(t) for previous and current timestamp respectively.

Here the hidden state is known as Short term memory and the cell state is known as Long term memory. Refer to the following image.

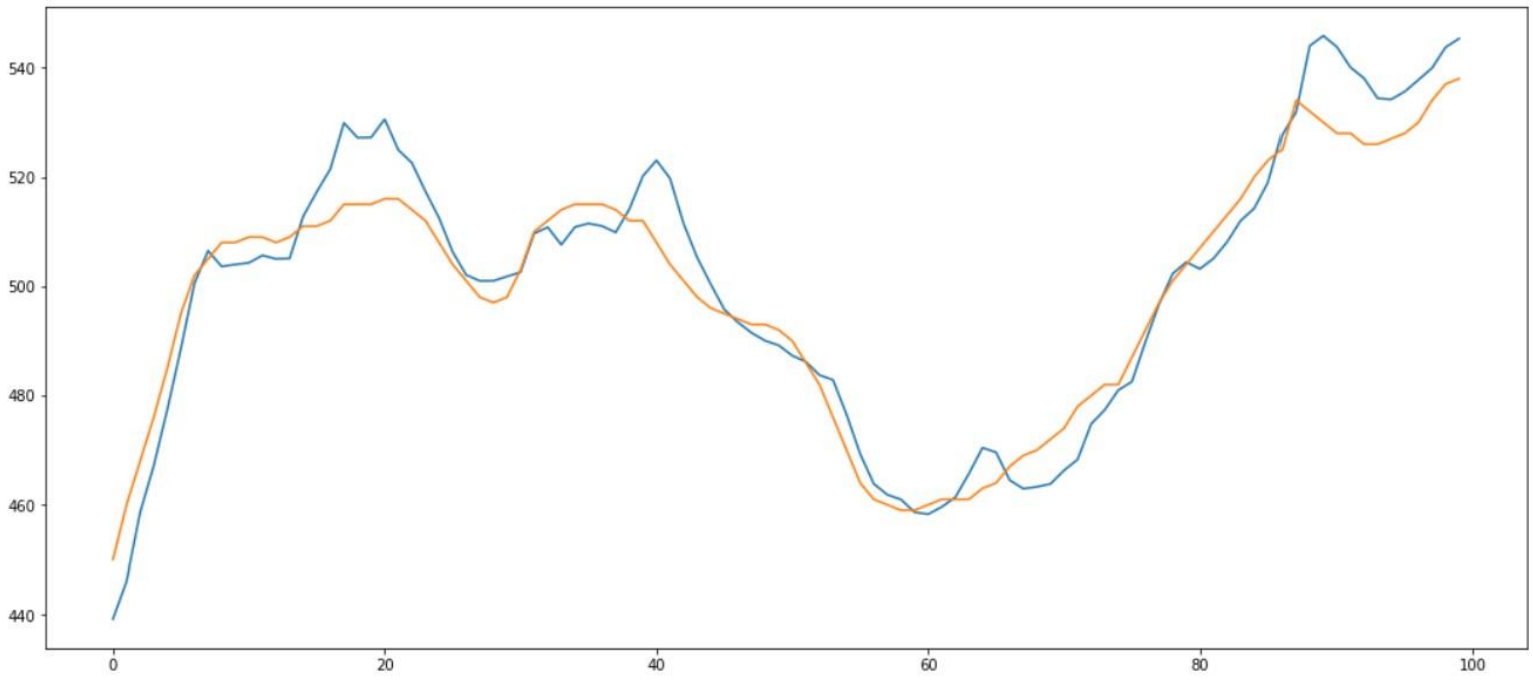


Model Specifications

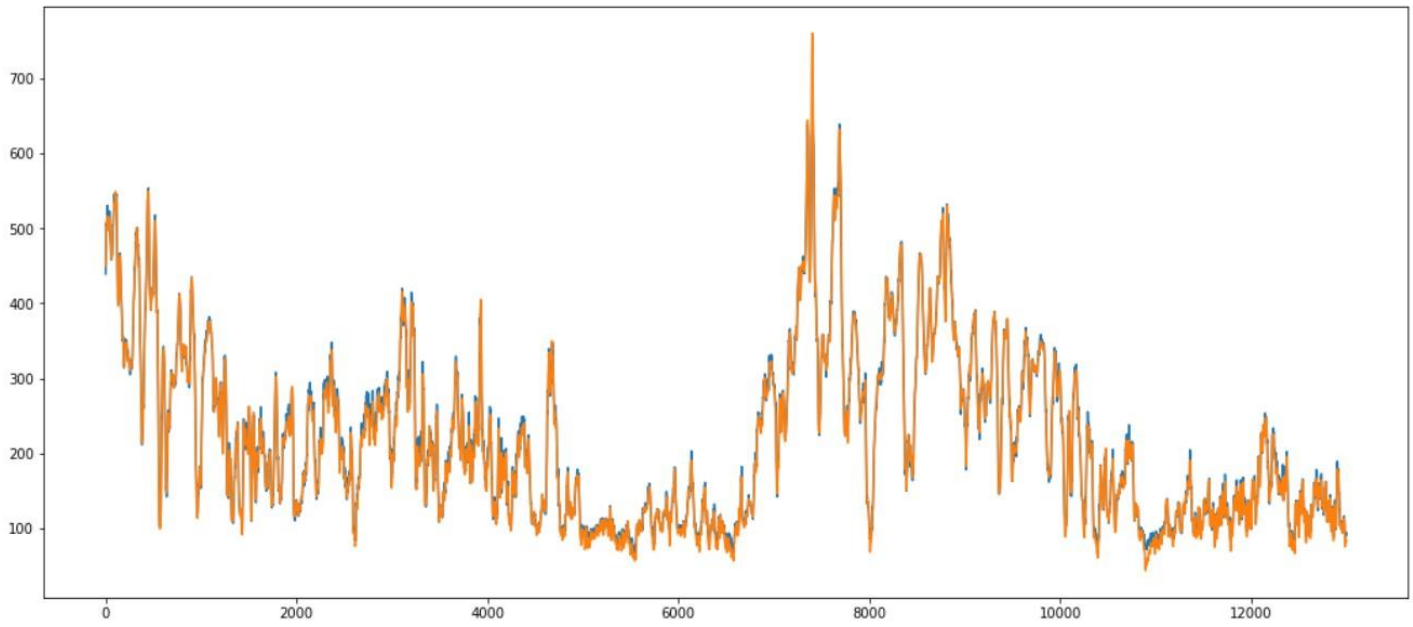
For the AQI value prediction, LSTM has been used as a multivariate time series model where, all the attributes including time has been used to train the model.

Number of layers in total : 4
LSTM Layer : (0, 64) as output dimension
Epochs : 30

