TSA MAJOR NOTES

Q1: What is Time series analysis? Write its applications.

Ans:

It is a random sequence recorded in a time ordered fashion. Time series analysis is the endeavor of extracting meaningful summary & statistical information from points arranged in temporal order. It is to diagnose past behavior to make predictions about future behavior. The most common concerns of time series analysis are forecasting the future and classifying the past.

1. **Finance and Stock Market Analysis:** Time series analysis is extensively used in finance for forecasting stock prices, currency exchange rates, and commodity prices. It helps in making investment decisions and managing risk.
2. **Economic Forecasting:** Economists use time series analysis to forecast economic indicators such as GDP, inflation rates, and unemployment rates. This data is crucial for policy-making and business planning.
3. **Demand Forecasting:** Businesses use time series analysis to predict future demand for their products or services. This helps in inventory management, production planning, and resource allocation.
4. **Weather Forecasting:** Meteorologists use time series data for weather forecasting. Analyzing historical weather data allows them to make predictions about future weather conditions.
5. **Energy Consumption and Load Forecasting:** Utility companies use time series analysis to predict electricity and energy consumption patterns. This helps them in optimizing energy production and distribution.
6. **Healthcare and Epidemiology:** Time series analysis is used to track and predict the spread of diseases, monitor patient vital signs, and forecast healthcare resource requirements.
7. **Environmental Monitoring:** Time series data is crucial for tracking environmental changes, such as temperature trends, air quality, and water pollution levels.
8. **Quality Control:** Manufacturers use time series analysis to monitor the quality of their products over time. It helps in identifying defects and improving production processes.
9. **Marketing and Sales:** Businesses use time series analysis to analyze sales data, track marketing campaign effectiveness, and forecast sales trends.
10. **Traffic and Transportation:** Time series analysis is applied to traffic data to predict congestion patterns, optimize transportation routes, and plan infrastructure improvements.
11. **Social Sciences:** Time series analysis is used in social sciences to analyze trends in social, economic, and political data. It helps researchers understand and predict human behavior.

Q2: Define trend, seasonality, cyclicity, Irregularity. (factor affecting time series analysis)

Trend is the increase or decrease in the series over a period of time, it persists over a long period of time.

Eg: population growth over the years can be seen as upward trend.

Seasonality is the regular pattern of up and down fluctuations. It is a short term variationoccurring due to seasonal factors.

Eg: sales of icecream increases during summer season

Cyclicity is the mid term variation caused by circumstances, which repeat in irregularintervals

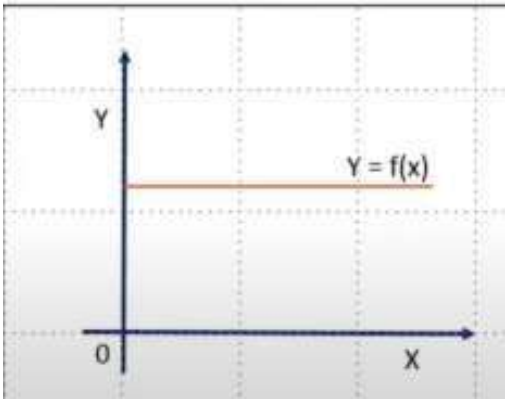
Eg: 5 years of economic growth , followed by 2 years of economic recession, followed by 7years of economic growth followed by 1 year of economic recession.

Irregularity refers to variations which due to predictable factors and also do not repeat inparticular patterns.

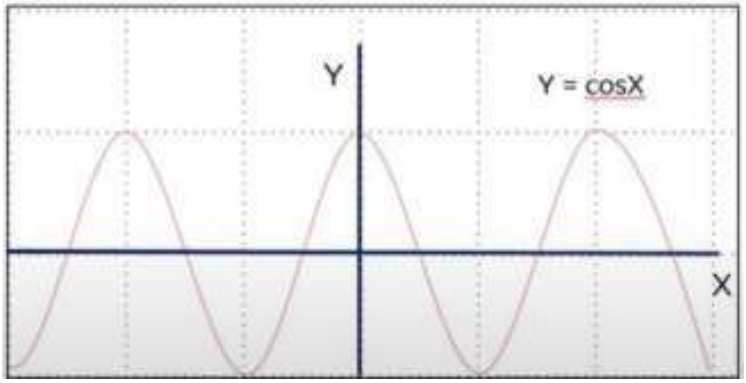
Eg: variations caused by incidents like earthquakes, flood, war,etc.

Q3: write conditions when to not use TSA?

1) When values are constant over a period of time.



2) When values can be represented by known functions like cosx, sinx, etc.



Q4: stationary and non stationary time series.

Ans:

In time series analysis, one of the fundamental concepts is the distinction between stationary and non-stationary time series data. These terms refer to the statistical propertiesand characteristics of the data over time. Understanding whether a time series is stationary or non-stationary is crucial because it has implications for the choice of analytical methods and the reliability of forecasts.

Stationary Time Series:

A time series is considered stationary when its statistical properties do not change over time.In a

stationary time series:

1. **Constant Mean (μ):** The mean (average) of the series remains constant over time.
2. **Constant Variance (σ^2):** The variance (spread or volatility) of the series remains constant over time.
3. **Constant Autocovariance:** The covariance between two observations at different time points (lag) remains constant.
4. **No Seasonal or Trend Components:** Stationary time series do not exhibit any systematic patterns or trends over time. There are no long-term upward or downward movements, and there are no repeating seasonal patterns.
5. **Statistical Tests Confirm Stationarity:** Formal statistical tests, such as the Augmented Dickey-Fuller (ADF) test, can be used to confirm the stationarity of a time series.

Stationary time series are easier to analyze and model because they exhibit stable statistical properties. Many traditional time series forecasting methods assume stationarity.

Non-Stationary Time Series:

Conversely, a time series is considered non-stationary when one or more of the properties mentioned above change over time. Non-stationary time series often exhibit trends, seasonality, and other time-dependent structures. Common characteristics of non-stationary time series include:

1. **Changing Mean:** The mean of the series exhibits a trend or systematic change overtime. It may be increasing, decreasing, or following some other pattern.
2. **Changing Variance:** The variance of the series may change over time, leading to varying levels of volatility.
3. **Seasonal Patterns:** Non-stationary time series can have repeating seasonal patterns or cycles.
4. **Trends:** Non-stationary time series often exhibit long-term trends, which can be upward (growth) or downward (decay).
5. **Unit Roots:** Non-stationary series may exhibit unit roots, which are indicative of non-stationarity. The Augmented Dickey-Fuller (ADF) test is commonly used to test for unit roots.

Non-stationary time series can be challenging to model and forecast directly. Often, it is necessary to transform or differentiate the data to make it stationary before applying traditional time series forecasting techniques.

To summarize, the key difference between stationary and non-stationary time series lies in the stability of their statistical properties over time. Stationary time series exhibit stable means, variances, and covariances, while non-stationary time series display changing patterns and trends. Identifying whether a time series is stationary or non-stationary is a critical step in time series analysis to choose the appropriate modelling and forecasting methods.

Q5: how to convert data into stationary?

Ans:

•**Differencing:** Differencing is a method of removing trends from time series data. This isdone by subtracting the previous value from the current value.

•**Logarithm transformation:** The logarithm transformation is a method of transforming thedata to make it more stationary. This is done by taking the logarithm of the data.

•**Seasonal adjustment**: Seasonal adjustment is a method of removing seasonal patterns from time series data. This is done by estimating the seasonal components of the data andthen subtracting them from the original data.

•**Detrending:** Detrending is a method of removing trends from time series data. This is doneby fitting a trend line to the data and then subtracting the trend line from the original data.

•**Combination of methods:** In some cases, it may be necessary to use a combination ofmethods to make time series data stationary.

Q6: Why retrofitting of time series data is performed from the collection of tables?

Ans:

- **To create a more complete time series**: If the tables contain different but overlapping time periods, then combining them can create a more complete time series.
- **To improve the quality of the time series:** If the tables contain different levels of noise or accuracy, then combining them can improve the overall quality of the time series.
- **To make the time series more accessible:** If the tables are stored in different locations or formats, then combining them can make the time series more accessible to users.

Q7: How to retrofit a time series data from a collection of tables?Ans:

- Step 1: Identify the tables that contain time series data.
- Step 2: Identify the columns in each table that contain the time series data.
- Step 3: Combine the columns from each table into a single column.
- Step 4: Normalize the data in the combined column so that it is all on the same scale.
- Step 5: Save the combined column as a new table.

Q8: How to handle missing data in time series?

Ans:

- **Imputation**

  When we fill in missing data based on observations about the entire data set.

- **Interpolation**

  When we use neighboring data points to estimate the missing value. Interpolation can also be a form of imputation.

- **Deletion of affected time periods**

  When we choose not to use time periods that have missing data at all.

Q9: What is forward fill, backward fill, moving average and Interpolation?

Ans:

Forward Fill (or "ffill"):

Forward fill involves filling missing data points with the most recent preceding valid observation. In other words, it propagates the last known value forward in the dataset untila new valid value is encountered. This method is often used when missing data is assumedto carry forward the last observed value. It's particularly useful in cases where the data represents a sequence or time series, where continuity is important.

Date	Value
2023-01-01	10
2023-01-02	NaN
2023-01-03	15
2023-01-04	NaN
2023-01-05	NaN

After ffill

Date	Value	
2023-01-01	10	
2023-01-02	10	# Filled with the preceding value (10)
2023-01-03	15	
2023-01-04	15	# Filled with the preceding value (15)
2023-01-05	15	# Filled with the preceding value (15)

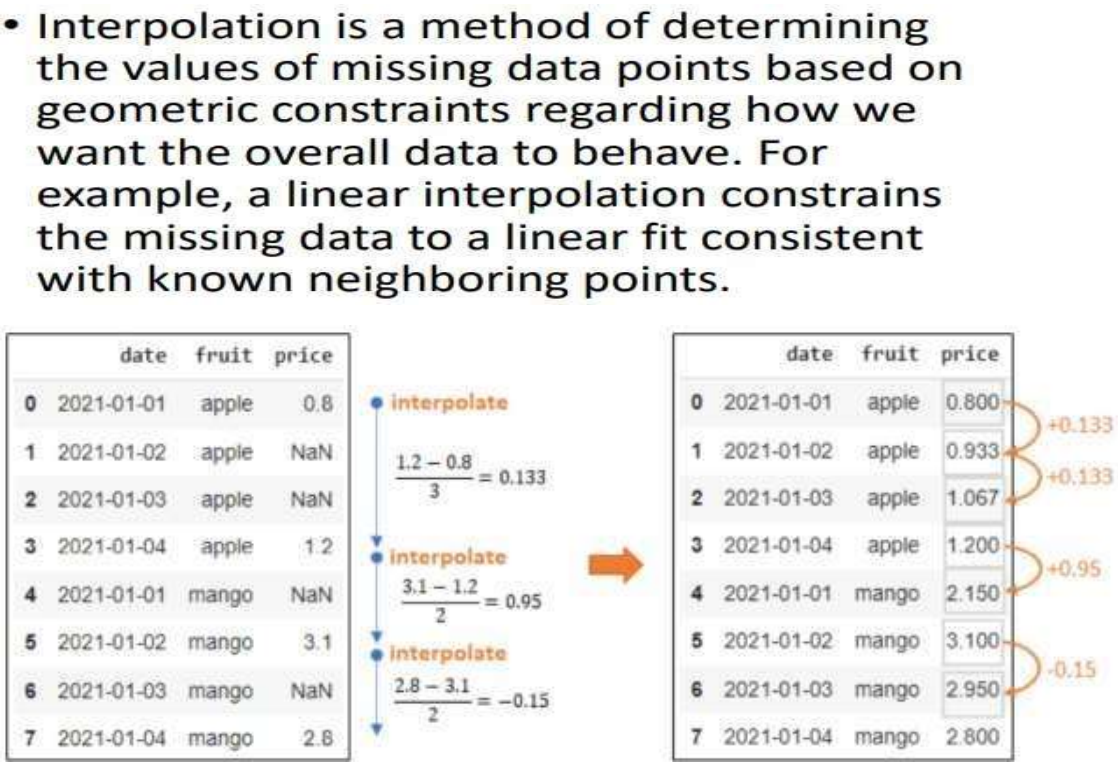
Backward fill is the opposite of forward fill. It involves filling missing data points with thefirst valid observation encountered after the missing data point. In other words, it propagates the next known value backward in the dataset until a new valid value is

encountered.

Using the same example data as above, but applying backward fill:

Date	Value
2023-01-01	10
2023-01-02	15 # Filled with the next valid value (15)
2023-01-03	15
2023-01-04	NaN
2023-01-05	NaN

A **moving average** is a commonly used statistical calculation applied to time series data to analyze and smooth out fluctuations in the data over time. It is particularly useful for identifying trends, patterns, and underlying patterns in noisy or volatile datasets. The moving average is calculated by taking the average of a set of data points within a sliding or"moving" window as it progresses through the dataset.



	date	fruit	price
0	2021-01-01	apple	0.8
1	2021-01-02	apple	NaN
2	2021-01-03	apple	NaN
3	2021-01-04	apple	1.2
4	2021-01-01	mango	NaN
5	2021-01-02	mango	3.1
6	2021-01-03	mango	NaN
7	2021-01-04	mango	2.8

	date	fruit	price
0	2021-01-01	apple	0.800
1	2021-01-02	apple	0.933
2	2021-01-03	apple	1.067
3	2021-01-04	apple	1.200
4	2021-01-01	mango	2.150
5	2021-01-02	mango	3.100
6	2021-01-03	mango	2.950
7	2021-01-04	mango	2.800

Q10: What is downsampling and upsampling? When to perform it?

Downsampling is subsetting data such that the timestamps occur at a lower frequency than in the original time series.This is most often done in the following cases.

- The original resolution of the data isn't sensible.
- Focus on a particular portion of a seasonal cycle.
- Match against data at a lower frequency.

Upsampling is representing data as if it were collected more frequently than was actually the case.

- Irregular time series.
- Inputs sampled at different frequencies.
- Knowledge of time series dynamics

Q11: what is smoothening of data? Why it is required? What are exponential smoothening methods and when to perform exponential smoothening?

Ans:

Smoothing of data refers to the process of reducing noise or random variations in a dataset to extract underlying trends, patterns, or signals. It involves applying mathematical or statistical techniques to create a smoother representation of the data by averaging or aggregating values over a certain period or window. Smoothing is required for several reasons:

1. **Noise Reduction:** Many real-world datasets contain random fluctuations or noise that can make it difficult to identify meaningful patterns or trends. Smoothing helpsto reduce the impact of this noise.
2. **Pattern Detection:** Smoothing can reveal hidden patterns or trends in the data that may not be immediately apparent in the raw data.
3. **Data Visualization:** Smoother data is often easier to visualize and interpret, making itvaluable for data exploration and communication.
4. **Forecasting:** Smoothing is often used as a preprocessing step for time seriesforecasting, as it can help create a more stable and predictable dataset.

One common method of data smoothing is exponential smoothing. Exponential smoothingis a time series forecasting technique that assigns exponentially decreasing weights to past observations, giving more weight to recent observations.

When to use exponential smoothening:

- When you are forecasting for a large number of items.
- The forecasting horizon is relatively short.
- There is little outside information available about cause and effect.
- Small effort in forecasting is desired. Efforts is measured by both a method's ease of application and by the computational requirements.
- Updating of the forecast as new data becomes available is easy.
- It is desired that the forecast is adjusted for randomness and tracks trends and seasonality.

New Base = Previous Bases + α(New Demand- Previous Base)

$S_t = S_{t-1} + \alpha(D_t - S_{t-1})$

't' is Current period

α = Smoothing Constant 0 to 1

$F_{t,1} = S_t$

- The smoothing constant, α, must be between 0.0 and 1.0. Popular values .1 to .3
- A large α provides a high impulse response forecast. $\alpha = 0$, $\alpha = 1$
- A small α provides a low impulse response forecast.

Q12: Numerical moving average and exponential smoothening

- Moving Average

Use the moving average method with an AP = 3 days to develop a forecast of the call volume in Day 13.

$F_{13} = (168 + 198 + 159)/3 = 175.0$ calls

.6
.3
.1
$\Sigma W = 1.0$

- Weighted Moving Average

Use the weighted moving average method with an AP = 3 days and weights of .1 (for oldest datum), .3, and .6 to develop a forecast of the call volume in Day 13.

$F_{13} = .6\,D_{12} + .3\,D_{11} + .1\,D_{10}$

D_{10} D_{11} D_{12}

$F_{13} = .1(168) + .3(198) + .6(159) = 171.6$ calls ✓ ≈ 172 Calls.

$\Sigma W = 1$

Note: The WMA forecast is lower than the MA forecast because Day 13's relatively low call volume carries almost twice as much weight in the WMA (.60) as it does in the MA (.33).

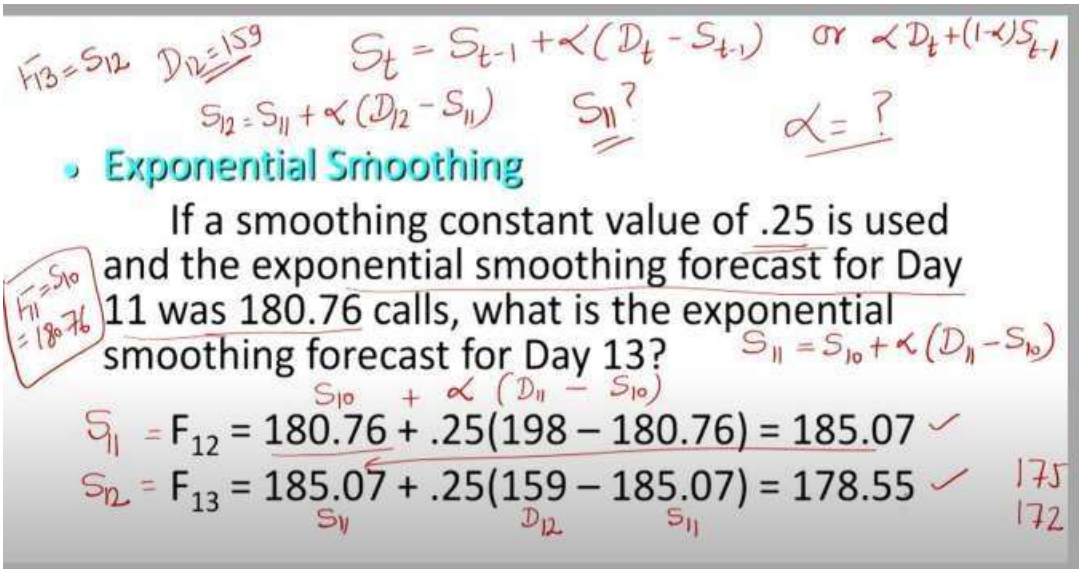
$\alpha = ?$

- Exponential Smoothing

If a smoothing constant value of .25 is used and the exponential smoothing forecast for Day 11 was 180.76 calls, what is the exponential smoothing forecast for Day 13?

