# Time Series

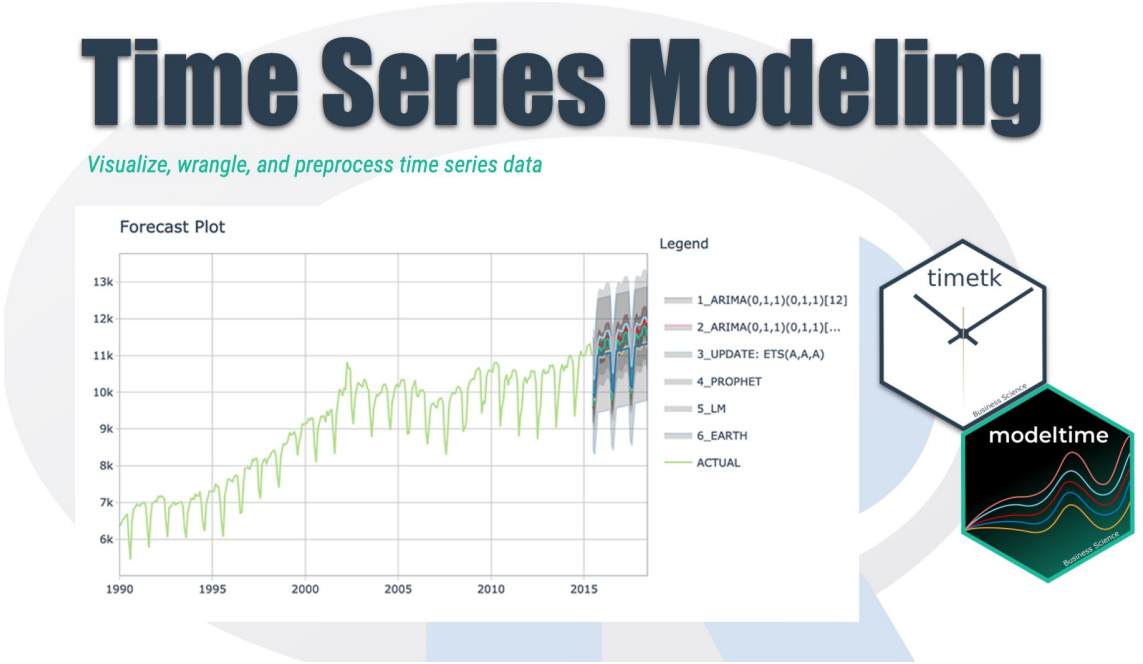
Autoregression Analysis

# Agenda

- Introduction
- Time Series Basic Rules
- Preprocess Data
- Plots
- Categories & Tests
- Auto Regressor
- Rolling Window

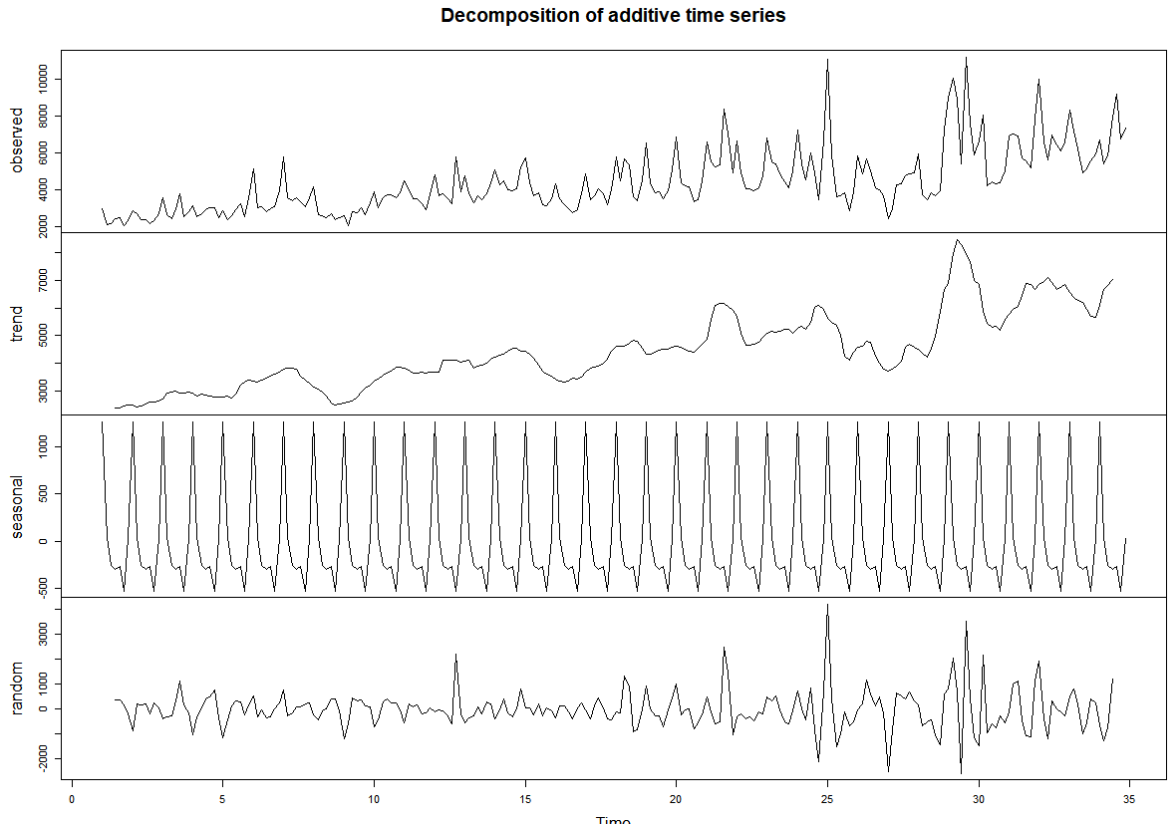
# Time Series Introduction

- We're going to discover how it differs from other types of data you've previously encountered and why
- Time series is a sequence of information which attaches a time period to each value
- The value can be pretty much anything measurable.
- It depends on time in some way, like prices, humidity or number of people.
- As long as the values we record are unambiguous, any medium could be measured with Time series.
- There aren't any limitations regarding the total time span of our Time series.
- It could be a minute, a day, a month or even a century.
- All we need is a starting and an ending point.