Model Accuracy



Actuals vs Predicted AQI for first 100 test cases



Actuals vs Predicted AQI for 13000 test cases

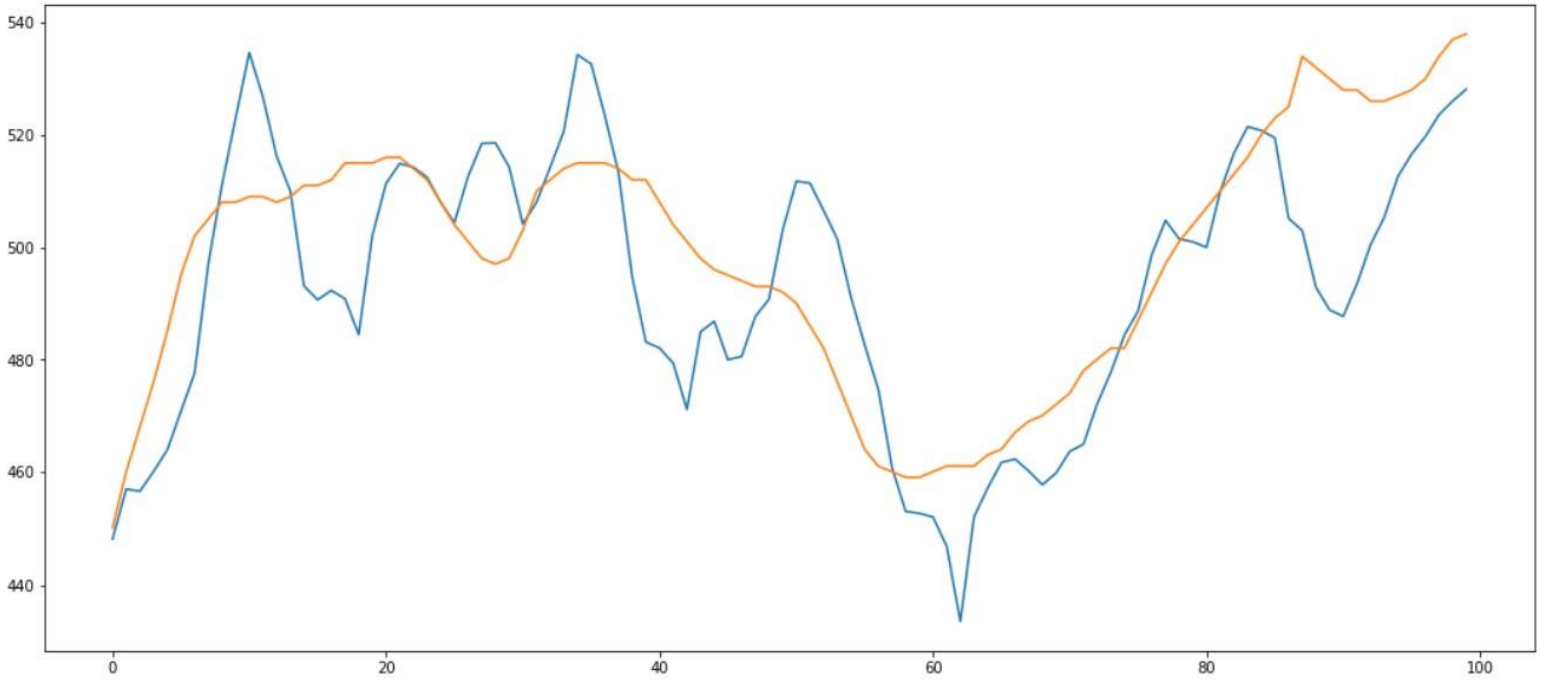
	r2_score	mean_absolute_error	median_absolute_error	mse	msle	mape	rmse