$F_{12} = 180.76 + .25(198 - 180.76) = 185.07$

$F_{13} = 185.07 + .25(159 - 185.07) = 178.55$



Q13: what is seasonality and why it is removed?

Seasonality is the repeated variation of a time series over a fixed period of time. For example, sales of ice cream may be higher in the summer and lower in the winter. **Seasonality can be removed from time series data for several reasons:**

- **To make the data easier to analyze**. Many statistical and machine learning algorithms are designed to work with stationary data. Seasonality can make the data non-stationary, which can make it more difficult to analyze.
- **To improve the accuracy of forecasts**. Forecasts of time series data are typically more accurate when the data is stationary. This is because seasonality can introduce noise into the data, which can make it more difficult to predict future values.
- **To make the data more comparable across different time periods.** Seasonality can make it difficult to compare data from different time periods. For example, if you are comparing sales data from different years, you may want to remove seasonality so that you can compare the underlying trends.

Q14: Lookahead bias

- Lookahead bias -occur in time series forecasting when the model is trained on data that includes future values.
- Preventing lookahead bias is important because it can lead to inaccurate forecasts. When a model is trained on data that includes future values, it is essentially learning to predict the future based on the past. This can lead to the model overfitting the data, and making predictions that are not accurate.

Why is it required?

- Lookahead bias is a serious problem because it can lead to inaccurate forecasts. This can have a significant impact on businesses and organizations that rely on time series forecasting. For example, a business that uses time series forecasting to set inventory levels could make incorrect decisions if the forecasts are inaccurate.

Ways to prevent lookaheads

- Use a sliding window. A sliding window only includes data that has already occurred, and it is updated as new data becomes available. This prevents the model from seeing future values.
- Use a causal model. A causal model is a model that only uses data that is causally related to the target variable. This means that the model cannot see future values, because they are not causally related to the target variable.

Q15: Explain Additive model and Multiplicative model?

ANS:

Additive Model

These models assume the observed time series is the sum of its elements:

$$y(t) = Trend + Seasonality + Residual$$

This model considers that the magnitudes of the seasonal and residual elements are independent of the trend.

Multiplicative Model

These models assume the observed time series is the product of its elements:

$$y(t) = Trend \times Seasonality \times Residual$$

This model implies that the seasonality and residuals are dependent on the trend.

This model can be transformed into an additive model by applying logarithmic transformations:

$$log(y(t)) = log(Trend) + log(Seasonality) + log(Residual)$$

Additive and multiplicative models are two fundamental approaches in time series analysis for decomposing a time series into its constituent components. These components help in understanding and modeling the underlying patterns and variations in the data. The primary difference between the two models is how they represent the relationship between the various components.

Q16: What is trend? How to remove trend?

ANS:

The trend represents an increase or decrease in time-series value over time. If we notice that the value of measurement over time is increasing or decreasing, then we can say that it has an upward or downward trend.

How to remove trend from time-series data?

• **Log Transformation**: To apply log transformation, we need to take a log of each individual value of time-series data.

•**Power Transformation:** Apply power transformation in data same way as that of log transformation to remove trend.

• **Local smoothing** - Applying moving window functions to time-series data: We can calculate rolling mean over a period of 12 months and subtract it from original time-series to get de-trended time-series.

• **Differencing** a time-series.

• **Linear Regression:** We can also apply a linear regression model to remove the trend. Below we are fitting a linear regression model to our time-series data. We are then using a fit model to predict time-series values from beginning to end. We are then subtracting predicted values from original time-series to remove the trend.

Q17: What is Seasonality? How to remove seasonality?

ANS:

The seasonality represents variations in measured value which repeats over the same time interval regularly. If we notice that variations in value are happening every week, month, quarter or half-yearly then we can say that time series has some kind of seasonality.

How to remove seasonality from time-series data?

• **Average de-trended values.**

Average de-trended values is a method for removing seasonality from time series data. It involves the following steps:

• Detrend the time series: This means removing the trend from the data. The trend can be removed using a variety of methods, such as linear regression or polynomial regression.

• Calculate the average of the de-trended values: This gives the average value of the data after the trend has been removed.

• Subtract the average value from the de-trended values: This leaves the seasonal component of the data.

• **Differencing a time series.**

Apply differencing to log-transformed time-series by shifting its value by 1 period and subtracting it from original log-transformed time-series.

Q18: correlation coefficient?

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

r = correlation coefficient
x_i = values of the x-variable in a sample
$\bar{x}$ = mean of the values of the x-variable
y_i = values of the y-variable in a sample
$\bar{y}$ = mean of the values of the y-variable

Subject	Age x	Glucose Level y
1	43	99
2	21	65
3	25	79
4	42	75
5	57	87
6	59	81

- **Question**: Find the value of the correlation coefficient from the following table:

Autocorrelation

The correlation of a series with its own lagged values is called ***autocorrelation*** or ***serial correlation***.

- The first autocorrelation of Y_t is $\text{corr}(Y_t, Y_{t-1})$
- The first ***autocovariance*** of Y_t is $\text{cov}(Y_t, Y_{t-1})$
- Thus

$$\text{corr}(Y_t, Y_{t-1}) = \frac{\text{cov}(Y_t, Y_{t-1})}{\sqrt{\text{var}(Y_t)\,\text{var}(Y_{t-1})}} = \rho_1$$

Autocorrelation (Serial Correlation) and Autocovariance

The j^{th} autocovariance of a series Y_t is the covariance between Y_t and its j^{th} lag, Y_{t-j}, and the j^{th} autocorrelation coefficient is the correlation between Y_t and Y_{t-j}. That is,

$$j^{\text{th}} \text{ autocovariance} = \text{cov}(Y_t, Y_{t-j})$$

$$j^{\text{th}} \text{ autocorrelation} = \rho_j = \text{corr}(Y_t, Y_{t-j}) = \frac{\text{cov}(Y_t, Y_{t-j})}{\sqrt{\text{var}(Y_t)\text{var}(Y_{t-j})}}.$$

The j^{th} autocorrelation coefficient is sometimes called the j^{th} serial correlation coefficient.

Q19: Stationary series and why non-stationary time series data is difficult to analyze?

Ans:

A Stationary series is one whose statistical properties such as mean, variance, covariance, and standard deviation do not vary with time, or these stats properties are not a function of time. In other words, stationarity in Time Series also means series without a Trend or Seasonal components.

Types of Stationary Series:

1. Strict Stationary – Satisfies the mathematical definition of a stationary process. Mean, variance & covariance are not a function of time.

2. Seasonal Stationary – Series exhibiting seasonality.

3. Trend Stationary – Series exhibiting trend.

• Note: Once the seasonality and trend are removed, the series will be strictly stationary difficult to analyze:

• Non-stationary data can be more sensitive to noise. This means that small changes in the data can have a large impact on the analysis.

• Non-stationary data can be more difficult to model. This is because the statistical properties of the data are changing over time.

• Non-stationary data can be more difficult to forecast. This is because the future values of the data are not likely to be constant.

Q20: Augmented Dickey-Fuller (ADF) Test.

Ans:

ADF test belongs to a category of tests called 'Unit Root Test', which is the proper method for testing the stationarity of a time series.

• In probability theory and statistics, a unit root is a feature of some stochastic processes (such as random walks) that can cause problems in statistical inference involving time series models. In simple terms, the unit root is non-stationary but does not always have a trend component.

The Augmented Dickey-Fuller (ADF) test is a statistical hypothesis test used to determine whether a unit root is present in a time series dataset. Developed by David Dickey and Wayne Fuller, this test is widely employed in econometrics and time series analysis to assess the stationarity of a given dataset.

The concept of stationarity is crucial in time series analysis, as it implies that the statistical properties of a dataset, such as mean, variance, and autocorrelation, remain constant over time. A series with a unit root (non-stationary) tends to exhibit trends or irregularities that make predictions challenging.

The ADF test evaluates the null hypothesis that a unit root is present in a time series dataset against the alternative hypothesis of stationarity. If the test statistics calculated from the ADF test are less than the critical values at certain confidence levels, then the null hypothesis of a unit root is rejected, indicating that the series is stationary. Conversely, if the test statistics exceed the critical values, the null hypothesis of a unit root cannot be rejected, suggesting the series is non-stationary.

The ADF test considers different model specifications, including the presence of a constant term and/or a trend in the time series equation, to account for potential variations in the data structure.

Understanding the stationarity of a time series is crucial for various applications, including forecasting, economic modeling, and financial analysis. The ADF test serves as a valuable tool in assessing and confirming the stationarity of a dataset, aiding analysts and researchers in making informed decisions based on reliable time series data.

A unit root is a characteristic of a time series variable whose value at each time point is influenced by its previous value, leading to a non-constant variance over time. In other words, a time series with a unit root exhibits a stochastic trend, and its statistical properties do not revert to a constant mean or trend over time.

The Augmented Dickey-Fuller (ADF) test is a statistical test used to assess the presence of a unit root in a time series. The null hypothesis of the ADF test is that a unit root is present, indicating that the time series is non-stationary. The alternative hypothesis is that there is no unit root, suggesting that the time series is stationary.

The ADF test is based on the Dickey-Fuller test, which was extended to include lagged differences of the variable. The augmented version accommodates autocorrelation and serial correlation in the time series data.

Steps of the ADF Test:

1. **Null Hypothesis (H_0):**
    - The presence of a unit root, indicating the time series is non-stationary.
2. **Alternative Hypothesis (H_1):**
    - No unit root, suggesting the time series is stationary.
3. **Test Statistic:**
    - The ADF test statistic is compared to critical values from tables to determine whether to reject the null hypothesis.
4. **Decision Rule:**
    - If the test statistic is less than the critical value, the null hypothesis is rejected, indicating that the time series is likely stationary. If the test statistic is greater than the critical value, the null hypothesis is not rejected, suggesting the presence of a unit root and non-stationarity.

The ADF test is commonly used in econometrics and time series analysis to assess the stationarity of a time series. Stationary time series are easier to model and analyze, and many time series analysis techniques assume stationarity. If a unit root is detected, differencing the time series may be applied to make it stationary and more amenable to analysis.

Q21. Difference b/w self and spurious correlation

Aspect	**Self-correlation**	**Spurious Correlation**
Definition	Correlation of a variable with itself	Correlation between two variables with a potential third variable influencing both

Aspect	Self-correlation	Spurious Correlation
Example	Time series analysis of monthly sales data	Ice cream sales vs. drowning incidents (influenced by temperature)
Focus	Temporal dependencies within a variable	Relationship between two variables, considering potential confounding factors
Context	Time series, repeated measures	Observational data, understanding limitations of correlation analysis
Causation	Doesn't imply causation	Emphasizes caution in inferring causation from correlation, especially with lurking variables
Application	Common in time series studies	Relevant in broader observational data analyses, correlation studies

Aspect	Self-Correlation	Spurious Correlation
Definition	Correlation of a time series with its lagged version	Correlation between two unrelated variables that appear to be related by chance
Calculation Method	Involves computing correlation between a series and its lagged values	Involves calculating correlation between unrelated variables
Relationship	Focuses on exploring the relationship and dependency within a single time series	Addresses the apparent correlation between two unrelated variables
Interpretation	Indicates the presence of patterns, seasonality, or dependencies within the time series	Indicates a misleading correlation that arises due to coincidence or external factors
Example	Correlation between a stock price today and the stock price from yesterday	Correlation between unrelated events, like ice cream sales and shark attacks, which might show a seemingly high correlation during summer months but have no causal relationship

Q22. Give the difference b/w linear interpolation and polynomial interpolation in table

Aspect	Linear Interpolation	Polynomial Interpolation
Degree of Polynomial	First-degree polynomial (linear)	Higher-degree polynomial (quadratic, cubic, etc.)

Aspect	Linear Interpolation	Polynomial Interpolation
Interpolating Function	Connects two adjacent data points with a line	Fits a polynomial curve through multiple data points
Flexibility	Simple and computationally less intensive	More flexible but may lead to overfitting
Accuracy	Less accurate, especially for non-linear data	Can achieve higher accuracy with the right degree
Smoothness	Produces piecewise linear segments	Can result in smoother curves or oscillations depending on degree
Computational Complexity	Lower computational cost	Higher computational cost, especially with higher degrees
Ease of Implementation	Easy to implement	More complex to implement, especially with higher degrees
Risk of Overfitting	Lower risk of overfitting	Higher risk of overfitting, especially with high-degree polynomials
Use Cases	Simple interpolations, quick approximations	When a more flexible and accurate fit is required

Q23. Why linear regression cannot always be used for time series forecasting?

Ans

- Linear regression assumes that the relationship between the independent and dependent variables is linear. However, time series data often exhibits non-linear relationships. For example, the demand for a product may increase exponentially over time. In this case, linear regression would not be able to accurately forecast the demand.
- Linear regression assumes that the errors are independent and identically distributed (iid). However, time series data often exhibits autocorrelation, which means that the errors are not independent. This can make it difficult for linear regression to accurately forecast the data.
- Linear regression is not able to capture the seasonality of time series data. Seasonality refers to the regular patterns that occur in time series data. For example, the demand for electricity may increase in the summer and decrease in the winter. Linear regression is not able to capture these patterns, which can lead to inaccurate forecasts.

In general, linear regression is a good choice for forecasting time series data that exhibits a linear relationship and does not have a lot of seasonality. However, if the data does not meet these conditions, then linear regression may not be the best choice.

Q24. What are Statistical Models for time series?

Ans

Statistical models for time series are mathematical representations that describe the temporal structure and dependencies present in a sequence of data points observed over time. These models help analyze and forecast future values based on historical observations.

Linear statistical models for time series are related to linear regression but account for the correlations that arise between data points in the same time series.

Some common statistical models for time series include:

1. **Autoregressive (AR) Model:**
   - **Description:** The autoregressive model represents a time series as a linear combination of its own past values. The term "autoregressive" indicates that the variable is regressed on its own previous values.
   - **Equation:** $Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \ldots + \beta_p Y_{t-p} + \epsilon_t$
   - **Order:** denoted by p, representing the number of past observations considered.
2. **Moving Average (MA) Model:**
   - **Description:** The moving average model represents a time series as a linear combination of past error terms (residuals). It captures short-term, random fluctuations in the data.
   - **Equation:** $Y_t = \mu + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q}$
   - **Order:** denoted by q, representing the number of past error terms considered.
3. **Autoregressive Integrated Moving Average (ARIMA) Model:**
   - **Description:** ARIMA combines autoregressive and moving average components with differencing to make a time series stationary. It's expressed as ARIMA(p, d, q), where p is the autoregressive order, d is the differencing order, and q is the moving average order.
   - **Equation:** $Y'_t = c + \phi_1 Y'_{t-1} + \phi_2 Y'_{t-2} + \ldots + \phi_p Y'_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q}$
4. **Vector Autoregression (VAR) Model:**
   - **Description:** VAR is a multivariate extension of the autoregressive model. It models a system of multiple time series variables as linear combinations of their own past values and the past values of other variables in the system.
   - **Equation:** $Y_t = A_1 Y_{t-1} + A_2 Y_{t-2} + \ldots + A_p Y_{t-p} + \epsilon_t$
   - **Order:** denoted by p, representing the number of past observations considered.

Q25. Difference b/w linear regression and linear statistical models?

Aspect	Linear Regression	Linear Statistical Models
Definition	A specific type of statistical model	A broader class of models including linear regression
Objective	Modeling the relationship between a dependent variable and one or more independent variables	Making statistical inferences, hypothesis testing, and parameter estimation
Scope	Specifically focuses on predicting a dependent variable	Encompasses various techniques such as ANOVA, ANCOVA, linear mixed-effects models, etc.
Equations	Equation expresses a linear relationship between variables (e.g., $y = mx + b$)	General form involves a linear combination of parameters and variables, often with an error term
Application	Used for prediction and understanding the relationship between variables	Used for hypothesis testing, parameter estimation, and making inferences
Examples	Simple linear regression, multiple linear regression	ANOVA, ANCOVA, linear mixed-effects models, multiple regression with interactions, etc.

Q26. Give difference between auto regressive and linear regression.

Aspect	Linear Regression	Autoregressive Model
Type of Model	General regression model for predicting a dependent variable based on one or more independent variables	Time series model specifically designed for modeling the relationship between a variable and its own past values
Nature of Variables	Typically used for cross-sectional data or data where observations are independent	Specifically designed for time series data where observations are dependent on past observations
Objective	Predict the value of a dependent variable based on the values of independent variables	Model the relationship between a variable and its own past values to capture temporal dependencies
Time Dependency	Assumes independence of observations	Explicitly models temporal dependencies and considers the order of observations
Equation Form	The regression equation is typically in the form $y = mx + b$, where y is the dependent variable, x is the independent variable, m is the slope, and b is the intercept	The autoregressive model is expressed as $Y_t = \beta_0 + \beta_1 Y_{t-1} + \epsilon_t$, where Y_t is the variable at time t, Y_{t-1} is the variable at the previous time step, β_0 and β_1 are coefficients, and ϵ_t is the error term
Application	Widely used in various fields for predictive modeling, causal inference, and understanding relationships between variables	Specifically used for modeling time series data in fields such as finance, economics, and signal processing
Assumption of Independence	Assumes independence of observations	Assumes that observations are dependent on their past values
Data Type	Can be applied to cross-sectional and time series data, but typically used for the former	Specifically designed for time series data

Q27. Auto regressive models

- The autoregressive (AR) model relies on the intuition that the past predicts the future and so a time series process in which the value at a point in time t is a function of the series values at earlier points in time.

The simplest AR model, an AR(1) model, describes a system as follows:

$$y_t = b_0 + b_1 \times y_{t-1} + e_t$$

The value of the series at time t is a function of a constant b_0, its value at the previous time step multiplied by another constant $b_1 \times y_{t-1}$ and an error term that also varies with time e_t. This error term is assumed to have a constant variance and a mean of 0. We denote an autoregressive term that looks back only to the immediately prior time as an AR(1) model because it includes a lookback of one lag.

Q27. How autoregressive models are remarkably flexible?

Autoregressive models are remarkably flexible at handling a wide range of different time series patterns. The two series in Figure show series from an AR(1) model and an AR(2) model. Changing the parameters $\phi_1, \ldots, \phi_p$ results in different time series patterns. The variance of the error term ε_t will only change the scale of the series, not the patterns.