# Analyse Time series in a meaningful way

- All time periods must be equal and clearly defined, which would result in a constant frequency.
- This frequency is a measurement of time and could range from a few milliseconds to several decades.
- Furthermore, patterns observed in Time series are expected to persist in the future. That is why we often try to predict the future by analyzing recorded values.
- if the data is not ordered chronologically, finding the correct pattern would be extremely difficult.
- For instance, simply knowing the highest temperature for the last five days would be useless unless

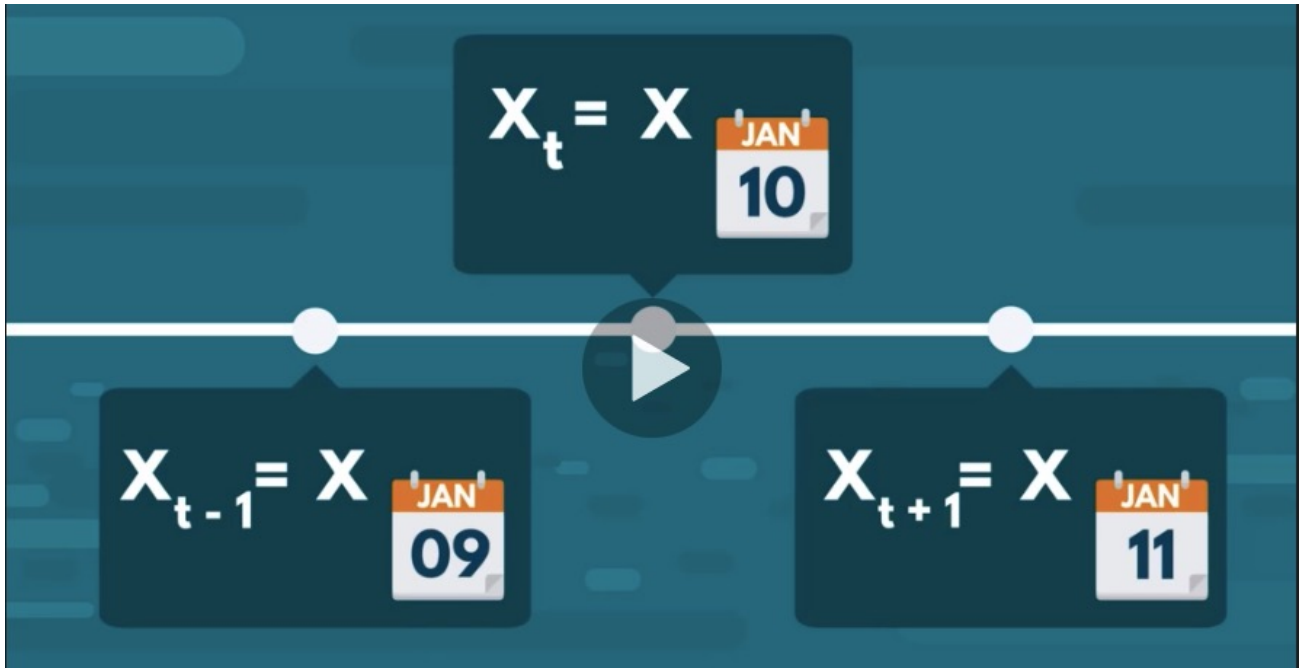


# Application Time Series

- Time series data is largely applied in finance for investors as well as company owners.
- It's crucial to determine whether prices, returns, profits and sales will increase or decrease in the future
- This means the values for every period are not only affected by outside factors, but also by the values of past periods.
- For instance, we expect tomorrow's temperature outside to be within some reasonable proximity to today’s values.

# Notations

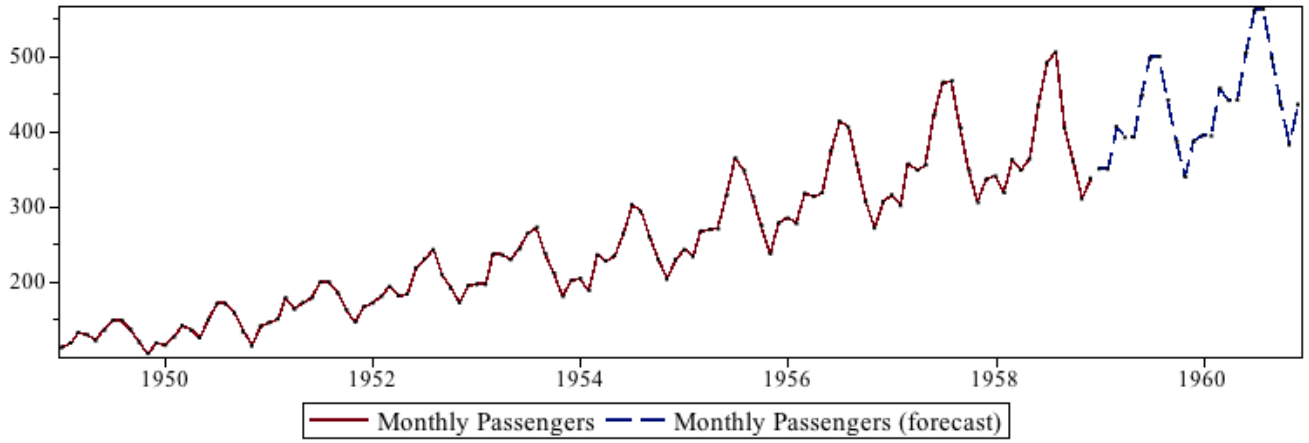
- T represents the entire year, the lowercase T would represent a single day then to denote the closing price at a specific day, we would use X of T.
- This notation will be extremely helpful when we try to model time series data to make predictions about the future.