0	0.996309	4.650987	3.267536	48.45126	0.002468	3.068429	6.960694

○ 2-LSTM Layers

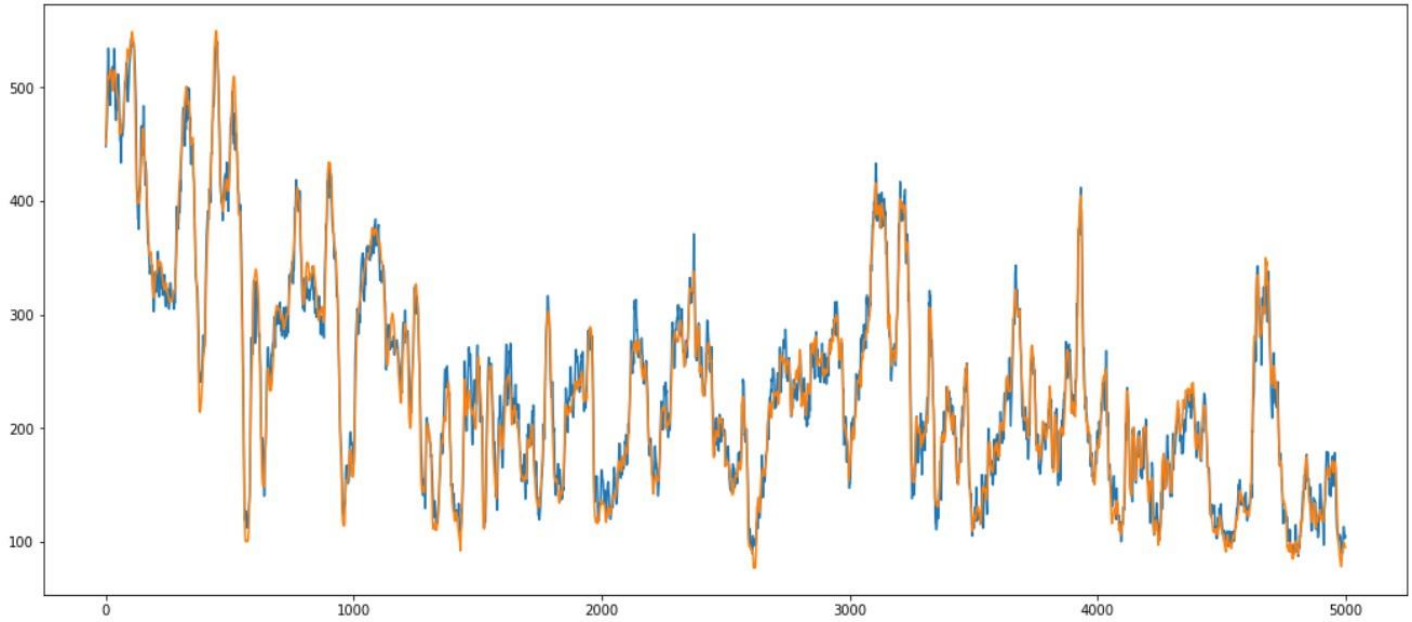
Model Specifications

Number of layers in total : 4
LSTM Layer 1 : (0, 32) as output dimension
LSTM Layer 1 : (0, 64) as output dimension
Epochs : 30