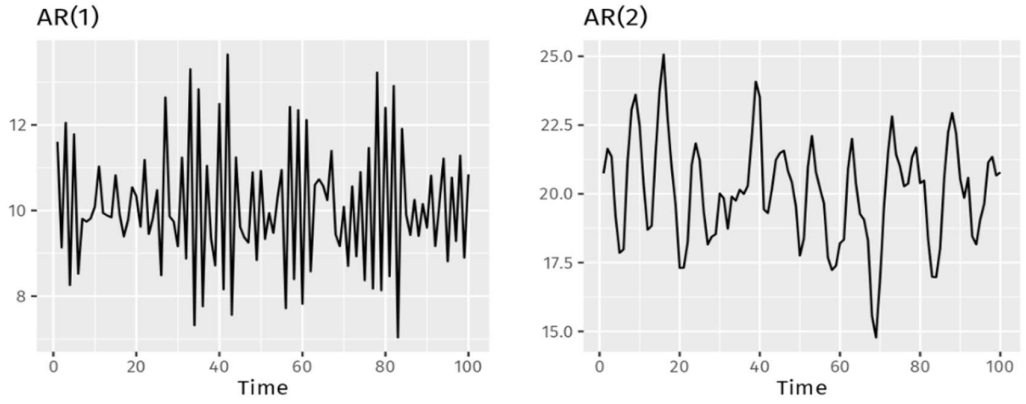


Figure : Two examples of data from autoregressive models with different parameters. Left: AR(1) with $y_t = 18 - 0.8y_{t-1} + \varepsilon_t$. Right: AR(2) with $y_t = 8 + 1.3y_{t-1} - 0.7y_{t-2} + \varepsilon_t$. In both cases, ε_t is normally distributed white noise with mean zero and variance one.

For an AR(1) model:

- when $\phi_1 = 0$, y_t is equivalent to white noise;
- when $\phi_1 = 1$ and $c = 0$, y_t is equivalent to a random walk;
- when $\phi_1 = 1$ and $c \neq 0$, y_t is equivalent to a random walk with drift;
- when $\phi_1 < 0$, y_t tends to oscillate around the mean.

We normally restrict autoregressive models to stationary data, in which case some constraints on the values of the parameters are required.

- For an AR(1) model: $-1 < \phi_1 < 1$.
- For an AR(2) model: $-1 < \phi_2 < 1, \phi_1 + \phi_2 < 1, \phi_2 - \phi_1 < 1$.

Q28. How to pick the Auto Regressive Model?

How to Pick the Auto-Regressive Model?

PACF Plot

- PACF is the partial autocorrelation function that explains the partial correlation between the series and lags of itself.
- In simple terms, PACF can be explained using a linear regression where we predict y(t) from y(t-1), y(t-2), ...

How to Pick the Auto-Regressive Model?

YEAR	PROFIT
2010	76
2011	80
2012	90
2013	88
2014	81
2015	82
2016	94
2017	95
2018	93
2019	84
2020	82

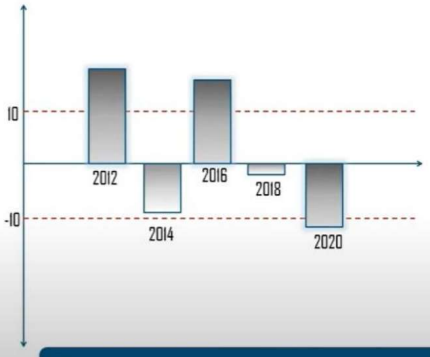
$t = 2$

$y_t - y_{t-2}$

YEAR	LAG
2010	0
2012	14
2014	-9
2016	13
2018	-1
2020	-11

How to Pick the Auto-Regressive Model?

YEAR	LAG
2010	0
2012	14
2014	-9
2016	13
2018	-1
2020	-11



$$F_{2021} = c + \varphi y_{2012} + \varphi y_{2016} + \varphi y_{2020} + \varepsilon t$$

Q29. Moving Average

- A moving average (MA) model relies on a picture of a process in which the value at each point in time is a function of the recent past value "error" terms, each of which is independent from the others. We will review this model in the same series of steps we used to study AR models.

The model

A moving average model can be expressed similarly to an autoregressive model except that the terms included in the linear equation refer to present and past error terms rather than present and past values of the process itself. So an MA model of order *q* is expressed as:

$$y_t = \mu + e_t + \theta_1 \times e_{t-1} + \theta_2 \times e_{t-2} \ldots + \theta_q \times e_{t-q}$$

Q30. Why are moving average models called weakly stationary?

MA models are by definition weakly stationary without the need to impose any constraints on their parameters. This is because the mean and variance of an MA process are both finite and invariant with time because the error terms are assumed to be iid with mean 0. We can see this as:

$$E(y_t = \mu + e_t + \theta_1 \times e_{t-1} + \theta_2 \times e_{t-2} \ldots + \theta_q \times e_{t-q})$$
$$= E(\mu) + \theta_1 \times 0 + \theta_2 \times 0 + \ldots = \mu$$

For calculating the variance of the process, we use the fact that the e_t terms are iid and also the general statistical property that the variance of the sum of two random variables is the same as their individual variances plus two times their covariance. For iid variables the covariance is 0. This yields the expression:

$$Var(y_t) = (1 + \theta_1^2 + \theta_2^2 + \ldots. + \theta_q^2) \times \sigma_e^2$$

So both the mean and variance of an MA process are constant with time regardless of the parameter values.

Q31. What are the properties of the moving average model?

Property 1: The mean of an MA(q) process is μ.

Proof:

$$E[y_i] = \mu + E[\varepsilon_i] + \theta_1 E[\varepsilon_{i-1}] + \cdots + \theta_q E[\varepsilon_{i-q}] = \mu + 0 + \theta_1 \cdot 0 + \cdots + \theta_q \cdot 0 = \mu$$

Property 2: The variance of an MA(q) process is

$$var(y_i) = \sigma^2(1 + \theta_1^2 + \cdots + \theta_q^2)$$

Proof:

$$var(y_i) = 0 + var(\varepsilon_i) + \theta_1^2 var(\varepsilon_{i-1}) + \cdots + \theta_q^2 var(\varepsilon_{i-k}) = \sigma^2 + \theta_1^2\sigma^2 + \cdots + \theta_q^2\sigma^2$$
$$= \sigma^2(1 + \theta_1^2 + \cdots + \theta_q^2)$$

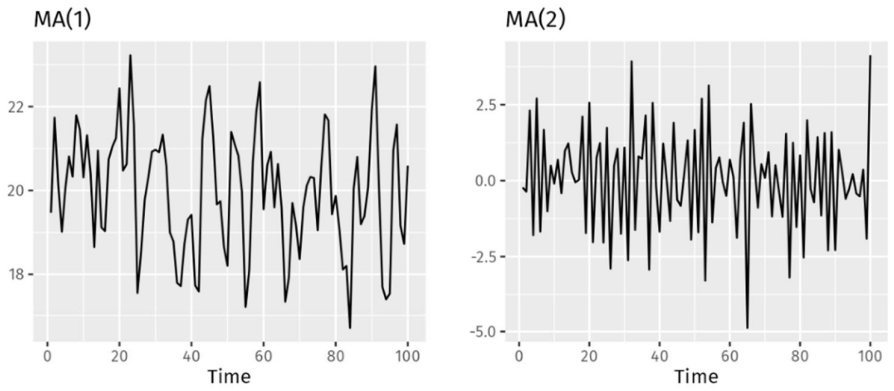


Figure Two examples of data from moving average models with different parameters. Left: MA(1) with $y_t = 20 + \varepsilon_t + 0.8\varepsilon_{t-1}$. Right: MA(2) with $y_t = \varepsilon_t - \varepsilon_{t-1} + 0.8\varepsilon_{t-2}$. In both cases, ε_t is normally distributed white noise with mean zero and variance one.

Figure shows some data from an MA(1) model and an MA(2) model. Changing the parameters $\theta_1, \ldots, \theta_q$ results in different time series patterns. As with autoregressive models, the variance of the error term ε_t will only change the scale of the series, not the patterns.

Q32. Invert an Autoregressive model to moving average model.

Invertibility AR(1)→MA(∞)

It is possible to write any stationary AR(p) model as an MA(∞) model. For example, using repeated substitution, we can demonstrate this for an AR(1) model:

$$\begin{aligned} y_t &= \phi_1 y_{t-1} + \varepsilon_t \\ &= \phi_1(\phi_1 y_{t-2} + \varepsilon_{t-1}) + \varepsilon_t \\ &= \phi_1^2 y_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\ &= \phi_1^3 y_{t-3} + \phi_1^2 \varepsilon_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\ &\text{etc.} \end{aligned}$$

Provided $-1 < \phi_1 < 1$, the value of ϕ_1^k will get smaller as k gets larger. So eventually we obtain

$$y_t = \varepsilon_t + \phi_1 \varepsilon_{t-1} + \phi_1^2 \varepsilon_{t-2} + \phi_1^3 \varepsilon_{t-3} + \cdots,$$

an MA(∞) process.

The reverse result holds if we impose some constraints on the MA parameters. Then the MA model is called **invertible**. That is, we can write any invertible MA(q) process as an AR(∞)

Q33. Invert a moving average model to autoregressive model.

Invertibility MA(1)→AR(∞)

For example, consider the MA(1) process, $y_t = \varepsilon_t + \theta_1 \varepsilon_{t-1}$. In its AR($\infty$) representation, the most recent error can be written as a linear function of current and past observations:

$$\varepsilon_t = \sum_{j=0}^{\infty} (-\theta)^j y_{t-j}.$$

When $|\theta| > 1$, the weights increase as lags increase, so the more distant the observations the greater their influence on the current error. When $|\theta| = 1$, the weights are constant in size, and the distant observations have the same influence as the recent observations. As neither of these situations make much sense, we require $|\theta| < 1$, so the most recent observations have higher weight than observations from the more distant past. Thus, the process is invertible when $|\theta| < 1$.

The invertibility constraints for other models are similar to the stationarity constraints.

- For an MA(1) model: $-1 < \theta_1 < 1$.
- For an MA(2) model: $-1 < \theta_2 < 1$, $\theta_2 + \theta_1 > -1$, $\theta_1 - \theta_2 < 1$.

Q34. What are ARIMA models?

Auto Regressive Integrated Moving Average' is actually a class of models that 'explains' a given time series based on its own past values, that is, its own lags and the lagged forecast errors, so that equation can be used to forecast future values. Any 'non-seasonal' time series that exhibits patterns and is not a random white noise can be modeled with ARIMA models.

The ARIMA can be modelled as:

$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t,$$

Where

c is the y-intercept

y'_{t-1} are the lagged terms from AR model

$\varepsilon_{t-1} to\ \varepsilon_t$ are the errors from MA model

$\emptyset_i\ with\ i\ =\ 1,2,3\ldots.are\ the\ coefficients\ for\ the\ lagged\ term\ from\ AR\ model$

$\theta_i\ with\ i\ =\ 1,2,3\ldots.are\ the\ coefficients\ for\ the\ lagged\ errors\ from\ MA\ model$

Q35. What is d in the ARIMA model?

d is the number of differencing required to make the time series stationary

To make a series stationary?

- The most common approach is to difference it. That is, subtract the previous value from the current value. Sometimes, depending on the complexity of the series, more than one differencing may be needed.

- The value of d, therefore, is the minimum number of differencing needed to make the series stationary. And if the time series is already stationary, then d = 0.

Q36. What are the p and q terms?

'p' is the order of the 'Auto Regressive' (AR) term. It refers to the number of lags of Y to be used as predictors. And 'q' is the order of the 'Moving Average' (MA) term. It refers to the number of lagged forecast errors that should go into the ARIMA Model.

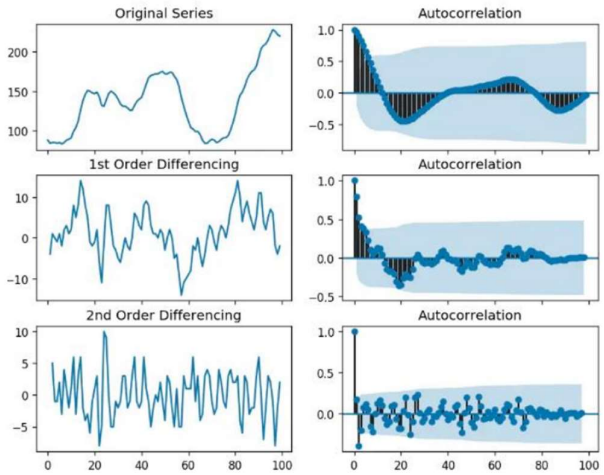
Table 8.1: Special cases of ARIMA models.

White noise	ARIMA(0,0,0)
Random walk	ARIMA(0,1,0) with no constant
Random walk with drift	ARIMA(0,1,0) with a constant
Autoregression	ARIMA(p,0,0)
Moving average	ARIMA(0,0,q)

Q37. How to find the order of differencing (d) in ARIMA model?

- The purpose of differencing it to make the time series stationary.
- But you need to be careful to not over-difference the series. Because, an over differencedseries may still be stationary, which in turn will affect the model parameters.
- The right order of differencing is the minimum differencing required to get a near-stationary

series which roams around a defined mean and the ACF plot reaches to zero fairlyquick.



- If the autocorrelations are positive for many number of lags (10 or more), then the seriesneeds further differencing. On the other hand, if the lag 1 autocorrelation itself is toonegative, then the series is probably over-differenced.
- In the event, you can't really decide between two orders of differencing, then go with theorder that gives the least standard deviation in the differenced series.
- the time series reaches stationarity with two orders of differencing. But on looking at the autocorrelation plot for the 2nd differencing the lag goes into the far negative zone fairly quick, which indicates, the series might have been over differenced.
- So, we can tentatively fix the order of differencing as 1 even though the series is not perfectly stationary (weak stationarity).

Q38. Difference between acf and pacf.

Aspect	ACF (Autocorrelation Function)	PACF (Partial Autocorrelation Function)
Definition	Measures correlation between a time series and its lagged values	Measures correlation between a data point and another data point at a specific lag, accounting for other lags in between
Interpretation	Provides information about the overall pattern of autocorrelation	Identifies direct influence, helping in AR component identification
Role in Model Identification	Useful for identifying the order of the moving average (MA) component	Particularly useful for identifying the order of the autoregressive (AR) component
Graphical Representation	ACF plot shows autocorrelation values at all lags	PACF plot shows partial autocorrelations at specific lags
Mathematical Relationship	Autocorrelation at lag k is the correlation between the time series and its values at lag k	Partial autocorrelation at lag k is the correlation between the time series and its values at lag k, with other lags' influence removed
AR(p)	Tail cuts off after lag p	Typically tails off after lag p, with some exceptions indicating additional structure
MA(q)	Tails off after lag q	Tail cuts off after lag q, indicating the end of significant moving average effects
ARIMA(p, q)	Tail cuts off after max(p, q)	Tail cuts off after max(p, q), representing the combined effect of autoregressive and moving average components

Autocorrelation Function (ACF)	Partial Autocorrelation Function (PACF)
Measures correlation of a series with its own lagged values at different lags.	Measures the correlation between two variables while accounting for the intermediate correlations.
Shows correlation at each lag 'k' including indirect correlations through lags '1' to 'k-1'.	Shows the correlation between two variables removing the effect of correlations at lags in between.
Used to identify potential autoregressive (AR) and moving average (MA) terms in time series models.	Used to determine the appropriate lag order for AR terms in time series models.
Helps to identify the order of the MA process (q) in an ARMA model.	Helps to identify the order of the AR process (p) in an ARMA model.
Decays gradually or in a sinusoidal pattern for a stationary time series.	Cuts off after the specified lag for a stationary time series.
Helps identify seasonality and general patterns in data.	Helps in understanding direct relationships between variables while removing indirect effects.

Q39. Difference b/w arma and arima models?

Aspect	ARMA (AutoRegressive Moving Average)	ARIMA (AutoRegressive Integrated Moving Average)
Full Name	AutoRegressive Moving Average	AutoRegressive Integrated Moving Average
Components	Autoregressive (AR) and Moving Average (MA)	Autoregressive (AR), Integrated (I), and Moving Average (MA)
Integration (I)	Does not include an integration component (d = 0)	Includes an integration component to make the series stationary (d > 0)
Purpose	Used for stationary time series data	Used for non-stationary time series data after differencing
Equation	•ARMA Model Equation •$r(t)=C+\phi r(t-1)+\theta\varepsilon(t-1)+\varepsilon(t)$ where, •r(t),r(t-1) = current value and value one period ago. •ε(t),ε(t-1) = current error term and one period ago. •c = baseline constant factor. •ϕ = value coefficient, what part of the last period value is relevant in explaining the current value. •θ = error coefficient, what part of the last period value is relevant in explaining the current error value.	2. ARIMA Model Equation $\Delta r(t)=C+\phi\Delta r(t-1)+\theta\varepsilon(t-1)+\varepsilon(t)$, where, •$\Delta r(t)= r(t)-r(t-1)$, difference in consecutive period. •other is same as the ARMA model.
Stationarity	Assumes the time series is stationary	Allows for non-stationary time series after differencing
Differencing (I)	Does not include differencing (I = 0)	Includes differencing to achieve stationarity (I > 0)
Application	Suitable for stationary data with constant mean and variance	Suitable for non-stationary data with trends and seasonality
Model Selection	ARMA(p, q), where p is the order of the autoregressive component and q is the order of the moving average component	ARIMA(p, d, q), where p is the order of the autoregressive component, d is the order of differencing, and q is the order of the moving average component

Example	ARMA(2, 1): $Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \epsilon_t + \theta_1 \epsilon_{t-1}$	ARIMA(1, 1, 1): $Y'_t = c + \phi_1 Y'_{t-1} + \epsilon_t + \theta_1 \epsilon_{t-1}$, where Y'_t is the differenced series

ARMA (Autoregressive Moving Average)	ARIMA (Autoregressive Integrated Moving Average)
Combines autoregressive (AR) and moving average (MA) components without differencing.	Incorporates differencing to achieve stationarity before applying ARMA components.
Suitable for stationary time series data.	Suitable for non-stationary time series data.
Requires the time series data to be stationary throughout (constant mean and variance).	Can handle non-stationary data by differencing to achieve stationarity.
Model is denoted as ARMA(p, q), where 'p' denotes the order of the AR part and 'q' denotes the order of the MA part.	Model is denoted as ARIMA(p, d, q), where 'p' is the order of the AR part, 'd' is the degree of differencing, and 'q' is the order of the MA part.
Assumes that the series is already stationary.	Allows for differencing the series to achieve stationarity before applying the ARMA model.
Used when the time series data exhibits autocorrelation but doesn't have a trend or seasonality.	Suitable for time series data with trends, seasonality, or other non-stationary patterns.
Focuses on modeling the correlation between observations based on their lagged values.	Addresses both autocorrelation and non-stationarity by differencing the series.
Does not include differencing steps as part of the model.	Includes differencing as an integral part of the model to make the series stationary.
Assumes no need for transformation to achieve stationarity.	Often requires examination and application of differencing to achieve stationarity before fitting the ARIMA model.

Q40. How to handle if a time series is slightly under or over differenced?

Ans

Handling Under-Differencing:

1. **Increase the Order of Differencing (*d*):**
    - If the time series is not stationary after the initial differencing (d=1), try increasing the order of differencing (d=2,3,…).
    - Examine the differenced series and check for stationarity using statistical tests or visual inspection.

2. **Check for Seasonal Patterns:**
    - If there are clear seasonal patterns in the data, consider applying seasonal differencing in addition to regular differencing.
3. **Use Augmented Dickey-Fuller (ADF) Test:**
    - Conduct an Augmented Dickey-Fuller test to formally test for stationarity. The test helps in determining the appropriate order of differencing.
4. **Add AR Terms (*p*):**
    - If the series is slightly under-differenced, consider adding one or more additional AR terms (*p*) to the model.
    - The additional AR terms may capture remaining autocorrelation patterns in the residuals, compensating for the under-differencing.