# Time Series Logic - Frequency

- Usually there are many points in between the start and the end of a time series, the interval of time between recording one point of the set and the next is called a **time period**.
- How often values of the data set are recorded is referred to as the **frequency** of the data set to be able to analyze Time series in a meaningful way.
- All time periods must be equal and clearly defined, which would result in a constant frequency.
- This frequency is a measurement of time and could range from a few milliseconds to several decades.

# Time Series Basics



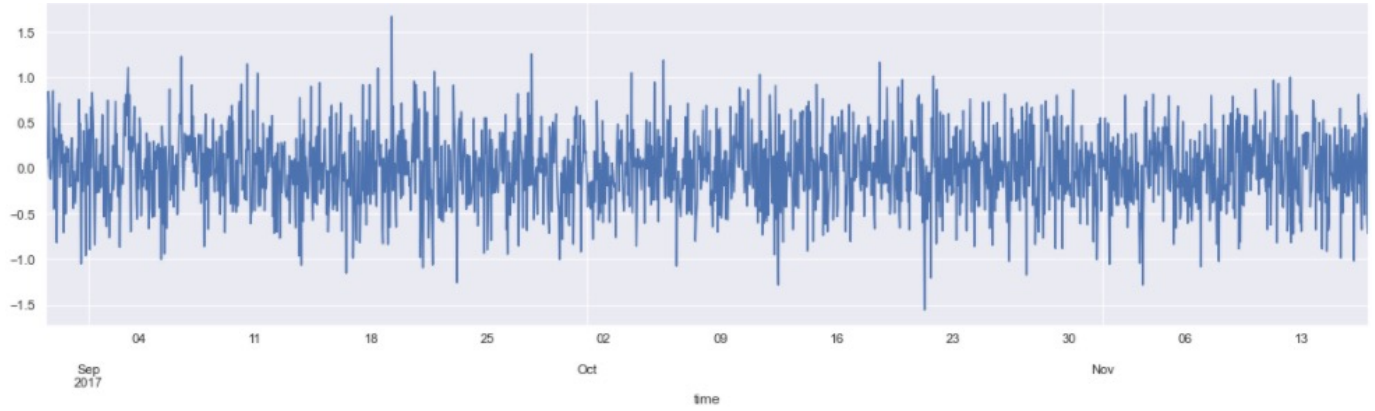

- Chronological Data
- Cannot be shuffled
- Each row indicate specific time record
- Train – Test split happens chronologically
- Data is analyzed univariately (for given use case)
- Nature of the data represents if it can be predicted or not

| time | IOT_Sensor_Reading | Error_Present | Sensor_2 | Sensor_Value |
|---|---|---|---|---|
| 2017-08-29 11:00:00 | -0.015871 | 0.353986 | -0.787655 | 0.008144 |
| 2017-08-29 12:00:00 | -0.101576 | 0.353986 | -0.787655 | -0.029860 |
| 2017-08-29 13:00:00 | -0.118241 | 0.353986 | -0.787655 | -0.021717 |
| 2017-08-29 14:00:00 | -0.214262 | 0.353986 | -0.787655 | 0.008144 |
| 2017-08-29 15:00:00 | -0.249972 | 0.353986 | -0.787655 | -0.108583 |

# Data Examining & Preprocessing

- Reading Data using Pandas – describe, head
- Check for null values
- Line plot for each feature
- Convert Date from String to Date time feature
- Setting the desired frequency
- Handling missing Values
- QQ plots

| time | IOT_Sensor_Reading | Error_Present | Sensor_2 | Sensor_Value |
|---|---|---|---|---|
| 2017-08-29 11:00:00 | -0.015871 | 0.353986 | -0.787655 | 0.008144 |
| 2017-08-29 12:00:00 | -0.101576 | 0.353986 | -0.787655 | -0.029860 |
| 2017-08-29 13:00:00 | -0.118241 | 0.353986 | -0.787655 | -0.021717 |
| 2017-08-29 14:00:00 | -0.214262 | 0.353986 | -0.787655 | 0.008144 |
| 2017-08-29 15:00:00 | -0.249972 | 0.353986 | -0.787655 | -0.108583 |



# Handling Missing Values

- **Forward Filling :** Forward filling means fill missing values with previous data.
- **Backward Filling:** Backward filling means fill missing values with next data point.
- **Mean Filling:** Mean filling means fill missing values with mean of data

```
df_comp.IOT_Sensor_Reading = df_comp.IOT_Sensor_Reading.fillna(method="ffill")
```

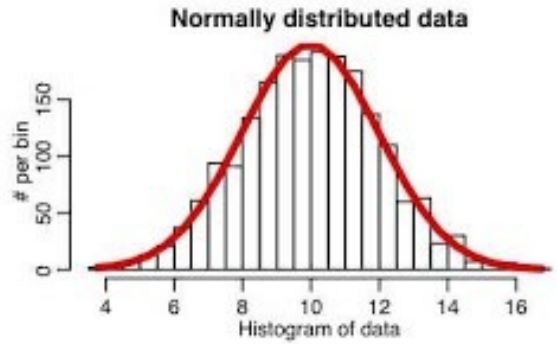
```
df_comp.Error_Present = df_comp.Error_Present.fillna(method="bfill")
```