Model Accuracy



Actuals vs Predicted AQI for first 100 test cases



Actuals vs Predicted AQI for all test cases

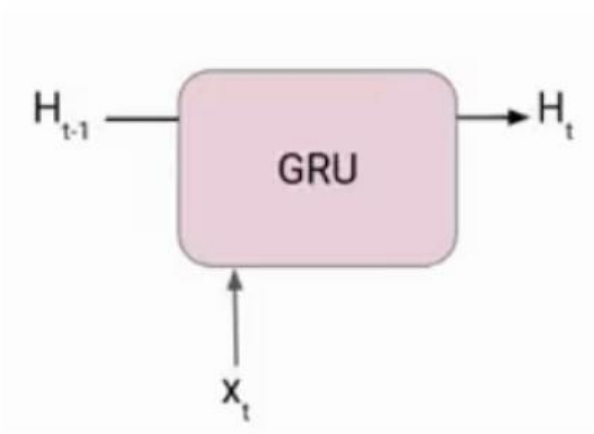
	r2_score	mean_absolute_error	median_absolute_error	mse	msle	mape	rmse
0	0.979224	11.835542	9.729752	272.739417	0.00758	6.700054	16.514824

GRU

GRU or Gated recurrent unit is an advancement of the standard RNN i.e recurrent neural network. It was introduced by ***Kyunghyun Cho et al*** in the year 2014. GRUs are very similar to Long Short Term Memory(LSTM). Just like LSTM, GRU uses gates to control the flow of information. They are relatively new as compared to LSTM. This is the reason they offer some improvement over LSTM and have simpler architecture.

Another Interesting thing about GRU is that, unlike LSTM, it does not have a separate cell state (Ct). It only has a hidden state(Ht). Due to the simpler architecture, GRUs are faster to train.

Architecture of Gated Recurrent Unit



At each timestamp t, it takes an input Xt and the hidden state Ht-1 from the previous timestamp t-1. Later it outputs a new hidden state Ht which again passed to the next timestamp.