Handling Over-Differencing:

1. **Reduce the Order of Differencing (*d*):**
    - If the time series shows excessive randomness or noise after differencing, consider reducing the order of differencing (*d*=0,1).
    - Revert back to the original series or the differenced series with a lower order to see if the patterns become clearer.
2. **Check for Seasonal Patterns:**
    - Ensure that you haven't overlooked seasonal patterns in the data. If needed, explore seasonal differencing or adjust the seasonal component of your model.
3. **Evaluate Model Performance:**
    - Assess the performance of different models with varying levels of differencing using metrics like AIC (Akaike Information Criterion) or BIC (Bayesian Information Criterion).
4. **Add MA Terms (*q*):**
    - If the series is slightly over-differenced, consider adding one or more additional MA terms (*q*) to the model.
    - The extra MA terms may help in introducing some correlation between neighboring residuals, mitigating the excess randomness introduced by over-differencing.

Q41. What is vector autoregression and how it different from the normal autoregression model?

Vector Autoregression (VAR) is a statistical model used in time series analysis to capture the joint dynamic relationship between multiple variables over time. Unlike normal autoregressive models, which focus on predicting a single variable based on its past values, VAR models simultaneously model the behavior of multiple variables, considering their mutual interactions and feedback mechanisms.

Aspect	**Vector Autoregression (VAR)**	**Autoregressive (AR)**
Focus	Joint dynamics of multiple variables	Prediction of a single variable

Equation Form	$Y_t = A_1 Y_{t-1} + A_2 Y_{t-2} + \ldots + A_p Y_{t-p} + \epsilon_t$	$Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots + \phi_p Y_{t-p} + \epsilon_t$
Variables	Multiple time series variables in a vector	A single time series variable
Endogeneity	Allows for endogenous relationships	Focused on the endogeneity of the target variable
Simultaneous Equations	Represents a system of simultaneous equations	Single-equation model
Impulse Response Analysis	Allows for impulse response analysis	Typically not used for impulse response analysis
Interdependencies	Explicitly models interdependencies and feedback	Considers only lagged values of the target variable
Application	Suitable for modelling systems with interconnected variables	Suitable for univariate time series forecasting
Equation Structure	Vector equation structure with coefficient matrices	Single equation with autoregressive coefficients
Stationarity Requirement	Stationarity assumption is often considered	Stationarity is often assumed

Q42. How does a VAR model's formula look like?

Let's suppose, you have two variables (Time series) Y1 and Y2, and you need to forecast the values of these variables at time (t). To calculate Y1(t), VAR will use the past values of both Y1 as well as Y2. Likewise, to compute Y2(t), the past values of both Y1 and Y2 be used.

- For example, the system of equations for a VAR(1) model with two time series (variables `Y1` and `Y2`) is as follows:
- Where, Y{1,t-1} and Y{2,t-1} are the first lag of time series Y1 and Y2 respectively.
- The above equation is referred to as a VAR(1) model, because, each equation is of order 1, that is, it contains up to one lag of each of the predictors (Y1 and Y2).
- Since the Y terms in the equations are interrelated, the Y's are considered as endogenous variables, rather than as exogenous predictors.

$$Y_{1,t} = \alpha_1 + \beta_{11,1} Y_{1,t-1} + \beta_{12,1} Y_{2,t-1} + \epsilon_{1,t}$$
$$Y_{2,t} = \alpha_2 + \beta_{21,1} Y_{1,t-1} + \beta_{22,1} Y_{2,t-1} + \epsilon_{2,t}$$

- Likewise, the second order VAR(2) model for two variables would include up to two lags for each variable (Y1 and Y2).

$$Y_{1,t} = \alpha_1 + \beta_{11,1} Y_{1,t-1} + \beta_{12,1} Y_{2,t-1} + \beta_{11,2} Y_{1,t-2} + \beta_{12,2} Y_{2,t-2} + \epsilon_{1,t}$$
$$Y_{2,t} = \alpha_2 + \beta_{21,1} Y_{1,t-1} + \beta_{22,1} Y_{2,t-1} + \beta_{21,2} Y_{1,t-2} + \beta_{22,2} Y_{2,t-2} + \epsilon_{2,t}$$

•Can you imagine what a second order VAR(2) model with three variables (Y1, Y2 and Y3) would look like?

$$Y_{1,t} = \alpha_1 + \beta_{11,1} Y_{1,t-1} + \beta_{12,1} Y_{2,t-1} + \beta_{13,1} Y_{3,t-1} + \beta_{11,2} Y_{1,t-2} + \beta_{12,2} Y_{2,t-2} + \beta_{13,2} Y_{3,t-2} + \epsilon_{1,t}$$
$$Y_{2,t} = \alpha_2 + \beta_{21,1} Y_{1,t-1} + \beta_{22,1} Y_{2,t-1} + \beta_{23,1} Y_{3,t-1} + \beta_{21,2} Y_{1,t-2} + \beta_{22,2} Y_{2,t-2} + \beta_{23,2} Y_{3,t-2} + \epsilon_{2,t}$$
$$Y_{3,t} = \alpha_3 + \beta_{31,1} Y_{1,t-1} + \beta_{32,1} Y_{2,t-1} + \beta_{33,1} Y_{3,t-1} + \beta_{31,2} Y_{1,t-2} + \beta_{32,2} Y_{2,t-2} + \beta_{33,2} Y_{3,t-2} + \epsilon_{3,t}$$

Q43. Advantages of Statistical models

1. **Simplicity and Transparency:**
    - Statistical models are simple and transparent.
    - They can be easily understood in terms of their parameters.
    - Facilitates quick model interpretation.
2. **Applicability to Small Data Sets:**
    - These models perform well with relatively small data sets.
    - They provide reliable results without requiring extensive data.
    - Ideal for scenarios with limited data availability.
3. **Performance Comparable to Complex Models:**
    - Despite their simplicity, statistical models and their related modifications exhibit high performance.
    - They perform extremely well, even when compared to sophisticated machine learning models.
    - Achieve competitive accuracy without the risk of overfitting.
4. **Mitigation of Overfitting Risk:**
    - The simplicity of statistical models reduces the risk of overfitting.
    - They are less prone to capturing noise in the training data, ensuring that learned patterns generalize to new, unseen data.
    - Enhances generalization to new data by minimizing overfitting.
5. **Automated Methodologies for Model Selection:**
    - Well-developed automated methodologies exist for choosing the appropriate orders of models and estimating their parameters.
    - These automated processes simplify the model-building task, making it easier for practitioners to generate accurate forecasts.
    - Streamlines the model selection process, improving efficiency.

Q44. Disadvantages of the statistical models.

1. **Limited Improvement with Large Data Sets:**

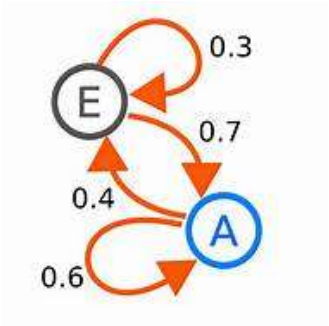
- Because statistical models are quite simple, they may not always improve performance with large data sets.
- For extremely large data sets, complex machine learning models and neural network methodologies may offer better results.

2. **Focus on Point Estimates, Limited Uncertainty Expression:**
    - Statistical models put the focus on point estimates of the mean value of a distribution, rather than on the distribution itself.
    - While sample variances can be derived as proxies for uncertainty, the fundamental model offers limited ways to express uncertainty relative to all the choices made in selecting a model.
3. **Inability to Handle Nonlinear Dynamics:**
    - By definition, statistical models are not built to handle nonlinear dynamics.
    - They perform poorly when describing data where nonlinear relationships are dominant.
4. **Limited Expression of Uncertainty:**
    - Statistical models primarily provide point estimates, offering limited expression of uncertainty in comparison to the diverse choices made during the model selection process.
    - The focus on mean values restricts the model's ability to convey the full range of uncertainties associated with forecasting.

Q45. What are Markov models?

- Markov models are a type of probabilistic model that is used to predict the future state of a system, based on its current state.
- In other words, Markov models are used to predict the future state based on the current hidden or observed states. Markov model is a finite-state machine where each state has an associated probability of being in any other state after one step. They can be used to model real-world problems where hidden and observable states are involved.
- Markov models can be classified into hidden, and observable based on the type of information available to use for making predictions or decisions.
- Hidden Markov models deal with hidden variables that cannot be directly observed but only inferred from other observations, whereas in an observable model also termed as Markov chain, hidden variables are not involved.

Q46. What is markov model, markov chain and markov process and what is their link or relation?



1. **Markov Model**: A Markov model is a type of mathematical model used to represent systems that change over time. It is a stochastic model, meaning it incorporates elements of randomness. The key characteristic of a Markov model is that it assumes the Markov property, which means that future states of the system depend only on the current state, not on the sequence of events that preceded it[12].

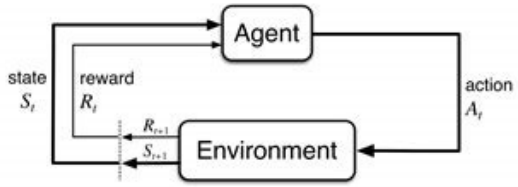
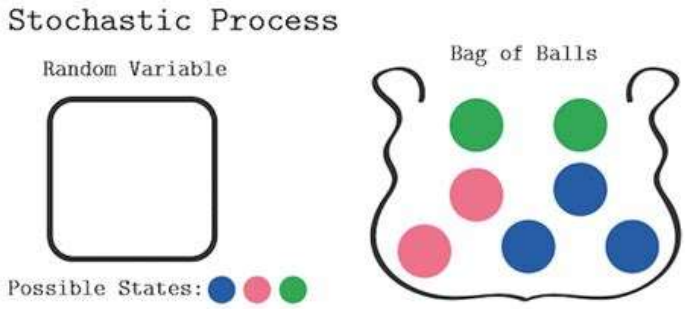


Figure 3.1: The agent–environment interaction in a Markov decision process.

2. **Markov Chain**: A Markov chain is a special case of a Markov model. It is a sequence of events, where the probability of each event depends only on the state attained in the previous event[34]. It can be visualized as a state diagram where each node represents a state, and the edges represent the probability of transitioning from one state to another[5].

3. **Markov Process**: A Markov process is a type of Markov model that is indexed by time, and the future is independent of the past, given the present[6]. It can be thought of as a continuous-time version of a Markov chain[7].



Relation:

- A Markov chain is a specific type of Markov model that deals with discrete-time processes and a finite set of states.
- A Markov process is a broader term encompassing both continuous-time and discrete-time processes, making it a more general framework.
- In summary, all Markov chains are Markov processes, but not all Markov processes are necessarily Markov chains. Markov models can refer to either continuous or discrete processes that exhibit the Markov property.

The relationship between these three concepts is that they all incorporate the Markov property, which is the principle that the future state depends only on the current state and not on the sequence of past states. A Markov chain is a specific type of Markov model, and a Markov process is a continuous-time version of a Markov chain[8].

Q47. What are hidden Markov models?

HMM is a statistical model in which the system being modeled are Markov processes with unobserved or hidden states. It is a hidden variable model which can give an observation of another hidden state with the help of the Markov assumption.

The hidden state is the term given to the next possible variable which cannot be directly observed but can be inferred by observing one or more states according to Markov's assumption.

An HMM consists of two types of variables: hidden states and observations. The hidden states are the underlying variables that generate the observed data, but they are not directly observable. The observations are the variables that are measured and observed.

Hidden Markov models.

- Set of states: $\{s_1, s_2, \ldots, s_N\}$
- Process moves from one state to another generating a sequence of states : $s_{i1}, s_{i2}, \ldots, s_{ik}, \ldots$
- Markov chain property: probability of each subsequent state depends only on what was the previous state:

$$P(s_{ik} \mid s_{i1}, s_{i2}, \ldots, s_{ik-1}) = P(s_{ik} \mid s_{ik-1})$$

- States are not visible, but each state randomly generates one of M observations (or visible states) $\{v_1, v_2, \ldots, v_M\}$

- To define hidden Markov model, the following probabilities have to be specified: matrix of transition probabilities $A=(a_{ij})$, $a_{ij}= P(s_i \mid s_j)$, matrix of observation probabilities $B=(b_i(v_m))$, $b_i(v_m)= P(v_m \mid s_i)$ and a vector of initial probabilities $\pi=(\pi_i)$, $\pi_i = P(s_i)$. Model is represented by $M=(A, B, \pi)$.

The relationship between the hidden states and the observations is modeled using a probability distribution[2]. The HMM is the relationship between the hidden states and the observations using two sets of probabilities: the transition probabilities and the emission probabilities[2]. The transition probabilities describe the probability of transitioning from one hidden state to another[2]. The emission probabilities describe the probability of observing an output given a hidden state[2].

Here are a couple of examples to illustrate the concept:

1. **Predicting the Weather**: Consider a simple example of an HMM where we are predicting the weather (hidden variable) based on the type of clothes that someone wears (observed)[3].
2. **Drawing Balls from Hidden Urns**: In its discrete form, a hidden Markov process can be visualized as a generalization of the urn problem with replacement (where each item from the urn is returned to the original urn before the next step)[4]. Consider this example: in a room that is not visible to an observer there is a genie. The room contains urns X1, X2, X3, … each of which contains a known mix of balls, each ball labeled y1, y2, y3, … . The genie chooses an urn in that room and randomly draws a ball from that urn. It then puts the ball onto a conveyor

belt, where the observer can observe the sequence of the balls but not the sequence of urns from which they were drawn[4].

Q48. What do you mean by Kalman filter? What are its applications?

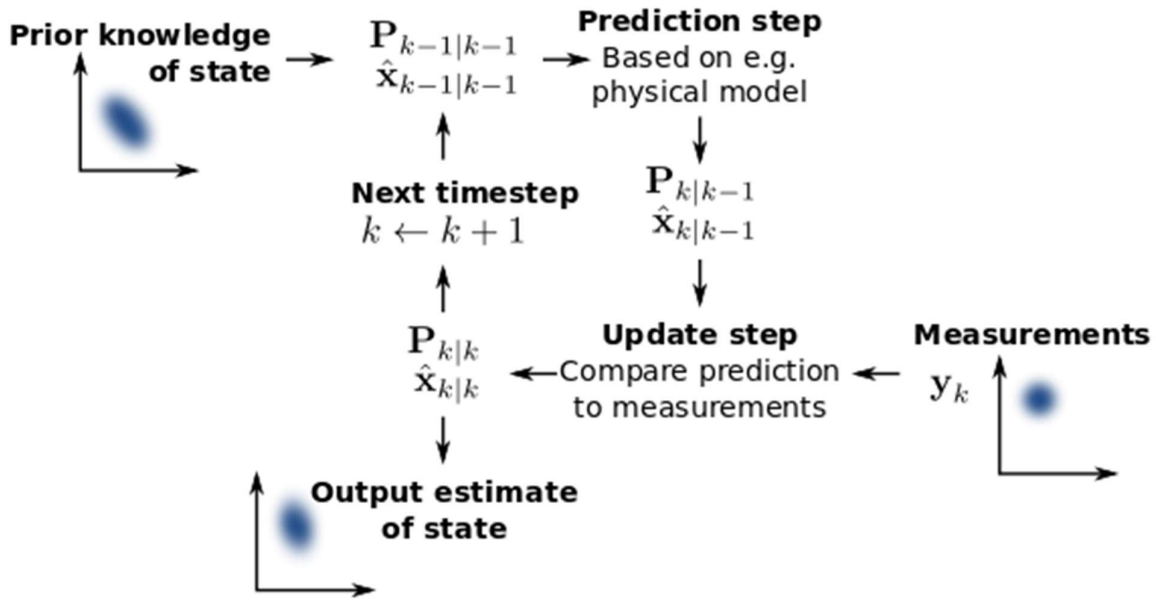
https://www.youtube.com/watch?v=mwn8xhgNpFY

A Kalman filter is an optimal estimation algorithm.

The Kalman filter, named after Rudolf E. Kálmán, is a mathematical method used in control theory and statistics to estimate the state of a system based on a series of measurements observed over time[1]. It's a recursive estimator, meaning only the estimated state from the previous time step and the current measurement are required to compute the estimate for the current state[2].

The Kalman filter works in two phases: the prediction phase and the update phase[1]. In the prediction phase, the Kalman filter produces estimates of the current state variables, along with their uncertainties[1]. Once the outcome of the next measurement (which is necessarily corrupted with some error, including random noise) is observed, these estimates are updated using a weighted average, with more weight being given to estimates with higher certainty[1].

For a visual explanation of how Kalman filters work, I recommend checking out this video which provides a visual introduction to Kalman Filters and the intuition behind them. There's also a detailed tutorial that visually explains the process of a Kalman filter, including a vehicle location estimation example.



The Kalman Filter has a wide range of applications in various fields:

1. **Navigation and Control Systems:** It is extensively used in aerospace for trajectory estimation of aircraft and spacecraft, GPS navigation, and robotics.
2. **Guidance, Navigation, and Control**: Inertial Measurement Unit (IMU) sensors can be used to estimate an object's location, velocity, and acceleration, and these estimates can be used to control the object's next moves[3].
3. **Economics and Finance:** In econometrics, the Kalman Filter is used for signal extraction in time series analysis, such as separating a signal that evolves over time from "noise".

4. **Engineering:** It is used for sensor fusion, where it combines data from various sensors to compute the best estimate of the state of interest.
5. **Computer Vision:** The Kalman Filter can track moving objects in video streams or predict the position of a moving object.
6. **Object Tracking**: Kalman filters can be used to use the measured position of an object to more accurately estimate the position and velocity of that object[3].
7. **Body Weight Estimate on Digital Scale**: They can also be used to estimate the weight of an object on a surface based on the measured pressure on that surface[3].

Q49. How Does the Kalman Filter Work?

The Kalman Filter operates in two steps: the "predict" or "time update" phase and the "update" or "measurement update" phase.

Predict Phase

In the predict phase, the Kalman Filter uses the state from the previous time step to produce estimates of the current state. This prediction includes the estimation of the system's state variables and the uncertainty of the estimate. The uncertainty is often expressed as a covariance matrix, which is a measure of the "spread" or the expected accuracy of the prediction.

Update Phase

During the update phase, the current prediction is combined with the current observation to refine the state estimate. This step adjusts the predicted state by a factor proportional to the difference between the actual measurement and the prediction. The Kalman Filter uses the covariance matrix to weigh the accuracy of the prediction against the accuracy of the new measurement, thus updating the state estimate and its uncertainty.

These two phases are repeated in a loop, with each iteration refining the estimates. This process allows the filter to react to new measurements and improve the estimate over time, which is why it is particularly useful for systems where the measurements are uncertain or vary over time.