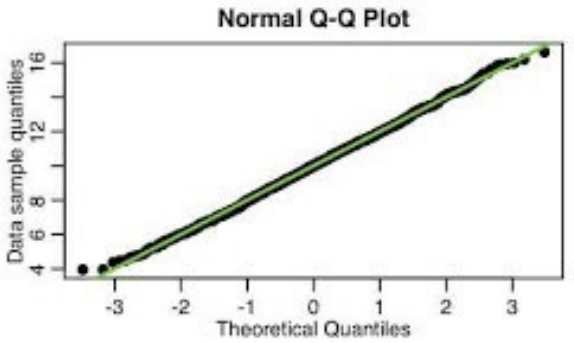
```
df_comp.Sensor_2 = df_comp.Sensor_2.fillna(method="bfill")
```

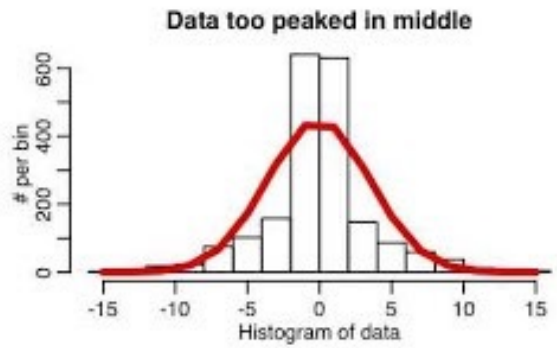
```
df_comp.Sensor_Value = df_comp.Sensor_Value.fillna(value=df_comp.Sensor_Value.mean())
```

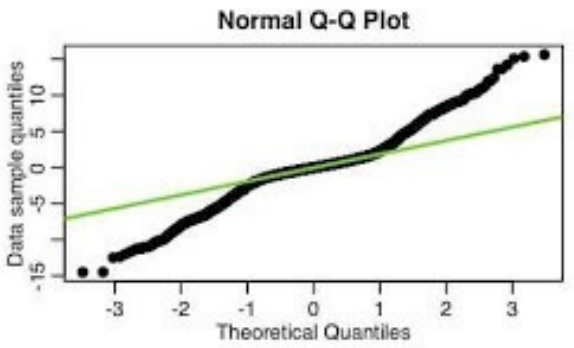
# QQ Plots

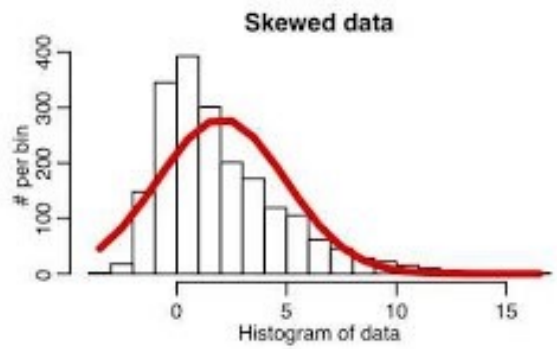
- Q Q Plots (Quantile-Quantile plots) are **plots of two quantiles against each other**.
- The median is a quantile where 50% of the data fall below that point and 50% lie above it. The purpose of Q Q plots is to find out if two sets of data come from the same distribution