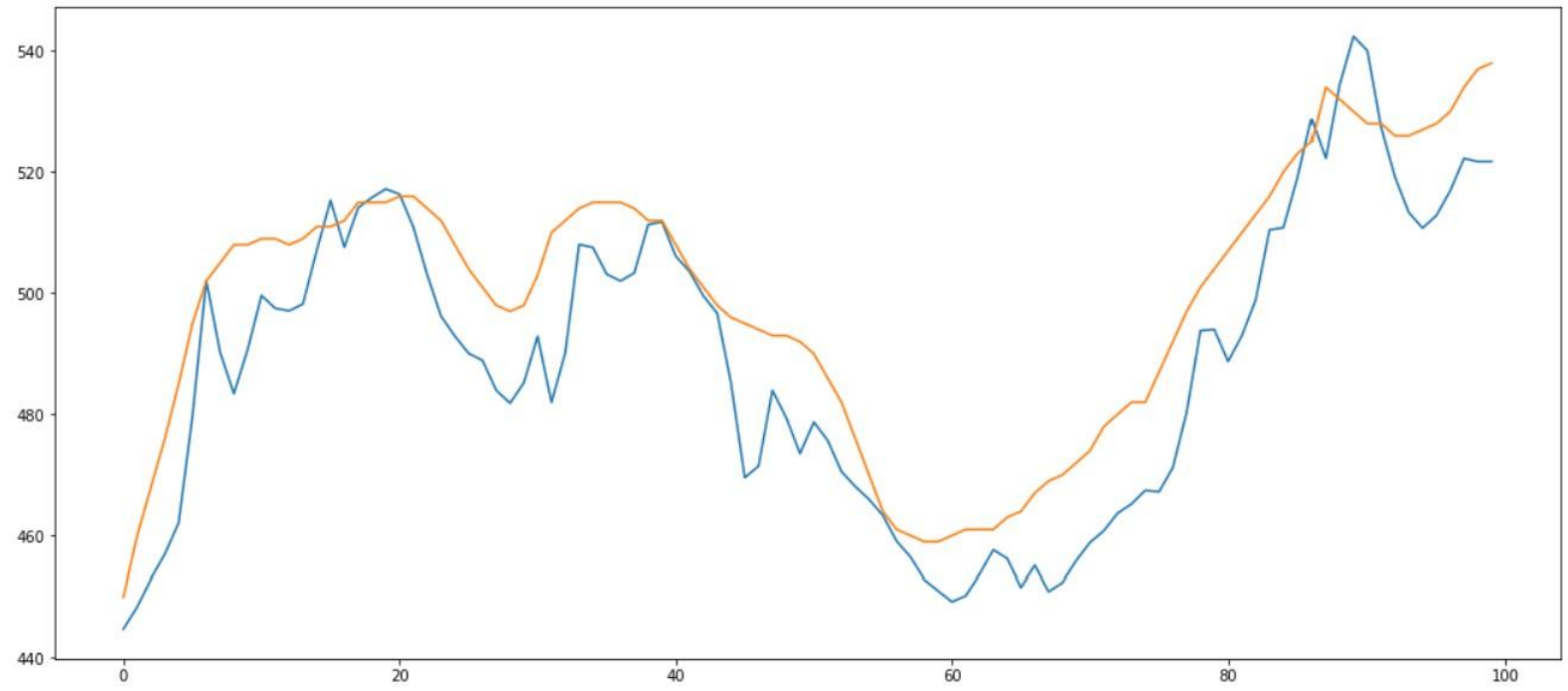
Now there are primarily two gates in a GRU as opposed to three gates in an LSTM cell. The first gate is the Reset gate and the other one is the update gate.

Model Specifications

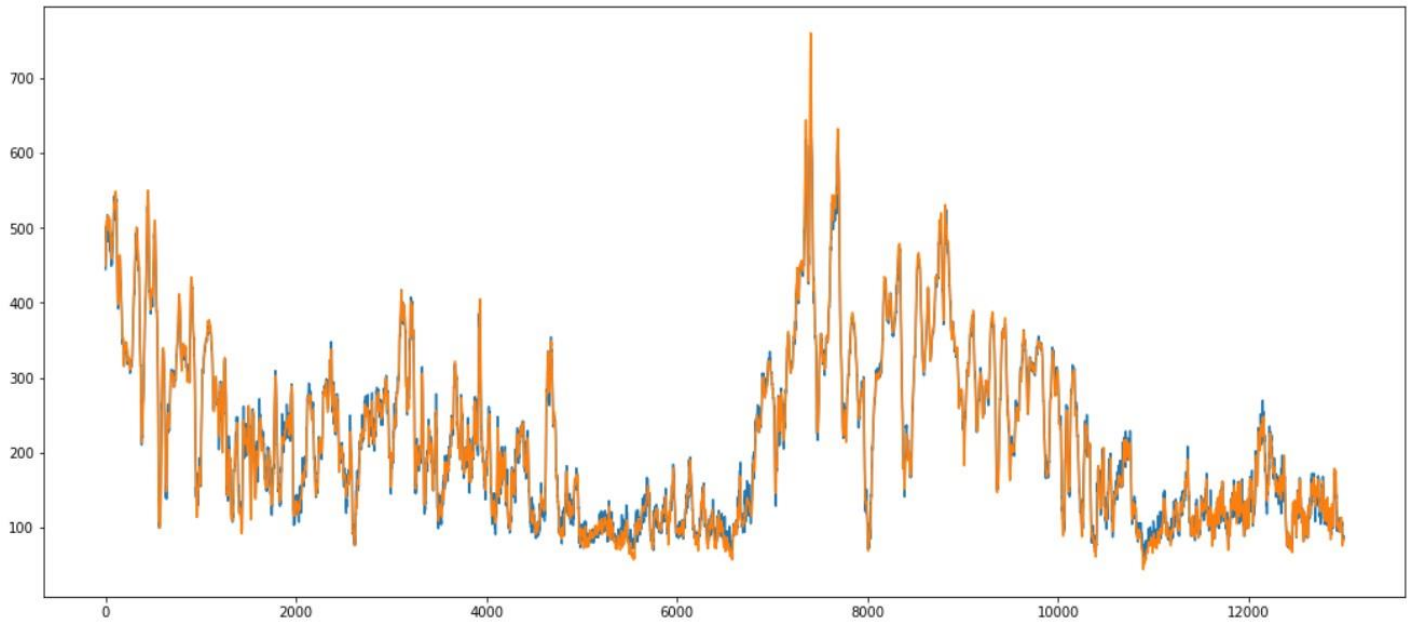
For the AQI value prediction, GRU has been used as a multivariate time series model where, all the attributes including time has been used to train the model.