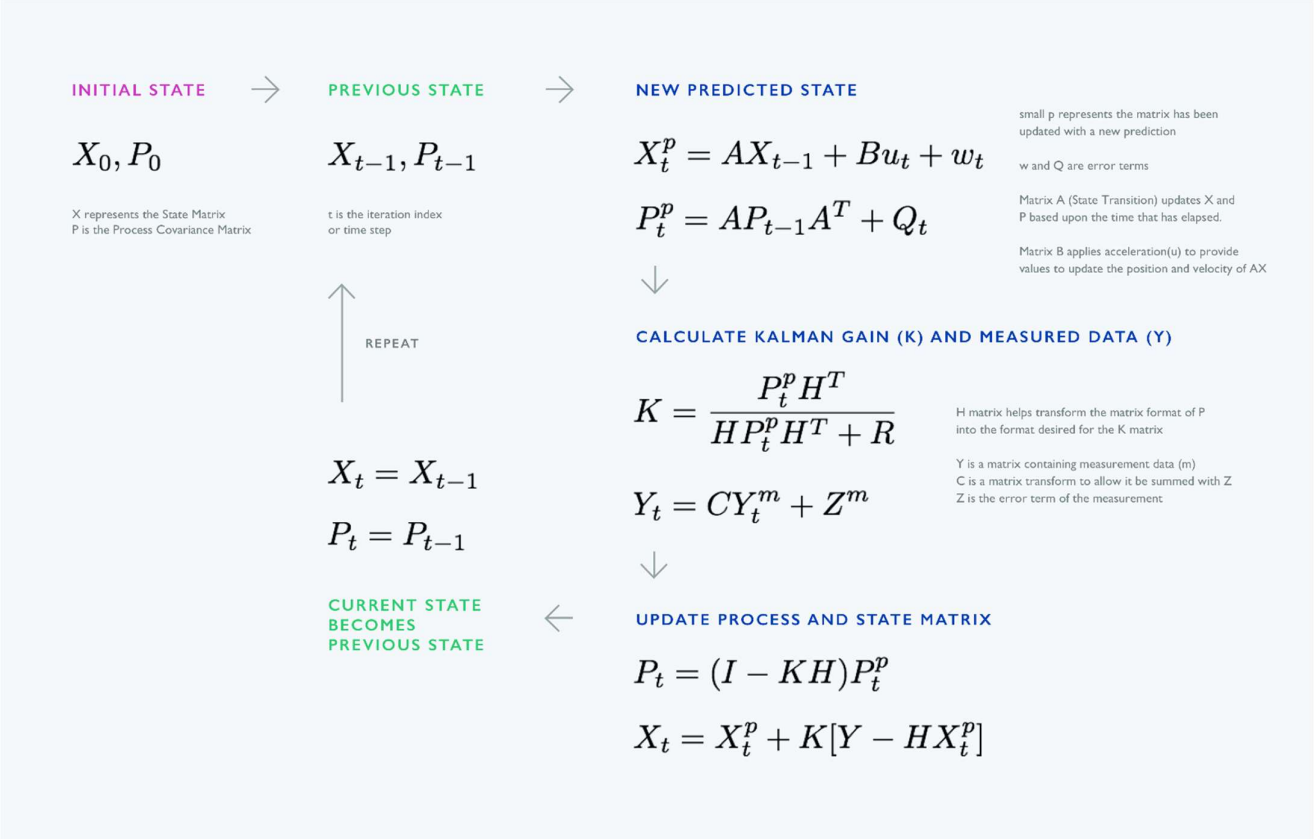
Q50. Write the algorithm for the Kalman Filter.

let's break down the Kalman filter algorithm into its basic steps:

1. **Initialization**: We start by initializing the state of the system and the error covariance matrix. The state of the system could be any variable we're interested in estimating[1]. The error covariance matrix is a measure of the estimated accuracy of the state estimate[2].
2. **Prediction**: In this step, we predict the state of the system at the next time step using the process model[1]. We also predict the error covariance matrix at the next time step[2].
3. **Update**: Once we have the actual measurement, we compute the residual (the difference between the predicted state and the actual measurement)[1]. We also compute the Kalman gain, which is used to update the state estimate[2].
4. **Estimate Update**: We update the state estimate and the error covariance matrix using the Kalman gain and the residual[12].

5. **Repeat**: We repeat steps 2-4 for each new measurement[1].

https://www.kalmanfilter.net/multiSummary.html



Q51. Why do we use Kalman filters? Give some examples as well.

Kalman Filters are used for state estimation in dynamic systems where there is uncertainty and noise in measurements. They are particularly valuable in situations where the true state of a system is not directly observable, but it can be estimated from a series of imperfect measurements. Here are some reasons why Kalman Filters are widely used:

1. **Optimal State Estimation:**
    - Kalman Filters provide optimal estimates by minimizing the mean square error between the estimated and true states. In situations with Gaussian noise and linear dynamics, the Kalman Filter gives the best linear unbiased estimate.
2. **Adaptability to Noisy Environments:**
    - In real-world applications, measurements are often subject to noise. Kalman Filters are designed to handle noisy measurements and provide accurate state estimates, even in the presence of uncertainty.
3. **Recursive Updating:**
    - Kalman Filters operate in a recursive manner, continually updating their estimates as new measurements become available. This adaptability to changing conditions makes them suitable for dynamic systems.
4. **Efficient Handling of Dynamic Systems:**

- Kalman Filters are well-suited for dynamic systems with linear state transition and observation models. They efficiently handle situations where the relationship between the state and measurements is linear and the system is subject to stochastic processes.

5. **Sensor Fusion:**
    - Kalman Filters are often employed in sensor fusion applications, where information from multiple sensors is combined to provide a more accurate and reliable estimate of the system state. This is crucial in navigation systems, robotics, and autonomous vehicles.
6. **Tracking Moving Objects:**
    - Kalman Filters are widely used in tracking applications, such as radar tracking of moving objects. They can predict the future position of an object based on past observations and adjust the prediction as new measurements are received.
7. **Financial Modeling:**
    - Kalman Filters find applications in financial modeling, particularly in estimating hidden factors or states in financial time series data. They can be used to filter out noise and provide more accurate estimates of underlying trends.
8. **Control Systems:**
    - Kalman Filters are used in control systems for state estimation. They help control systems adapt to changes and disturbances, making them more robust and responsive.

Examples of Kalman Filter Applications:

1. **Navigation Systems:**
    - Inertial navigation systems use Kalman Filters to fuse data from accelerometers and gyroscopes, providing accurate estimates of position and orientation.
2. **Aircraft Tracking:**
    - Kalman Filters are employed in radar and sensor systems to track the position and velocity of aircraft, enabling accurate air traffic control.
3. **Robotics:**
    - Kalman Filters are used in robotics for localization and mapping, allowing robots to estimate their position and environment.
4. **Financial Time Series Analysis:**
    - Kalman Filters can be applied to financial time series data to estimate hidden factors influencing stock prices or economic indicators.
5. **Autonomous Vehicles:**
    - In autonomous vehicles, Kalman Filters are used to fuse data from various sensors, such as GPS, lidar, and radar, to estimate the vehicle's position and navigate through the environment.

Q52. Advantages and Limitations of Kalman Filter.

The Kalman filter has several advantages:

1. **Linear Estimation and Gaussian Assumption**: The Kalman Filter is advantageous because it is a linear estimator. It operates optimally under the assumption that the errors are Gaussian, making it suitable for systems with normally distributed noise.
2. **Computational Efficiency for Real-Time Applications**: The Kalman Filter is computationally efficient, allowing it to run in real-time applications. Its efficiency makes it applicable to systems that require quick and continuous state estimation updates.
3. **Handling Unknown Noise Statistics**: The Kalman Filter is capable of handling cases where the noise statistics are not fully known. This adaptability is valuable in practical situations where the exact characteristics of the noise may not be precisely defined.
4. **Robustness to Noise**: The method is very robust to measurement noise[1].
5. **Optimal Estimation**: It provides an optimal estimation method for linear systems with Gaussian error statistics[2].
6. **Real-Time Operation**: The algorithm is recursive and can operate in real time, using only the present input measurements and the state calculated previously and its uncertainty matrix; no additional past information is required[3].
7. **Light on Memory**: They don't need to keep any history other than the previous state, making them well suited for real-time problems and embedded systems[4].
8. **Adaptability**: The Kalman filter can adjust the Kalman gain according to the actual measurement accuracy, so as to obtain the optimal solution[5].

However, it also has some limitations:

1. **Gaussian and White Noise Assumptions:** One limitation of the Kalman Filter is its assumption that both the process and measurement noise are Gaussian and white. In real-world scenarios, noise may not always follow these idealized distributions.
2. **Linear System Dynamics Assumption:** The Kalman Filter assumes that the system dynamics are linear. This limitation can be restrictive, as many real-world systems exhibit non-linear behavior. In such cases, the Kalman Filter may not perform optimally.
3. **System Model**: The effectiveness of the Kalman filter is highly dependent on the accuracy of the system model[5].
4. **Initial Conditions**: While it does not depend on good initial conditions (after finite time the filter will converge to the correct system state), it does require a larger computational complexity to get better results[1].
5. **External Factors**: Many external hidden factors like thermal noise, receiver clock precession, materials, GPS satellite location, etc., can create problems in accuracy and precision[6].
6. **Comparison with Other Filters**: For single-input, single-output systems, a plain old IIR or FIR filter, specified in the frequency domain, can sometimes do a better job (because of robustness issues) than some laboriously designed Kalman filter[5].

Q53. Give some instances of the Open Source Time Series Feature Generation Libraries.

1. The **tsfresh** Python module: It implements a large and general set of features, which includes:
    - Descriptive statistics
    - Physics-inspired indicators of nonlinearity and complexity
    - History-compressing counts
2. The **Cesium** time series analysis platform:
    - Features that describe the overall distribution of the data values, without regard to its temporal relationships.
    - Features that describe the distribution of the timing of data:
    - Features that describe measures of the periodicity of the behavior within the time series.

Q54. What is a decision tree? It's characteristics and applications?

A **Decision Tree** is a predictive modeling approach that represents a series of decisions and their possible consequences in a tree-like structure. This method is particularly adept at mimicking the way humans make decisions—one step at a time, and in a highly nonlinear fashion. Decision Trees are widely used in various domains, such as stock marketing and the analysis of medical signals like EEG (Electroencephalogram) and ECG (Electrocardiogram).

Key Characteristics of Decision Trees:

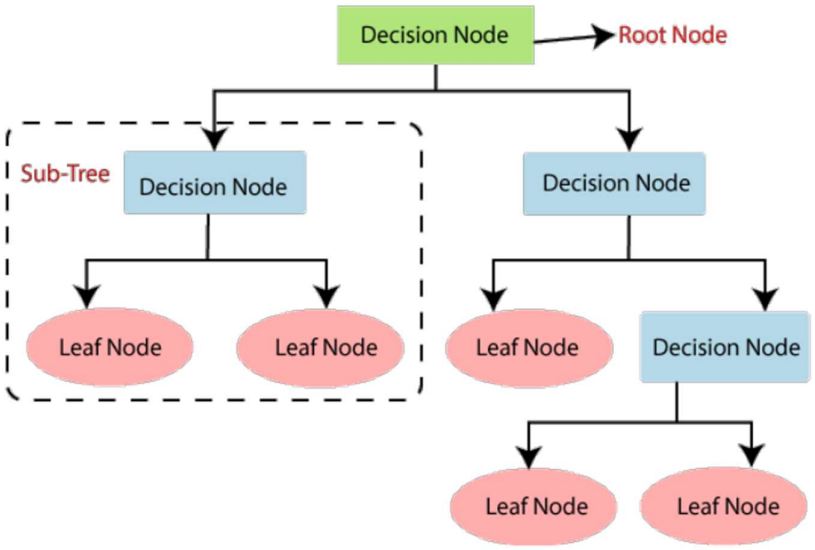
1. **Tree-Based Structure:**
    - Decision Trees are structured like a tree, where each node represents a decision or a test on a particular attribute.
2. **Sequential Decision-Making:**
    - The decision-making process follows a sequential, step-by-step approach. At each decision node, a test is performed on a specific attribute, leading to different branches based on the outcome of that test.
3. **Nonlinear Decision Logic:**
    - Unlike linear models, Decision Trees allow for highly nonlinear decision logic. The path from the root node to the leaf nodes captures the intricate decision-making process.
4. **Representation of Events and Decisions:**
    - The tree-like model illustrates a series of events and decisions that lead to specific outcomes. Each branch in the tree represents an outcome of a particular test, guiding the flow of decisions.
5. **Attribute Tests:**
    - Each node in the tree represents a test on a specific attribute. The test determines the criteria for branching to subsequent nodes, influencing the final decision at the leaf nodes.

Example Applications:

1. **Stock Marketing:**
    - Decision Trees can be used in stock marketing to model the decision-making process based on various factors such as market conditions, financial indicators, and historical trends.
2. **Medical Signal Analysis:**
    - In the context of medical signals like EEG and ECG, Decision Trees can aid in analyzing and interpreting the signals to make decisions related to patient health or diagnosis.
3. **Classification and Regression Tasks:**
    - Decision Trees are commonly employed for classification tasks, where the goal is to assign an object or an observation to a specific category. They can also be used for regression tasks to predict numerical values.
4. **Customer Churn Prediction:**
    - Decision Trees are utilized in business analytics, such as predicting customer churn. By analyzing various customer-related factors, the tree can help predict whether a customer is likely to churn or not.

Q55. Give the general terminologies in Decision Tree.

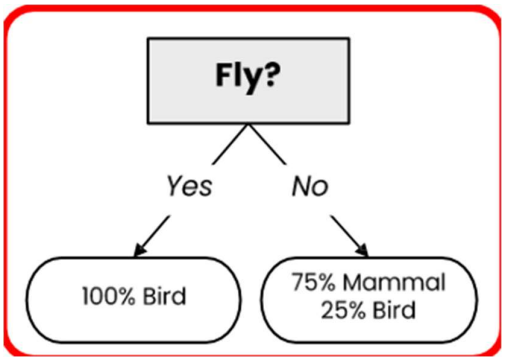
- **Root Node:** Represents the entire population and further divides into two or more homogeneous sets.
- **Splitting:** The process of dividing a node into two or more sub-nodes.
- **Decision Node:** Occurs when a sub-node splits into further sub-nodes.
- **Leaf / Terminal Node:** Nodes that do not split.
- **Pruning:** Involves removing sub-nodes of a decision node.



- **Branch/Sub-Tree:** A subsection of the entire tree.
- **Parent and Child Node:** A node divided into sub-nodes is called a parent node, and the sub-nodes are the children of the parent node.

Q56. what are the pure subsets

In a decision tree, a pure subset refers to a subset of data where all instances belong to the same class[1]. This is the ideal scenario for a decision tree, as it means that the tree has accurately classified all instances in that subset[2].

Example1: let's consider a decision tree that is trying to predict whether a person will buy a new car based on their income and age[3]. The root node might split the data based on income, creating two subsets: one for people with income less than or equal to $50,000, and one for people with income greater than $50,000[3]. The subset for people with income less than or equal to $50,000 might then be split based on age, creating further subsets[3]. If, in one of these subsets, all people are under the age of 30 and none of them bought a new car, this subset would be considered pure[3].	Example 2: 

Q57. When to stop Building the tree?

Deciding when to stop building decision trees is critical for achieving a balance between model complexity and generalization. Common criteria include:

1. **Pure Leaf Nodes:**
    - Stop when all leaf nodes are pure, containing data of a single class. This ensures that the tree effectively classifies instances.
2. **Height Limit:**
    - Consider stopping when the tree's height exceeds a predefined limit. This prevents deep trees that may capture noise in the training data and overfit, promoting better generalization.
3. **Minimum Samples per Leaf:**
    - Halt tree growth if the number of samples in a leaf falls below a specified threshold. This guards against overfitting to small data subsets, promoting robust predictions.
4. **Minimum Impurity Decrease:**
    - Stop building the tree if the impurity decrease from splitting a node is below a threshold. This avoids unnecessary splits that don't significantly enhance predictive performance.
5. **All Features Used:**
    - Stop when all available features are used in the tree. Additional splits may lead to overfitting, hindering the model's ability to generalize to new data.
6. **Early Stopping and Cross-Validation:**
    - Implement early stopping techniques, monitoring model performance on a validation set during training. Cross-validation helps identify the optimal tree depth where additional growth doesn't improve generalization.

Selecting the appropriate stopping criteria depends on the specific task and experimenting with different criteria while evaluating their impact on model performance is often necessary. This approach ensures that the decision tree achieves the right balance between simplicity and accuracy.

Q58. what are attribute selection measure? also explain the 2 popular techniques

Attribute selection measures, also known as feature selection or variable selection, are methods used in machine learning to choose the most relevant features or attributes for a model. The goal is to improve model performance, reduce overfitting, and enhance interpretability.

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems Attribute selection measure methods are used.

There are two popular techniques for ASM, which are:

1. **Entropy and Information Gain:**
    - Information Gain is a measure used in decision trees to evaluate the effectiveness of a feature in classifying the data. It is based on the concept of entropy from information theory.
    - Entropy measures the degree of disorder or randomness in a system.

$$E(S) = \sum_{i=1}^{c} -pi\ log_2 p_i$$

    - The Information Gain is calculated as the difference between the entropy of the parent node and the weighted sum of entropies of its child nodes after a split.

$$\text{Information Gain}(T, X) = \text{Entropy}(T) - \text{Entropy}(T, X)$$

    - Steps for Information Gain:
        - Calculate the entropy of the parent node.
        - For each possible split, calculate the entropy of the child nodes and their weighted sum.
        - Calculate the Information Gain as the difference between the parent node's entropy and the weighted sum of child node entropies.
        - Choose the split with the highest Information Gain.
2. **Gini Index:**
    - Gini Index is another measure used in decision trees to assess the impurity or disorder in a dataset. It quantifies the likelihood of incorrectly classifying an instance in a dataset. The Gini Index for a node is computed as the sum of the squared probabilities of each class subtracted from one. The Gini Index for a split is the weighted sum of the Gini Index values of its child nodes.
    - Steps for Gini Index:
        - Calculate the Gini Index for each feature's possible split.

- Choose the split with the lowest Gini Index, as it represents the least impurity.

Both Information Gain and Gini Index are employed during the construction of decision trees. The features with higher Information Gain or lower Gini Index are considered more informative for making decisions. These techniques aid in selecting attributes that contribute the most to the predictive power of the model, leading to improved accuracy and efficiency.

Q59. What is id3 algorithm? Provide its steps and advantages and limitations.

ID3 (Iterative Dichotomiser 3) is a classic and influential algorithm for building decision trees, specifically designed for classification tasks. It was developed by Ross Quinlan and is one of the foundational algorithms in the field of machine learning. ID3 is particularly known for its simplicity and effectiveness in constructing decision trees based on information gain.