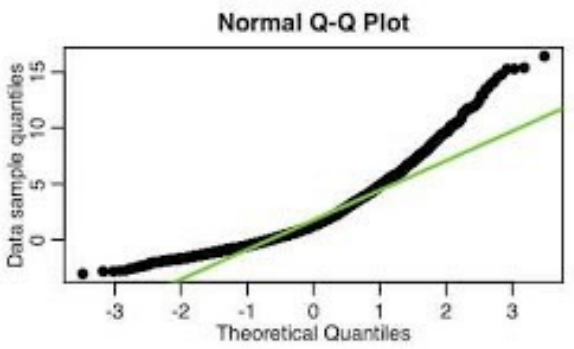


# White Noise

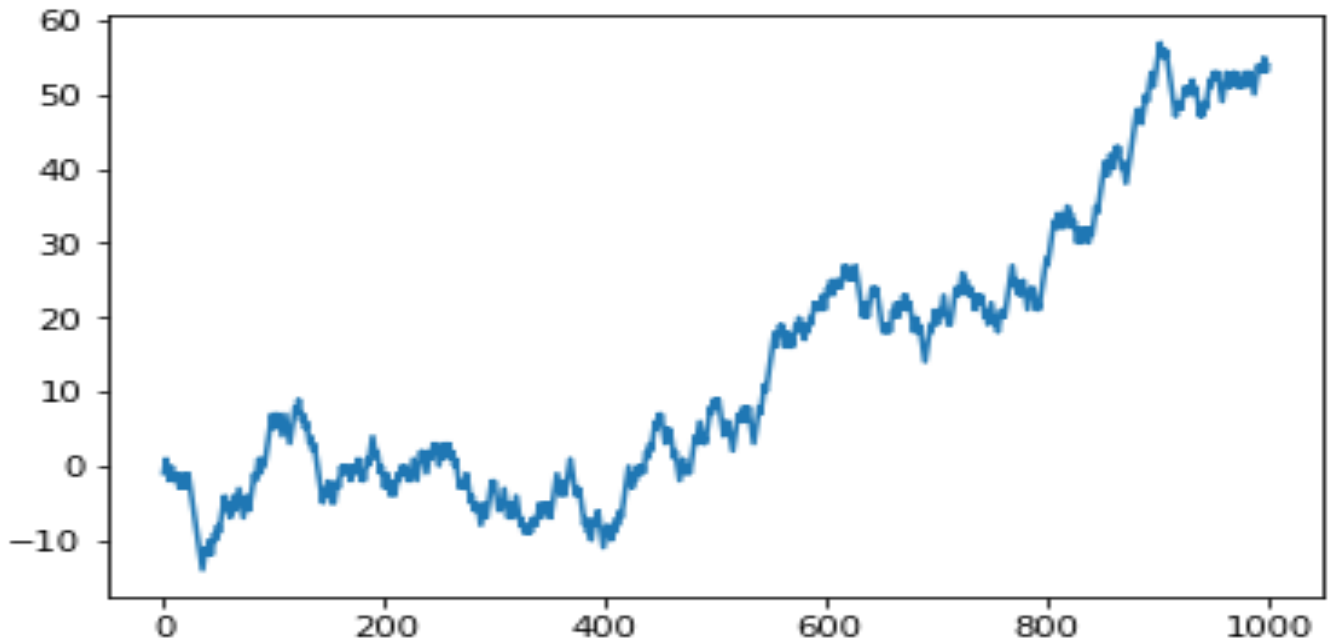
A time series is not white noise if one or more of the following conditions are true:

- Is the mean/level non-zero?
- Does the mean/level change over time?
- Does the variance change over time?
- Do values correlate with lag values?

White noise time series is defined by a **zero mean, constant variance, and zero correlation**. If our time series is white noise, it cannot be predicted/modeled.

## Random Walk

A *random walk* is a time series model $x_t$ such that $x_t = x_{t-1} + w_t$, where $w_t$ is a discrete white noise series.



- A random walk is another time series model where the current observation is equal to the previous observation with a random step up or down.

- Random walk is not Stationary

- If we plot the first-order difference of a time series and the result is white noise, then it is a random walk.

# Stationarity in Time Series

The stationary time-series is the one with :

- no trend
- Fluctuates around the constant mean and has constant variance.
- Covariance between different lags is constant, it doesn't depend on absolute location in time-series.
- Inferences become easy
- Is **strong stationary** if the distribution of a time-series is exactly the same trough time

# How to check for stationarity

- Look at Plots: We can review a time series plot of the data and visually check if there are any obvious trends or seasonality.

- Summary Statistics: We can review the summary statistics for  data for seasons or random partitions and check for obvious or significant differences.

- Statistical Tests: We can use statistical tests to check if the expectations of stationarity are met or have been violated. Two mainly used statistical tests are:
  - Augmented Dickey-Fuller test
  - The Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test

# Augmented Dickey-Fuller test(ADFuller)

- The Augmented Dickey-Fuller test is a type of statistical test called a unit root test.
- The intuition behind a unit root test is that it determines how strongly a time series is defined by a trend.
- In python, the statsmodel package is used for the implementation of AD Fuller and KPSS test.
- The null hypothesis of the test is that the time series can be represented by a unit root, that it is not stationary.
- The alternate hypothesis (rejecting the null hypothesis) is that the time series is stationary.
- Result is interpreted using the p-value from the test.

  - **p-value > 0.05: Fail to reject the null hypothesis (H0), the data has a unit root and is non-stationary.**
  - **p-value <= 0.05: Reject the null hypothesis (H0), the data does not have a unit root and is stationary.**

# KPSS Test

- A statistical test to check for stationarity of a series around a deterministic trend.

- A key difference from ADF test is the null hypothesis of the KPSS test is that the series is stationary.

- The interpretation of p-value is just the opposite to AD Fuller Test.

  - **if p-value is < 0.05, then the series is non-stationary.**
  - **if p-value is > 0.05, then the series is trend stationary**

# Patterns in Time- Series

- Any time series may be split into the following components:

  Base Level + Trend + Seasonality + Error

- A **Trend** is observed when there is an increasing or decreasing slope observed in the time series.
- **Seasonality** is observed when there is a distinct repeated pattern observed between regular intervals due to seasonal factors. It could be because of the month of the year, the day of the month, weekdays or even time of the day.
- A time series can be modeled as an additive or multiplicative, wherein, each observation in the series can be expressed as either a sum or a product of the components:
  - Additive time series: Value = Base Level + Trend + Seasonality + Error
  - Multiplicative Time Series: Value = Base Level x Trend x Seasonality x Error