Number of layers in total : 4
GRU Layer : (0, 64) as output dimension
Epochs : 30

Model Accuracy



Actuals vs Predicted AQI for first 100 test cases

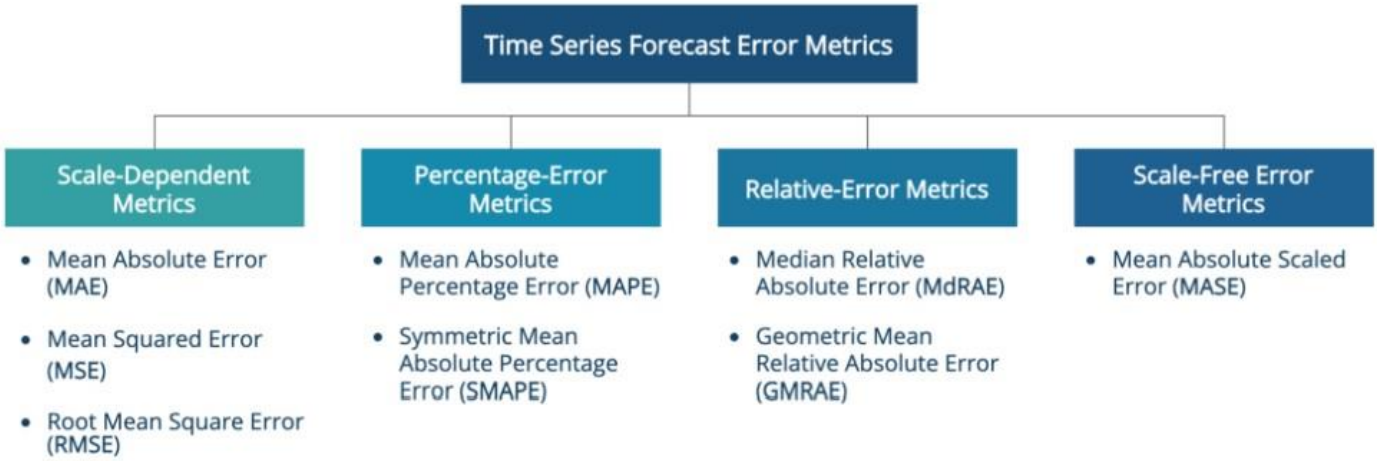


Actuals vs Predicted AQI for 13000 test cases

	r2_score	mean_absolute_error	median_absolute_error	mse	msle	mape	rmse
0	0.993259	6.792429	5.309525	88.490493	0.003824	4.314875	9.406939

05. Error Metrics

Whether a model is working well or not, it is necessary to have different scoring metrics to know the quality of the model and how it is predicted, we have different metrics to evaluate this.



We will apply the below error metrics to evaluate the models we are using for the time series predictions.

1. **R-Squared Error**
2. **Mean Absolute Error**
3. **Mean Squared Error**
4. **Log Mean Square Error**
5. **Mean Absolute Percentage Error**

- **Scale Dependent Metrics:** Scale-dependent means the error metrics are expressed in the units (i.e. Dollars, Inches, etc.) of the underlying data. The main advantage of scale-dependent metrics is that they are usually easy to calculate and interpret. However, they cannot be used to compare different series, because of their scale dependency (Hyndman, 2006).
- **Percentage Error Metrics:** They are scale independent and used to compare forecast performance between different time series. However, their weak spots are zero values in a time series. Then they become infinite or undefined which makes them not interpretable (Hyndman 2006).

1. R-Squared Error

A statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model. Whereas correlation explains the strength of the relationship between an independent and dependent variable, R-squared explains to what extent the variance of one variable explains the variance of the second variable. So, if the R2 of a model is 0.50, then approximately half of the observed variation can be explained by the model's inputs.