Key Steps of the ID3 Algorithm:

1. **Selecting the Best Attribute:**
    - ID3 selects the attribute that provides the highest information gain to split the dataset. Information gain is calculated based on the entropy measure, representing the amount of disorder or uncertainty in a dataset.
2. **Creating Decision Nodes:**
    - A decision node is created based on the selected attribute, representing a decision or test condition. Each branch emanating from the decision node corresponds to a different outcome of the attribute test.
3. **Partitioning the Dataset:**
    - The dataset is partitioned into subsets based on the outcomes of the selected attribute. Each subset corresponds to a branch from the decision node.
4. **Recursion:**
    - The algorithm recursively applies the same process to each subset, creating decision nodes, selecting attributes, and partitioning the data until certain stopping criteria are met.
5. **Stopping Criteria:**
    - The recursion stops when one of the following conditions is met:
        - All instances in a subset belong to the same class (pure leaf node).
        - No more attributes are available for splitting.
        - A predefined depth limit is reached.
6. **Building the Tree:**
    - The recursive process of attribute selection and dataset partitioning continues until the tree is fully constructed, with decision nodes representing tests on different attributes and leaf nodes representing the final class predictions.

Advantages of ID3:

- **Simplicity:** ID3 is straightforward and easy to understand, making it accessible for educational purposes.
- **Interpretability:** The resulting decision trees are human-readable, aiding interpretability.

Limitations of ID3:

- **Binary Splits:** ID3 only handles binary splits, meaning each decision node considers two possible outcomes of an attribute.
- **Overfitting:** ID3 tends to create deep trees, which can lead to overfitting on the training data. Techniques like pruning are often employed to address this issue.

While ID3 itself is rarely used in practice due to its limitations, it laid the groundwork for more advanced decision tree algorithms, such as C4.5 and CART, which addressed some of its drawbacks and became more widely adopted in real-world applications.

Q60. Provide the algorithm of ID3 in python.

```
# ID = Iterative Dichotomiser
def ID3(X):
    node = TreeNode(X)
    if all_points_have_same_class(X):
        node.label = majority_label(X)
    else:
        a = select_attribute_with_highest_information_gain(X)
        if gain(X, a) == 0:
            node.label = majority_label(X)
        else:
            for v in values(a):
                Xv={x∈X | x[a] == v}
                node.children.append(ID3(Xv))
    return node
```

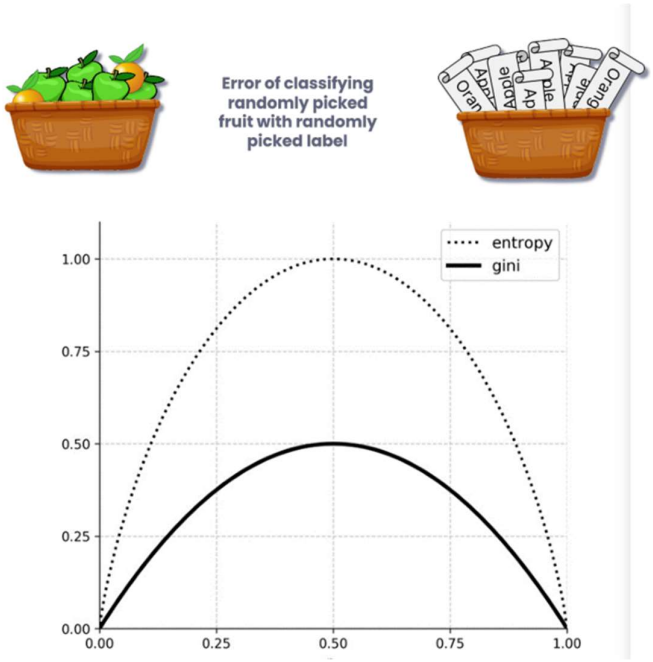
ID3(Examples, Target_attribute, Attributes)

- *Examples are the training examples.*
- *Target_attribute is the attribute whose value is to be predicted by the tree.*
- *Attributes is a list of other attributes that may be tested by the learned decision tree.*
- *Returns a decision tree that correctly classifies the given Examples.*

- Create a *Root* node for the tree
- If all *Examples are* positive, Return the single-node tree *Root,* with label → +
- If all *Examples are* negative, Return the single-node tree *Root, with* label → -
- If *Attributes* is empty, Return the single-node tree *Root,* with label = most common value of *Target_attribute in Examples*

- Otherwise Begin
  - A ← the attribute from *Attributes* that best* classifies *Examples*
  - The decision attribute for *Root* ← *A*
  - For each possible value, of A,
    - Add a new tree branch below *Root,* corresponding to the test A = vi
    - Let $Examples_{vi}$ be the subset of *Examples* that have value vi for A
    - If $Examples_{vi}$ is empty
      - Then below this new branch add a leaf node with label = most common value of *Target_attribute* *in Examples*
    - Else
      - below this new branch add the subtree
      - *ID3(*$Examples_{vi}$*,Target_attribute, Attributes - {A}))*
- End
- Return *Root*

Q61. What is gini impurity and gini index?



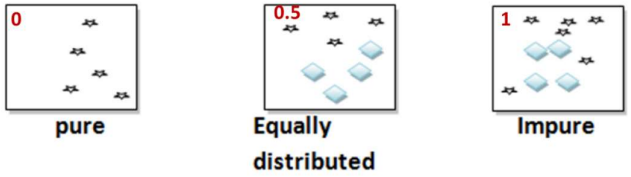
- **Gini Impurity:**
  - Measures how often a randomly chosen example would be incorrectly labeled if it was randomly labeled according to the label distribution.
- **Gini Index as a Cost Function:**
  - Serves as a cost function used to evaluate splits in the dataset. It favors larger partitions and is easy to implement, while information gain favors smaller partitions with distinct values.

In the context of decision trees used in machine learning, the Gini index is a metric that measures the impurity or homogeneity of a node in a decision tree. It's one of the methods used to evaluate which feature to split on at each node in the tree-building process.

The Gini index is used in classification algorithms, particularly in decision trees and ensemble methods like Random Forests, to determine the best split for partitioning the data based on categorical target variables.

Q62. How gini index is calculated?

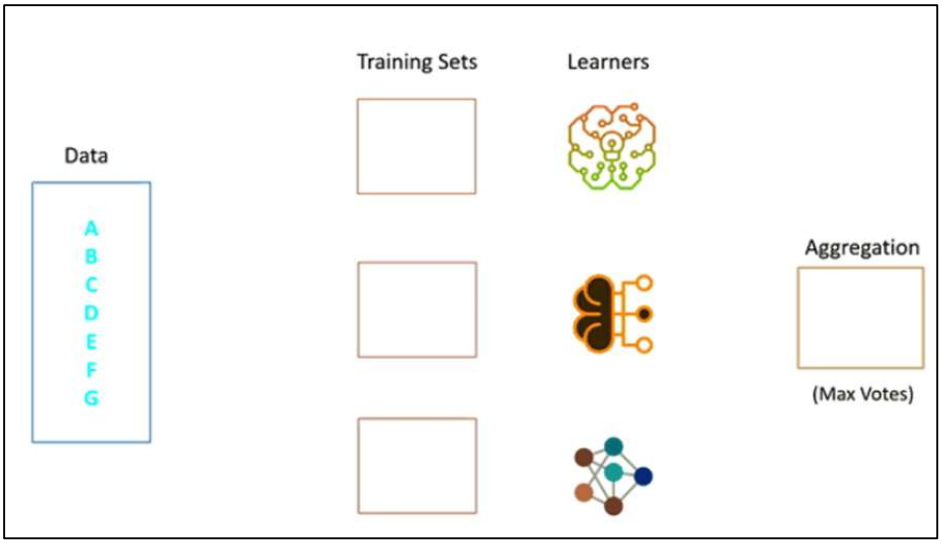
- It is calculated by subtracting the sum of the squared probabilities of each class from one.
- It favors larger partitions and easy to implement whereas information gain favors smaller partitions with distinct values.
- Gini Index works with the categorical target variable “Success” or “Failure”. It performs only Binary splits.
- Calculate Gini index using: $\text{Gini Index} = 1 - \sum_j P_j^2$ (Probability of each class)
- If all the elements are linked with a single class then it is called pure. It ranges from 0-1

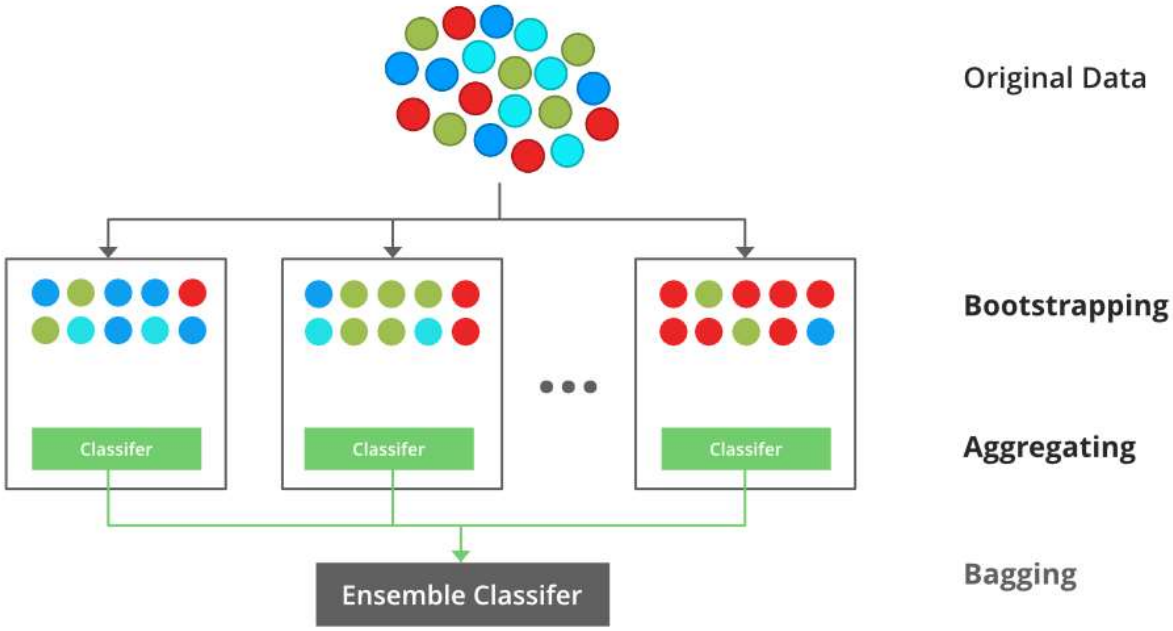


Note: An attribute with a lower Gini index should be preferred.

Q63. What is Bagging?

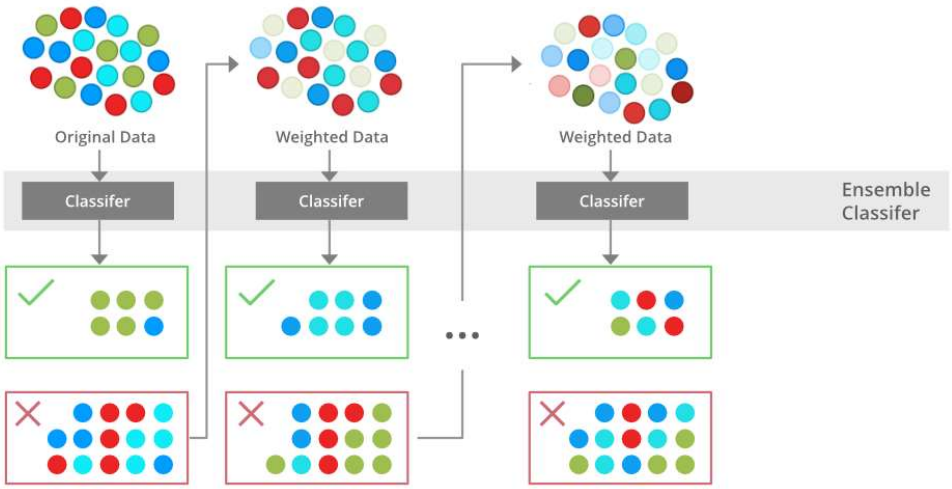
- **Bagging in Supervised Machine Learning:**
  - Primarily used to address supervised machine learning problems, bagging aims to reduce variance and prevent overfitting.
- **Bagging is typically completed in two steps:**
  - **Bootstrapping:** Utilizes random sampling with replacement to derive data samples for the primary model. Random data samples are fed to the primary model, and a base learning algorithm is run on the samples to facilitate the learning process.
  - **Aggregation:** Involves combining the output of all base models. Based on their outputs, an aggregate result is predicted, leading to greater accuracy and reduced variance.

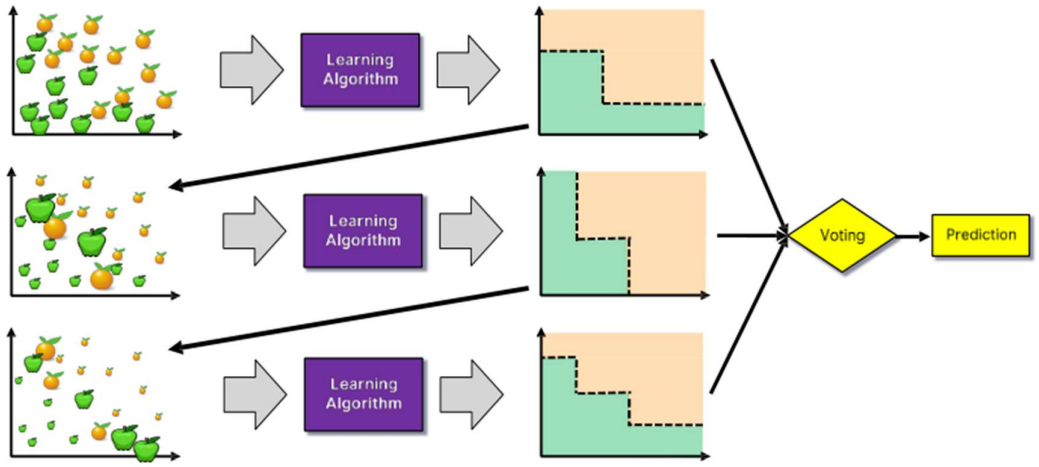




Q64. What is boosting?

- **Boosting as an Ensemble Method:**
  - Boosting is an ensemble method designed to enable each member to learn from the preceding member's mistakes, enhancing predictions for future instances.
- **Sequential Arrangement of Base Learners:**
  - In boosting, all base learners (weak) are arranged in a sequential format, allowing them to learn from the mistakes of their preceding learner.
- **Transformation of Weak to Strong Learners:**
  - Through this sequential learning process, all weak learners are transformed into strong learners, contributing to the creation of a better predictive model with significantly improved performance.



- 

Q65. Write the difference b/w bagging and boosting

Bagging	Boosting
Various training data subsets are randomly drawn with replacement from the whole training dataset.	Each new subset contains the components that were misclassified by previous models.
Bagging attempts to tackle the over-fitting issue.	Boosting tries to reduce bias.
If the classifier is unstable (high variance), then we need to apply bagging.	If the classifier is steady and straightforward (high bias), then we need to apply boosting.
Every model receives an equal weight.	Models are weighted by their performance.
Objective to decrease variance, not bias.	Objective to decrease bias, not variance.
It is the easiest way of connecting predictions that belong to the same type.	It is a way of connecting predictions that belong to the different types.
Every model is constructed independently.	New models are affected by the performance of the previously developed model.

S.NO	Bagging	Boosting
1.	The simplest way of combining predictions that belong to the same type.	A way of combining predictions that belong to the different types.
2.	Aim to decrease variance, not bias.	Aim to decrease bias, not variance.
3.	Each model receives equal weight.	Models are weighted according to their performance.
4.	Each model is built independently.	New models are influenced by the performance of previously built models.
5.	Different training data subsets are selected using row sampling with replacement and random sampling methods from the entire training dataset.	Every new subset contains the elements that were misclassified by previous models.
6.	Bagging tries to solve the over-fitting problem.	Boosting tries to reduce bias.
7.	If the classifier is unstable (high variance), then apply bagging.	If the classifier is stable and simple (high bias) the apply boosting.
8.	In this base classifiers are trained parallelly.	In this base classifiers are trained sequentially.
9	Example: The Random forest model uses Bagging.	Example: The AdaBoost uses Boosting techniques

Q66. What is Random Forest and how do we obtain slightly different trees?

Random Forest is an ensemble learning method that combines the predictions of multiple individual decision trees to create a more accurate and robust model. It was introduced by Leo Breiman and Adele Cutler.

The key idea behind Random Forest is to introduce randomness both in the data and the features used for building the trees, which helps to reduce overfitting and enhance generalization.

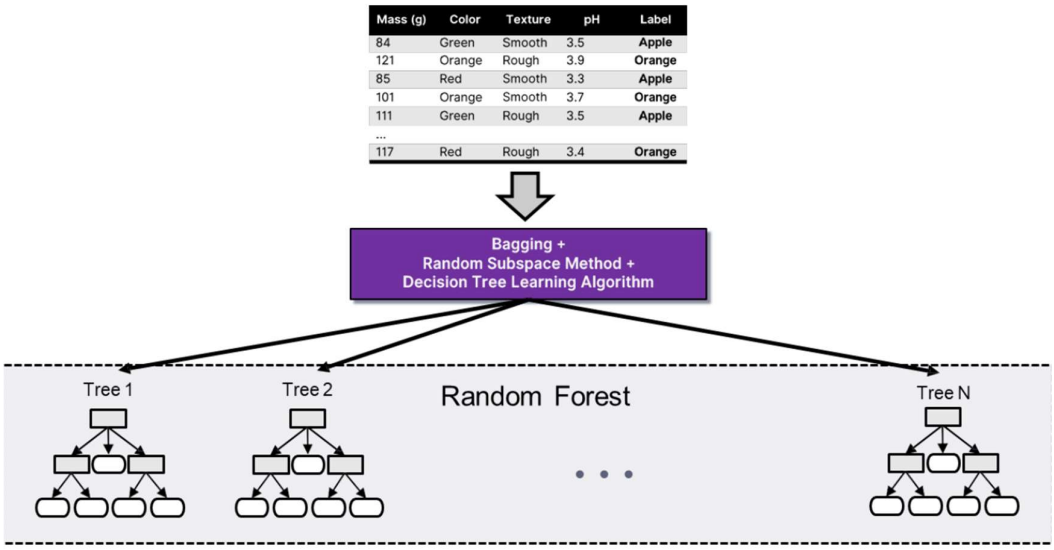
To obtain slightly different trees from a single dataset, two common techniques are employed:

1. **Bagging (Bootstrap Aggregating):**
    - Utilize random subsets of data points from the training set to create N smaller datasets.
    - Fit a decision tree on each subset independently.
2. **Random Subspace Method (Feature Bagging):**
    - Fit N different decision trees, each constrained to operate on a random subset of features.
    - This method introduces diversity among the trees by varying the features considered during tree construction.

These techniques, Bagging and the Random Subspace Method, aim to create an ensemble of diverse trees that collectively contribute to a more robust and accurate model. The variability introduced during the creation of each tree helps mitigate overfitting and improves the overall generalization performance of the ensemble.