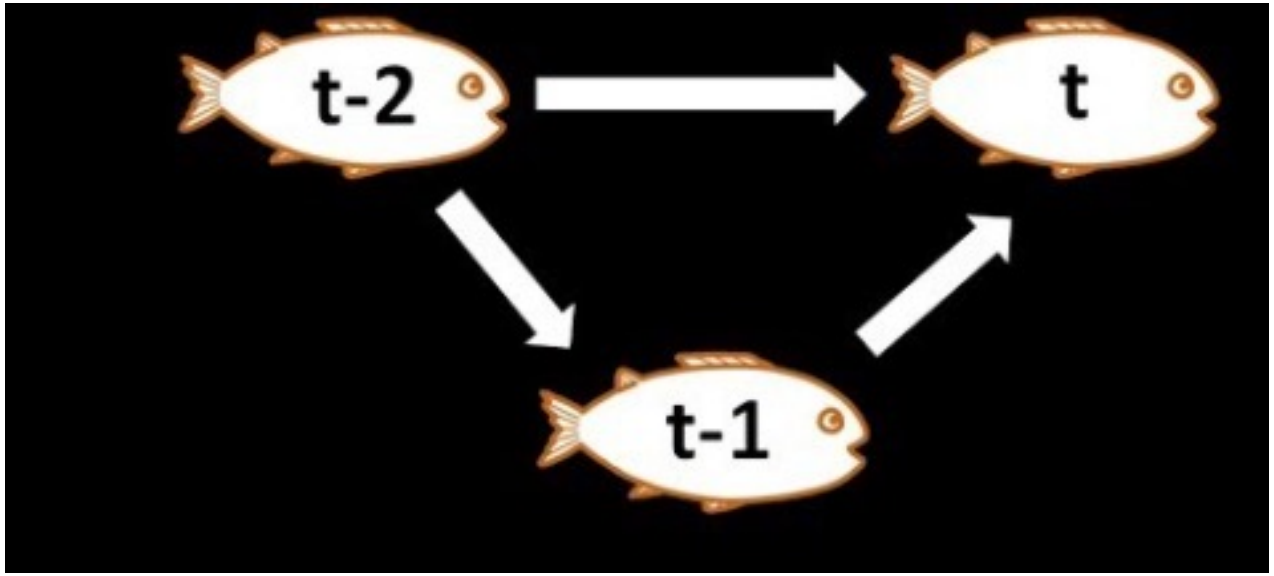
- We can do decomposition of a time series by considering the series as an additive or multiplicative combination of the base level, trend, seasonal index and the residual.

- **The seasonal_decompose in statsmodels** is used to implement this and we can extract the components.

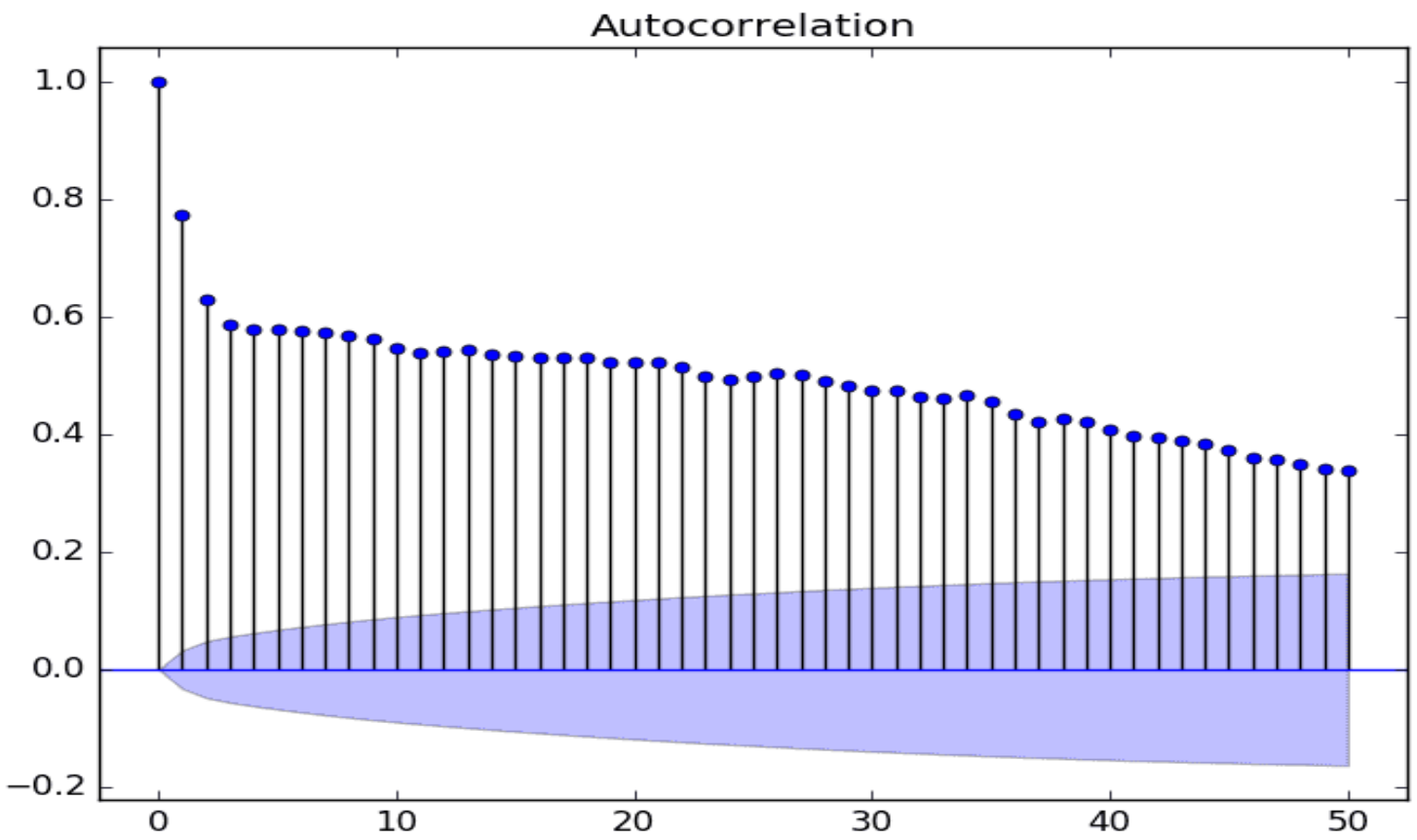
* **We are using Additive Decomposition in our project**

# ACF

- ACF is an (complete) auto-correlation function which gives us values of auto-correlation of any series with its lagged values
- It describes how well the present value of the series is related with its past values.
- A time series can have components like trend, seasonality, cyclic and residual. ACF considers all these components while finding correlations hence it's a 'complete auto-correlation plot'.