Formula for R-Squared

$$R^2 = 1 - \frac{\text{Unexplained Variation}}{\text{Total Variation}}$$

You would calculate predicted values, subtract actual values and square the results. This yields a list of errors squared, which is then summed and equals the unexplained variance.

To calculate the total variance, you would subtract the average actual value from each of the actual values, square the results and sum them. From there, divide the first sum of errors (explained variance) by the second sum (total variance), subtract the result from one, and you have the R-squared.

2. Mean Absolute Error

The Mean Absolute Error (MAE) is calculated by taking the mean of the absolute differences between the actual values (also called y) and the predicted values (y_hat).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

3. Mean Squared Error

If we want to put more attention on outliers (huge errors) you can consider the Mean Squared Error (MSE). Like its name implies it takes the mean of the squared errors (differences between y and y_hat). Due to its squaring, it heavily weights large errors more than small ones, which can be in some situations a disadvantage. Therefore the MSE is suitable for situations where you really want to focus on large errors. Also we need to keep in mind that due to its squaring the metric loses its unit.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

4. Root Mean Squared Error

To avoid the MSE's loss of its unit we can take the square root of it. The outcome is then a new error metric called the Root Mean Squared Error (RMSE). It comes with the same advantages as its siblings MAE and MSE. However, like MSE, it is also sensitive to outliers.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

5. *Mean Squared Logarithmic Error*

Mean squared logarithmic error (MSLE) can be interpreted as a measure of the ratio between the true and predicted values. The introduction of the logarithm makes MSLE only care about the relative difference between the true and the predicted value, or in other words, it only cares about the percentual difference between them. MSLE also penalizes underestimates more than overestimates, introducing an asymmetry in the error curve. The loss is the mean over the seen data of the squared differences between the log-transformed true and predicted values, or writing it as a formula:

$$L(y, \hat{y}) = \frac{1}{N}\sum_{i=0}^{N}(\log(y_i + 1) - \log(\hat{y}_i + 1))^2$$

6. *Mean Absolute Percentage Error*

The mean absolute percentage error (MAPE) is one of the most popular used error metrics in time series forecasting. It is calculated by taking the average (mean) of the absolute difference between actuals and predicted values divided by the actuals. MAPE's advantages are it's scale-independency and easy interpretability. As said at the beginning, percentage error metrics can be used to compare the outcome of multiple time series models with different scales.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i} * 100\right|$$

06. Model Comparison

We maintained an accuracy metric table for each of the model and the following table represents the error values obtained by each of the model we used to train and predict AQI values.

Model	Type	Test Cases	R2-Score	Mean Absolute Error	Median Absolute Error	Mean Squared Error	Mean Squared Log error	Mean Absolute Percentage Error	Root Mean Squared Error
Autoregressive (AR)	Univariate	48177 (hourly)	0.383	76.515	66.905	9184.376	0.1627	37.22	95.835
ARIMA	Univariate	18(monthly)	0.674	37.412	29.825	2289.487	0.0748	24.82	47.838
SARIMA	Univariate	18(monthly)	0.652	41.731	28.739	2442.668	0.1457	26.76	49.423
Prophet	Multivariate	657(daily)	-	49.722	-	4235.592	-	20.54	65.081
1D-CNN	Multivariate	13000(hour)	0.998	2.667	2.031	13.266	0.0005	1.50	3.642
LSTM	Multivariate	13000(hour)	0.996	4.651	3.267	48.451	0.0025	3.07	6.961
LSTM 2-Layers	Multivariate	13000(hour)	0.979	11.835	9.729	272.739	0.0076	6.70	16.515
GRU	Multivariate	13000(hour)	0.993	6.792	5.309	88.490	0.0038	4.32	9.407

Since the Application we are going to develop should represent a live AQI prediction, the best possible way of selecting the dataset should be hourly. So in that case observing the above error metrics of each model we can conclude that 1D-CNN has been the best accurate model to forecast AQI values. Since it has taken hourly data to train the model, we can assure that 1D-CNN is the best model that perfectly fit to our hourly AQI dataset.

Now as per AQI, we can forecast every pollutant using 1D-CNN model and to seek for each pollutant and how accurately the model is going to predict future AQI values.

Conclusion

- We may move forward with either CNN, LSTM or GRU according the accuracy of each pollutant prediction. We can move forward with a combination of above models that may be investigated further.

Thank You.