Random Forest algorithm

1. **Step 1**: Choose the number of trees you want in your forest (the more trees, the more robust the forest, but also the longer it will take to compute).
2. **Step 2 (Bootstrap sampling)**: For each tree, select a random subset of the training data with replacement. This process is known as bootstrapping. This means that some samples will be used multiple times in a single tree.
3. **Step 3 (Random subspace)**: Grow the decision tree. At each node:
    - Randomly select a subset of features. This is the random subspace method. It ensures that each split decision is made based on a subset of features, not all of them.
    - Split the node by finding the feature that provides the best split according to the objective function, for instance, by maximizing the information gain.
4. **Step 4**: Repeat steps 2 and 3 until you have a forest of trees.
5. **Step 5**: To classify a new object, take the majority vote of the ensemble.



Q67. What are Temporally aware distance metrics?

Temporally aware distance metrics consider the temporal relationship between data points, in contrast to traditional distance metrics that do not account for temporal dynamics. Several notable temporally aware distance metrics have been proposed:

1. **Dynamic Time Warping (DTW):**
    - *Definition:* DTW is a dynamic programming algorithm that calculates the distance between two sequences by warping the time axis.
    - *Key Feature:* It accommodates misalignment between sequences, allowing for non-perfect temporal alignment.
2. **Elastic Distance:**
    - *Definition:* Elastic Distance is a generalization of DTW, providing more flexibility in the warping of the time axis.
    - *Key Feature:* Offers increased flexibility in capturing temporal relationships by allowing elastic warping.
3. **Time Warped Euclidean Distance (TWED):**
    - *Definition:* TWED is a distance metric based on DTW, but it employs Euclidean distance instead of a dynamic programming algorithm.
    - *Key Feature:* Utilizes Euclidean distance while considering temporal warping between sequences.
4. **Shapelet Distance:**
    - *Definition:* Shapelet Distance employs shapelets, which are small subsequences representing the overall shape of the data points.
    - *Key Feature:* Uses shapelets to capture representative subsequences, emphasizing the shape of the data points.

These temporally aware distance metrics play a crucial role in time series analysis by capturing temporal dynamics and allowing for more accurate comparisons between sequences. The choice

of a specific metric depends on the characteristics of the data and the desired level of sensitivity to temporal variations.

Q68. What can Temporally aware distance metrics be used for?

Applications of Temporally Aware Distance Metrics in Time Series Analysis:

1. **Time Series Classification:**
    - *Description:* Temporally aware distance metrics, such as Dynamic Time Warping (DTW), can be employed to classify time series data into distinct categories. For instance, DTW can be used to differentiate events like heartbeats or speech patterns.
    - *Example:* Utilize DTW to measure the distance between time series instances, enabling accurate classification of diverse events within the data.
2. **Time Series Clustering:**
    - *Description:* Temporally aware distance metrics are effective for clustering time series data based on temporal relationships. DTW, for instance, can facilitate the grouping of similar time series patterns.
    - *Example:* Apply DTW to cluster time series data into groups with similar temporal patterns, aiding in the identification of distinct patterns or behaviors.
3. **Time Series Similarity Search:**
    - *Description:* Temporally aware distance metrics play a crucial role in searching for similar time series instances. DTW, as an example, enables the identification of time series that closely resemble a given reference series.
    - *Example:* Employ DTW to perform similarity searches, finding all time series that exhibit patterns closely resembling a specified time series.

These applications demonstrate the versatility of temporally aware distance metrics in various aspects of time series analysis. Whether for classification, clustering, or similarity search, these metrics enhance the accuracy and effectiveness of analytical tasks by considering the temporal relationships within time series data.

Q69. What is Dynamic Time Warping (DTW) in Time Series Analysis?

Dynamic Time Warping (DTW) is a widely utilized measure of similarity between two time series in the context of time series analysis. Originally designed to address automatic speech recognition, DTW has found applications in various domains where accurate alignment of temporal patterns is essential.

Key Features of DTW:

- **Optimal Global Alignment:**
    - DTW provides an optimal global alignment between two time series, considering potential temporal distortions between them. This allows for a flexible matching of corresponding points in the time series, accommodating variations in timing and pacing.

Applications and Significance:

- **Similarity Measure:**
    - DTW is a robust similarity measure that takes into account the inherent temporal distortions present in time series data. It is particularly effective when comparing sequences that may have variations in speed, duration, or phase.
- **Origin in Speech Recognition:**
    - Initially designed for automatic speech recognition, DTW has proven valuable in scenarios where traditional distance metrics might fall short due to the temporal misalignments present in the data.

Example Scenario:

- **Time Series Alignment:**
    - In a time series analysis scenario, DTW can be applied to align and compare two time series with different temporal characteristics. For instance, it can be employed to find the optimal alignment between two speech patterns, allowing for accurate comparison despite variations in speech speed or duration.
- **Flexible Matching:**
    - Consider a use case where DTW is applied to compare heart rate patterns. Due to natural variations in heart rate timing, DTW enables a flexible matching that captures the true similarity between the sequences, providing a more accurate assessment than traditional distance measures.

In summary, Dynamic Time Warping (DTW) is a valuable tool in time series analysis, offering optimal global alignment between time series with temporal distortions. Its flexibility makes it suitable for various applications, ensuring accurate similarity measures in scenarios where temporal variations are a critical consideration.

Q70. Provide the algorithm of DTW.

DTW algorithm

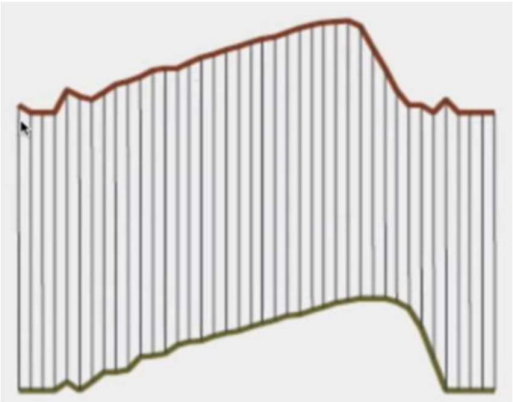
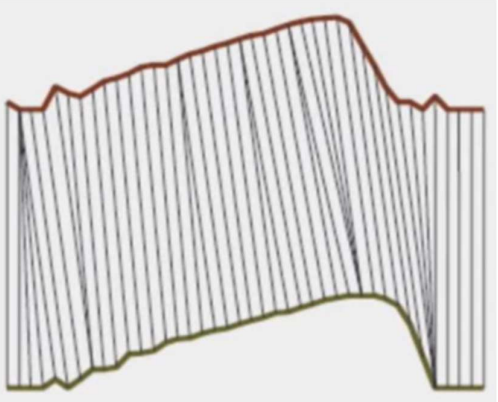
- Inputs: $x_{1:N}$ and $y_{1:M}$
- Cost matrix: $\mathbf{D} \in \mathbb{R}^{(N+1)\times M+1}$
- Initialization:
  for $i = 1$ to N:
  for $j = 1$ to M:
- Calculate cost matrix:
  for $i = 1$ to N:
  for $j = 1$ to M:

$$D_{i,j} = d(x_i, y_j) + \min \begin{cases} D_{i-1,j-1} & \text{(match)} \\ D_{i-1,j} & \text{(insertion)} \\ D_{i,j-1} & \text{(deletion)} \end{cases}$$

- Get alignment: Trace back from $D_{N,M}$ to $D_{0,0}$

Q71. Give the difference b/w Euclidean distance and DTW?

Feature	Euclidean Distance	Dynamic Time Warping (DTW)
Scope of Application	Suitable for aligned time series data with uniform sampling intervals.	Effective for comparing time series with temporal distortions and misalignments.
Sensitivity to Time Distortions	Insensitive to temporal misalignments; assumes one-to-one correspondence between points.	Sensitive to temporal distortions; allows for flexible alignment of time series.
Alignment Mechanism	Straightforward point-to-point alignment.	Allows for optimal global alignment, considering potential warping of the time axis.
Use Cases	Appropriate for scenarios where time series exhibit consistent timing and pacing.	Beneficial in scenarios with variable speeds, durations, or phases in time series.
Applications	Commonly used when time series have a consistent temporal structure.	Widely applied in speech recognition, gesture matching, and any domain with varying temporal patterns.
Mathematical Formulation	$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$	Dynamic programming approach, considering optimal warping of the time axis.
Computational Complexity	Lower computational complexity.	Higher computational complexity, especially for longer time series.

Feature	Euclidean Distance	Dynamic Time Warping (DTW)
Robustness to Variability	Less robust when dealing with time series variations and misalignments.	More robust in scenarios with temporal variability, allowing for accurate similarity measures.
Distance Measure Interpretation	Provides a direct geometric interpretation.	Requires a more nuanced interpretation due to the dynamic programming alignment.
Example		

Q72. What is Backpropagation? Give its key points.

Backpropagation is an optimization algorithm utilized in the training of neural networks. It operates by iteratively adjusting the model's parameters, including weights and biases, based on the computed gradients of the cost function. The algorithm employs the chain rule of calculus to propagate errors backward through the network, facilitating the minimization of the cost function.

Key Points:

1. **Chain Rule Method:**
    - Backpropagation leverages the chain rule of calculus to compute the gradients of the cost function with respect to the network's parameters (weights and biases).
    - It involves a systematic method of calculating how changes in each parameter contribute to the overall change in the cost function.
2. **Forward and Backward Pass:**
    - The algorithm comprises both a forward pass, where the input propagates through the network to generate predictions, and a backward pass, where adjustments to model parameters are made based on computed errors.
    - After each forward pass, backpropagation performs a backward pass, iteratively refining the model's parameters.
3. **Minimization of Cost Function:**
    - Backpropagation aims to minimize the cost function, representing the disparity between predicted and true outputs.
    - The adjustment of network weights and biases during training contributes to the reduction of this cost.

4. **Adjustment Determined by Gradients:**
    - The extent of adjustment to the model's parameters is determined by the gradients of the cost function with respect to those parameters.
    - Gradients indicate the direction and magnitude of change needed to minimize the cost, guiding the iterative optimization process.

In essence, backpropagation is a fundamental algorithm in neural network training, enabling the model to learn from data by iteratively refining its parameters. Through a combination of forward and backward passes, the algorithm systematically adjusts weights and biases to minimize the cost function, ultimately enhancing the model's predictive capabilities.

Q73. Give the algorithm for backpropagation.

1. **Initialize:**
    - Set the initial weights and biases randomly or using predefined values.
    - Specify the learning rate (η) and the number of training epochs.
2. **Forward Pass:**
    - Input a training sample into the network.
    - For each layer, calculate the weighted sum (z) and the output after activation (a) using the current weights and biases.
3. **Compute Error:**
    - Calculate the error (δ) between the predicted output and the true output using a specified cost function.
4. **Backward Pass:**
    - Starting from the output layer:

      a. Compute the gradient of the cost function with respect to the weighted sum ($\frac{\partial E}{\partial z}$).

      b. Update the weights and biases using the gradient and the learning rate:

      $w_{ij} \text{ new} = w_{ij} \text{ old} - \eta \frac{\partial E}{\partial w_{ij}}$

      $b_j \text{ new} = b_j \text{ old} - \eta \frac{\partial E}{\partial b_j}$
    - Propagate the error backward to hidden layers:

      a. Compute the gradient of the cost function with respect to the weighted sum.

      b. Update the weights and biases for each layer.
5. **Repeat:**
    - Repeat steps 2-4 for each training sample in the dataset for a specified number of epochs.
6. **Training Completion:**
    - Once the specified number of epochs is reached or convergence is achieved, the neural network is considered trained.

Q74. What is gradient descent and what are its types with their pros and cons?

Gradient Descent is an optimization algorithm used to minimize a function iteratively. It is particularly prevalent in machine learning, where the goal is to find the minimum of a cost function by adjusting model parameters. The "gradient" refers to the partial derivatives of the function with respect to its parameters.

Basic Idea: The algorithm starts with an initial set of parameters and moves toward the minimum of the cost function by iteratively adjusting the parameters in the direction opposite to the gradient. The learning rate determines the size of each step, and the process continues until convergence or a specified number of iterations.

Types of Gradient Descent:

1. **Batch Gradient Descent:**
    - **Description:** Considers the entire training dataset for each iteration.
    - **Pros:** Stable convergence, particularly with smooth and convex cost functions.
    - **Cons:** Computationally expensive for large datasets.
2. **Stochastic Gradient Descent (SGD):**
    - **Description:** Randomly selects one data point for each iteration.
    - **Pros:** Faster iteration, suitable for large datasets.
    - **Cons:** High variance in the update direction due to single-sample selection.
3. **Mini-Batch Gradient Descent:**
    - **Description:** Uses a randomly selected subset (mini-batch) of the training data for each iteration.
    - **Pros:** Combines advantages of batch and stochastic gradient descent.
    - **Cons:** Requires tuning the batch size.
4. **Gradient Descent with Momentum:**
    - **Description:** Introduces a momentum term to accelerate convergence by taking into account the moving average of past gradients.
    - **Pros:** Helps navigate through shallow plateaus and noisy gradients.
    - **Cons:** Requires tuning of momentum hyperparameter.
5. **Adagrad (Adaptive Gradient Algorithm):**
    - **Description:** Adjusts the learning rate for each parameter based on the historical gradient information.
    - **Pros:** Adapts learning rates to parameters, suitable for sparse data.
    - **Cons:** Learning rates can become too small, causing slow convergence.
6. **RMSprop (Root Mean Square Propagation):**
    - **Description:** Improves Adagrad by using a moving average of squared gradients.
    - **Pros:** Mitigates Adagrad's diminishing learning rates.

- **Cons:** Requires tuning hyperparameters.

7. **Adam (Adaptive Moment Estimation):**
    - **Description:** Combines ideas from Momentum and RMSprop, incorporating both first-order momentum and second-order scaling of the gradients.
    - **Pros:** Efficient and widely used in practice.
    - **Cons:** Requires tuning hyperparameters.

Q75. Why computing gradients?

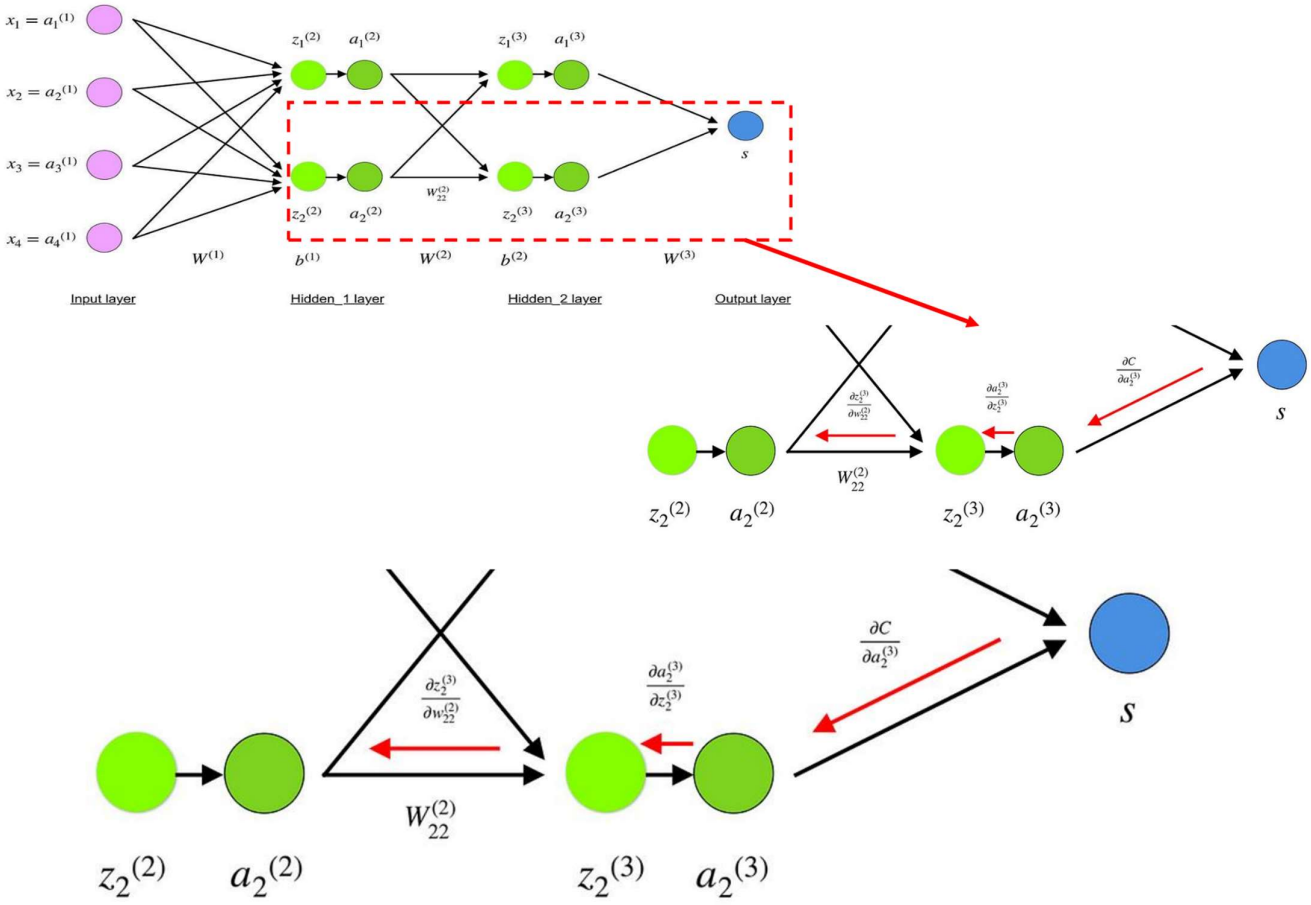
- **Gradient Definition:**
    - The gradient of a function (1,2,…,) $C(x_1,x_2,…,x_m)$ in point x is a vector comprising the partial derivatives of C with respect to each of its input variables at point x.
- **Derivative Equation:**
    - The equation for the derivative of C in x is expressed as a vector of partial derivatives.

$$\frac{\partial C}{\partial x} = [\frac{\partial C}{\partial x_1}, \frac{\partial C}{\partial x_2}, \ldots, \frac{\partial C}{\partial x_m}]$$

- **Sensitivity to Change:**
    - The derivative of function C with respect to x measures the sensitivity of the function's output value to a change in its input value. It signifies the direction in which C is changing.
- **Gradient Interpretation:**
    - The gradient provides information on how much the parameter x needs to change (either positively or negatively) to minimize the function C.
- **Chain Rule Technique:**
    - The computation of these gradients involves the use of a technique known as the chain rule in calculus. The chain rule facilitates the calculation of derivatives for composite functions.

Understanding the gradient and its derivatives is crucial in optimization algorithms, such as gradient descent, where the goal is to iteratively adjust parameters to minimize a cost function. The gradient provides insights into the direction and magnitude of changes needed to optimize the function.

Q76. Calculate the gradient of C with respect to a single weight $(W22)^2$?