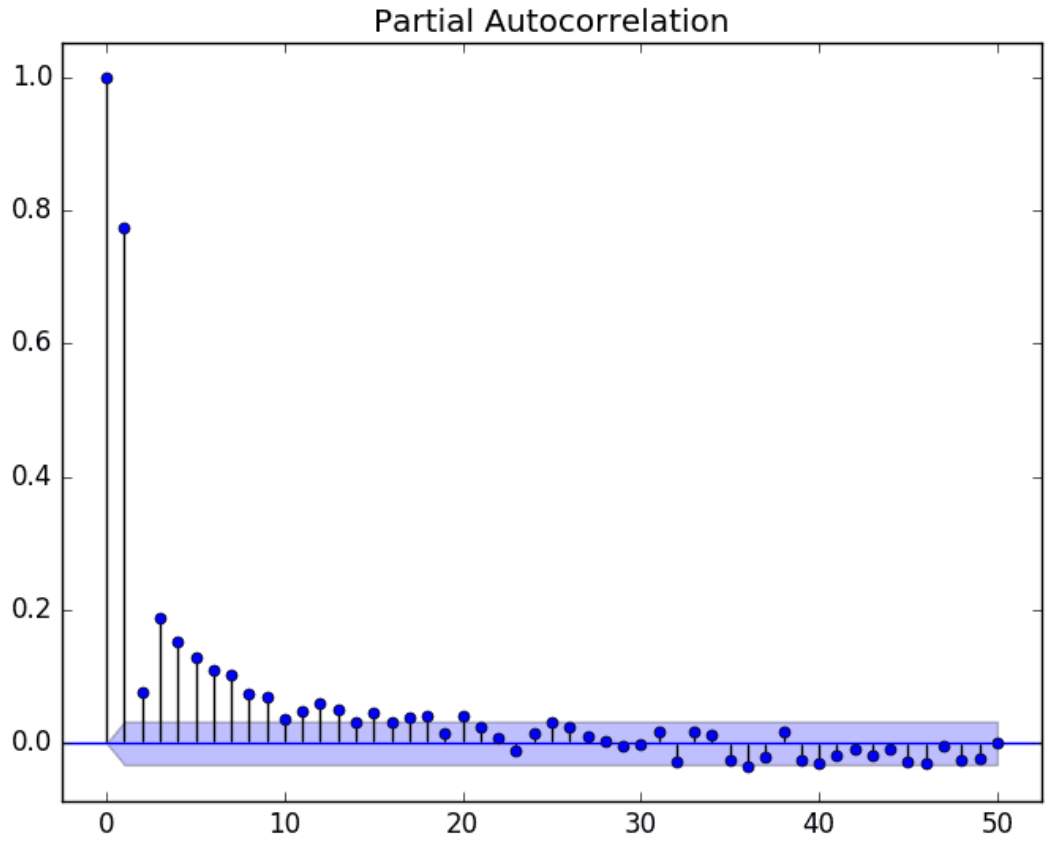


# ACF Plot Example:



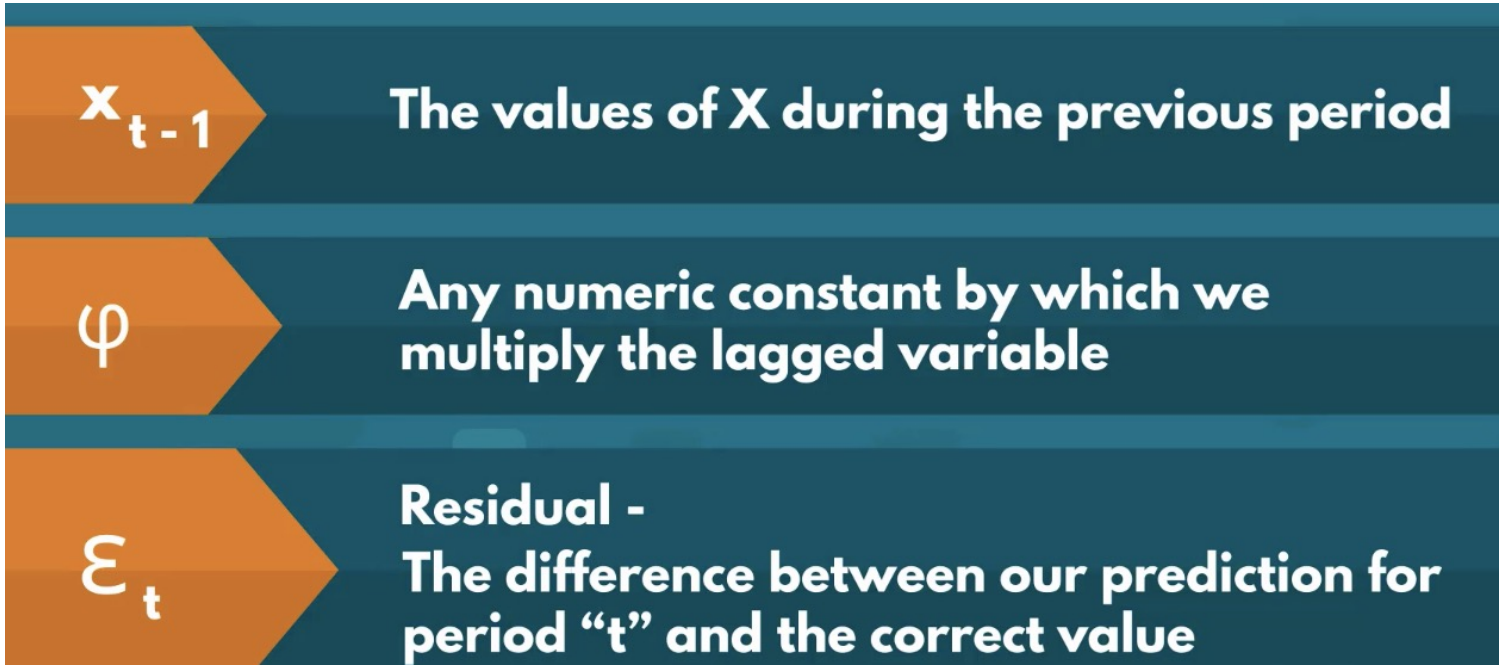
# PACF

- It finds correlation of the residuals ,which remain after removing the effects which are already explained by the earlier lags.



# Autoregression AR Time Series

- By knowing the prices today, we can often make a rough prediction about prices tomorrow.
- The auto regressive model or a model for short, relies on past period values and past periods only to predict current period values.
- It's a linear model where current period values are a sum of past outcomes multiplied by a numeric factor.
- Value of phi lies between -1 and 1



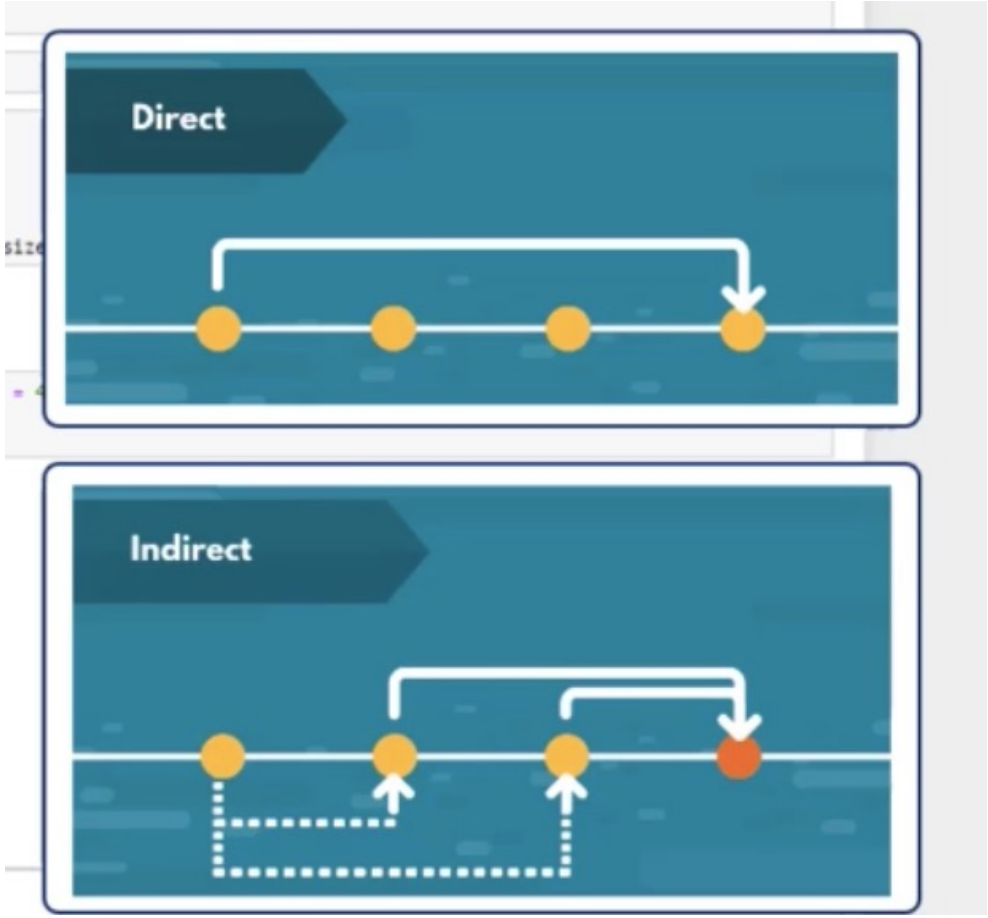

$$x_t = C + \varphi x_{t-1} + \varepsilon_t$$

# How many lags

- Time series about meteorological conditions wouldn't solely rely on the weather statistics a day ago, it's realistic to say it would use data from the last seven days.
- Hence, the model should take into account values up to seven periods back.
- The more lag's we include, the more complicated our model becomes, the more complicated the model, the more coefficients we have to determine and as a result, the more likely it is that some of them would not be significant.
- However, if the coefficients are not significantly different from zero, they would have no effect on the predicted values. So it makes little sense to include them.
- To determine the correct number of lags we should incorporate in our model, we rely on the **autocorrelation** and **partial autocorrelation functions**.

# Examining ACF and PACF

- The ACF captures both direct and indirect effects of previous value has on the current one, since we want to get the most efficient model.
- We only wish to include past lag's which have a direct, significant effect on the present.
- PACF gives the direct impact it has on the current one



# Model Identification from ACF and PACF Plot:

| ACFs | PACFs | Model |
|---|---|---|
| Decay to zero with exponential pattern | Cuts off after lag $p$ | AR($p$) |
| Cuts off after lag $q$ | Decay to zero with exponential pattern | MA($q$) |
| Decay to zero with exponential pattern | Decay to zero with exponential pattern | ARMA($p$, $q$) |

# Rolling Window

- The calculation is also called a “rolling mean” because it's calculating an average of values within a specified range for each row as you go along the DataFrame
- Here we will use it to check if we have constant mean and variance across different sections of the dataset