- Weight $(W_{22})^2$ connects $(a_2)^2$ and $(Z_2)^2$, so computing the gradient requires applying the chain rule through $(Z_2)^3$ and $(a_2)^3$:

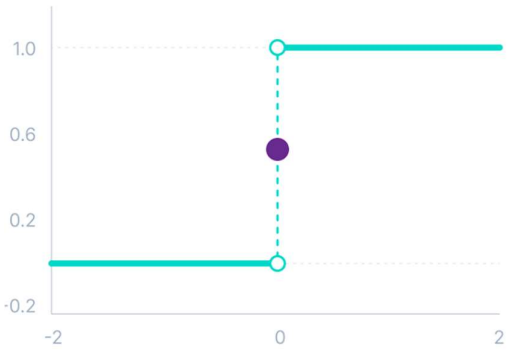
$$\frac{\partial C}{\partial w_{22}^{(2)}} = \frac{\partial C}{\partial z_2^{(3)}} \cdot \frac{\partial z_2^{(3)}}{\partial w_{22}^{(2)}} = \frac{\partial C}{\partial a_2^{(3)}} \cdot \frac{\partial a_2^{(3)}}{\partial z_2^{(3)}} \cdot a_2^{(2)} = \frac{\partial C}{\partial a_2^{(3)}} \cdot f'(z_2^{(3)}) \cdot a_2^{(2)}$$

- Equation for derivative of C in $(W_{22})^2$
- Calculating the final value of derivative of *C* in $(a_2)^3$ requires knowledge of the function *C*. Since *C* is dependent on $(a_2)^3$,

Q77. Provide a list of activation functions.

1. **Binary Step Function**

   Binary step function depends on a threshold value that decides whether a neuron should be activated or not.



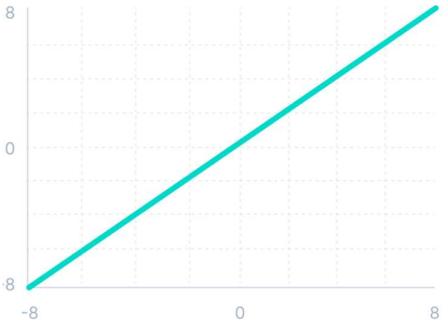
$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$$

   Limitations of binary step function:

- It cannot provide multi-value outputs—for example, it cannot be used for multi-class classification problems.
- The gradient of the step function is zero, which causes a hindrance in the backpropagation process.

2. **Linear Activation Function**

The linear activation function, also known as "no activation," or "identity function" (multiplied x1.0), is where the activation is proportional to the input.



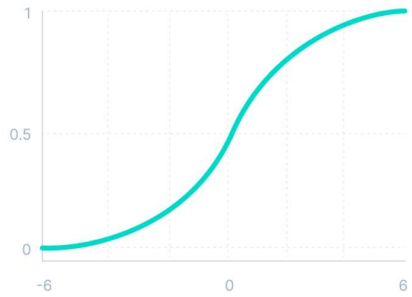
$$f(x) = x$$

linear activation function has two major problems :

- It's not possible to use backpropagation as the derivative of the function is a constant and has no relation to the input x.
- All layers of the neural network will collapse into one if a linear activation function is used. No matter the number of layers in the neural network, the last layer will still be a linear function of the first layer. So, essentially, a linear activation function turns the neural network into just one layer.

3. **Sigmoid / Logistic Activation Function**

This function takes any real value as input and outputs values in the range of 0 to 1.



$$f(x) = \frac{1}{1 + e^{-x}}$$

Advantages of Sigmoid/Logistic Activation Function:

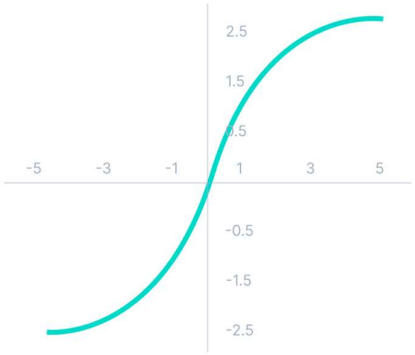
- Probability Prediction: Suitable for models predicting probabilities due to its output range between 0 and 1.
- Differentiability: Differentiable and smooth gradient prevent abrupt jumps in output values, ensuring stable learning.

Limitations of Sigmoid/Logistic Activation Function:

- Vanishing Gradient Problem: Limited significant gradients in the range -3 to 3 can lead to the vanishing gradient problem, hindering effective learning.
- Asymmetry Around Zero: Non-symmetry around zero makes training neural networks more challenging and less stable.

4. **Tanh Function (Hyperbolic Tangent)**

Tanh function is very similar to the sigmoid/logistic activation function, and even has the same S-shape with the difference in output range of -1 to 1. In Tanh, the larger the input (more positive), the closer the output value will be to 1.0, whereas the smaller the input (more negative), the closer the output will be to -1.0.



$$f(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$$

Advantages of using this activation function are:

- The output of the tanh activation function is Zero centered; hence we can easily map the output values as strongly negative, neutral, or strongly positive.
- Usually used in hidden layers of a neural network as its values lie between -1 to; therefore, the mean for the hidden layer comes out to be 0 or very close to it. It helps in centering the data and makes learning for the next layer much easier.

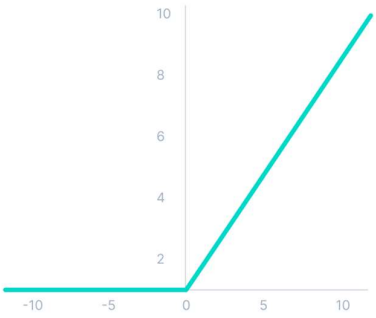
Limitations:

- It also faces the problem of vanishing gradients similar to the sigmoid activation function.
- Plus, the gradient of the tanh function is much steeper as compared to the sigmoid function.

5. **ReLU Function**

ReLU stands for Rectified Linear Unit.

Although it gives an impression of a linear function, ReLU has a derivative function and allows for backpropagation while simultaneously making it computationally efficient.



$$f(x) = max\,(0, x)$$

Advantages of using ReLU as an activation function are as follows:

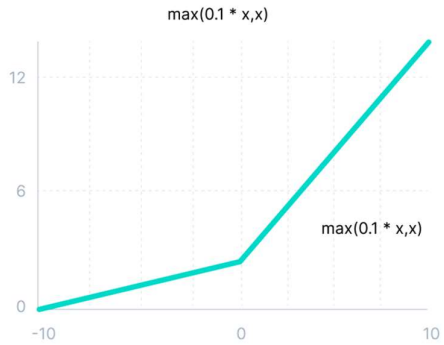
- Since only a certain number of neurons are activated, the ReLU function is far more computationally efficient when compared to the sigmoid and tanh functions.
- ReLU accelerates the convergence of gradient descent towards the global minimum of the loss function due to its linear, non-saturating property.

Limitations:

- The Dying ReLU problem
- All the negative input values become zero immediately, which decreases the model's ability to fit or train from the data properly.

6. Leaky ReLU Function

Leaky ReLU is an improved version of ReLU function to solve the Dying ReLU problem as it has a small positive slope in the negative area.



$$f(x) = max\ (0.1x, x)$$

Advantages:

- same as that of ReLU
- it does enable backpropagation, even for negative input values.

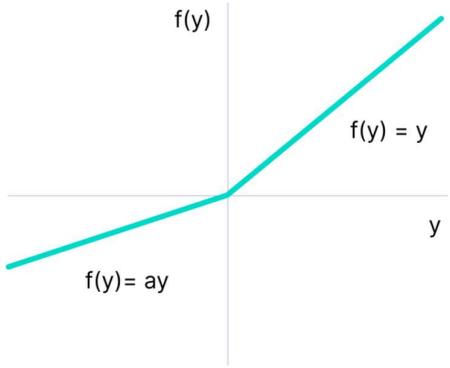
The limitations that this function faces include:

- The predictions may not be consistent for negative input values.
- The gradient for negative values is a small value that makes the learning of model parameters time-consuming.

7. Parametric ReLU Function

Parametric ReLU is another variant of ReLU that aims to solve the problem of gradient's becoming zero for the left half of the axis.

This function provides the slope of the negative part of the function as an argument a. By performing backpropagation, the most appropriate value of a is learnt.



$$f(x) = max\ (ax, x)$$

Where *"a"* is the slope parameter for negative values.

Advantage:

- The parameterized ReLU function is used when the leaky ReLU function still fails at solving the problem of dead neurons, and the relevant information is not successfully passed to the next layer.

Limitation:

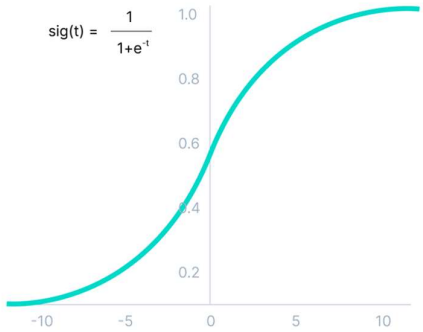
- It may perform differently for different problems depending upon the value of slope parameter ***a.***

8. **SoftMax Function**

The output of the sigmoid function was in the range of 0 to 1, which can be thought of as probability.

The Softmax function is described as a combination of multiple sigmoids.

It calculates the relative probabilities. Similar to the sigmoid/logistic activation function, the SoftMax function returns the probability of each class.



$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

Pros:

- Useful for multi-class classification problems as it provides probabilities for each class.

Cons:

- Computationally expensive due to the summation in the denominator.

Q78. How Non- stationary time series are related to random walks and unit roots?

- **Random Walks and Non-Stationarity:**
  - Random walks are inherently non-stationary time series. In a random walk, the current value is the sum of the previous value and a random noise term. This dynamic leads to changing mean and variance over time, rendering the random walk non-stationary.
- **Unit Roots and Non-Stationarity:**
  - Unit roots are a characteristic of certain stochastic processes, including random walks. A unit root in the characteristic equation of a linear stochastic process signifies non-stationarity. Specifically, if 1 is a root of the characteristic equation, the process is non-stationary. However, it may not always exhibit a trend. Stationarity is achieved by differencing the process, and the number of differences required depends on the location of other roots relative to the unit circle.
- **Transforming Non-Stationary Time Series:**
  - Non-stationary time series, including those with unit roots, can be transformed into stationary series by differencing. Differencing involves computing the difference between the current and previous values of the series. This process effectively removes trends and seasonality, making the series stationary and suitable for various time series analyses.

Understanding the connection between random walks, unit roots, and non-stationarity is crucial in time series analysis, as it informs the choice of appropriate transformations and methods to achieve stationarity for improved modeling and statistical inference.

Q79. Explain Dickey Fuller Test.

Dickey-Fuller Test

Definition

$y_t = \phi y_{t-1} + u_t$

- The test examines the value of ϕ. In particular, it tests the null hypothesis that $\phi = 1$ against the alternative that $\phi < 1$. In practice, the test implores the use of the differenced form.

$H_0: \phi = 1$ (Non-stationary)

$H_a: \phi < 1$ (Stationary Series)

- $\Delta y_t = \psi y_{t-1} + u_t$
- We derive this by using the first $AR(1)$ ($y_t = \phi y_{t-1} + u_t$) and it's immediate lag. If we subtract the immediate lag of the y_t, i.e. y_{t-1} to the both sides of the equation, we are essentially getting this difference equation.
- In this case, $\psi = \phi - 1$

- Likewise, the test can be extended further to accommodate the inclusion of an intercept and a deterministic time trend.
  - $\Delta y_t = \psi y_{t-1} + \mu + \beta t + u_t$
- As with the base difference equation, the null and alternative hypothesis are formulated in the manner
  - $H_0 : \psi = 0$
  - $H_a : \psi \neq 0$

$\psi = \phi - 1$

if $H_0: \psi = 0$

$0 = \phi - 1$

$\phi = 1$ (N.S)

if $H_a: \psi \neq 0$

$-0.5 = \phi - 1$

$0.5 = \phi$ (S)

Q80. Explain Augmented Dickey Fuller Test.

- So far, the Dickey Fuller test assumes that u_t is a white noise error term. However, if u_t is autocorrelated, we would need a drift version of the test which allows for higher order lags.
- Running the original Dickey Full test in this case would result in an oversized test suggesting that the true size of the test which is the proportion of times a correct null hypothesis is incorrectly rejected would be higher than the normal sized used.
- As such, we 'augment' the test using p lags of the original series
- $\Delta y_t = \psi y_{t-1} + \mu + \alpha t + \sum_{i=1}^{p} \beta_i \Delta y_{t-i} + u_t$

Q81. Explain KPSS Test.

- The test assumes that the time series can be divided into a deterministic trend, a random walk and a stationary error. This means time series can be illustrated as :

$$Y_t = \beta t + (r_t + \alpha) + e_t$$

where:

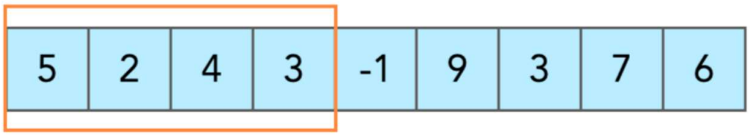
  - $r_t = r_{t-1} + u_t$ is a random walk, the initial value $r_0 = \alpha$ serves as an intercept,
  - t is the time index,
  - u_t are independent identically distributed $(0, \sigma_u^2)$.

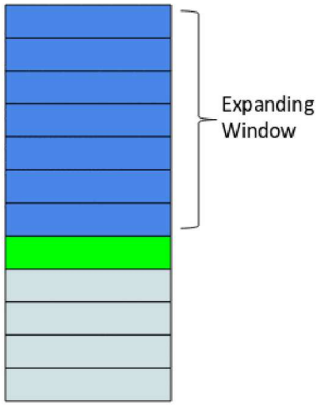
- The null and alternative hypothesis are
  - $H_0 : \sigma_u^2 = 0$
  - $H_a : \sigma_u^2 > 0$
- If it was found that $\sigma_u^2 = 0$, it means that r_t is just a constant and reduces to a trend r. Therefore, y_t is trend stationary. If however, the variance is significantly different from zero, then r varies over time suggesting that y_t is not stationary.

Q82. Explain rolling window, expanding window and custom rolling functions.

Rolling windows

A window that is sliding with every next point, the features generated using this method are called the 'rolling window' features.





Expanding window

with every step, the size of the window increases by one as it takes into account every new value in the series.

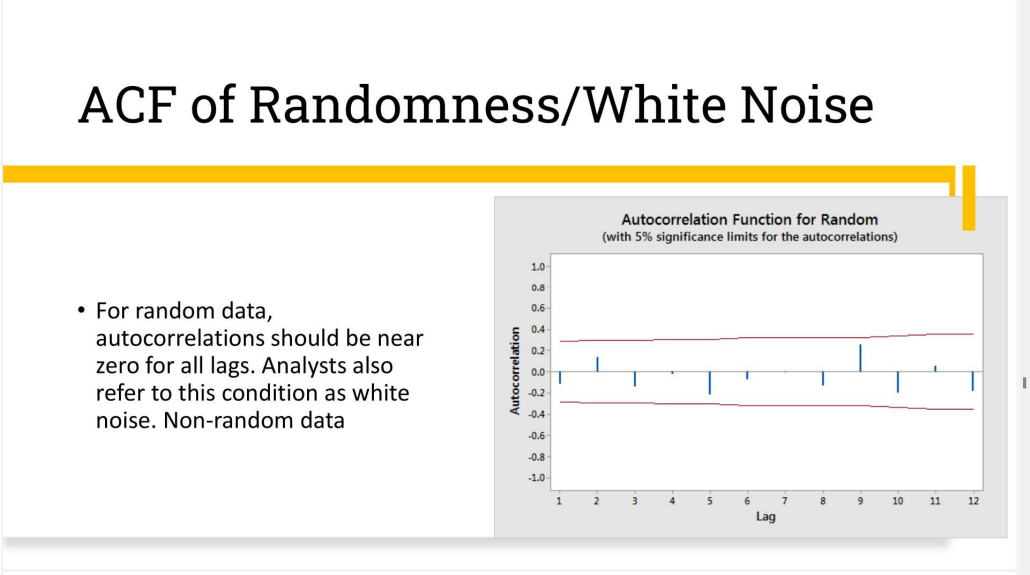
Custom rolling functions

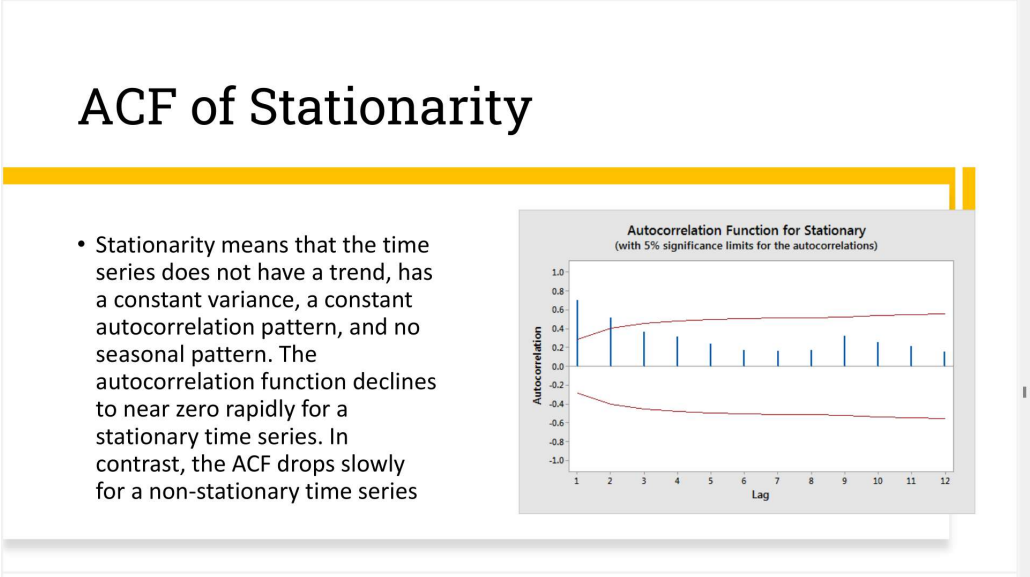
In practice, this is something you are likely to see when analyzing time series domains that have known underlying fundamental laws of behavior or useful heuristics that are necessary for proper analysis.

Q83. Describe Self correlation/Autocorrelation Function (ACF) and its plots.

Autocorrelation Function (ACF) is a statistical tool used to examine the correlation between a time series and its own past values at different lags. It helps identify significant correlations, revealing patterns and properties within the time series.

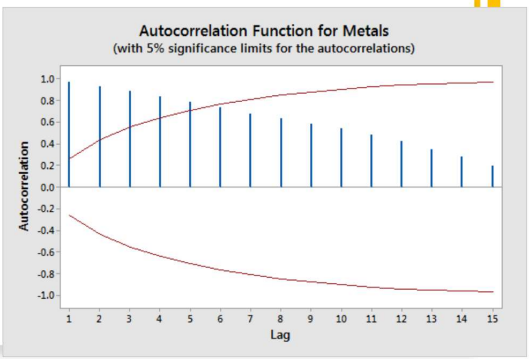
- **Purpose:** ACF is utilized to:
    - Identify significant correlations at different lags.
    - Understand patterns and properties of the time series data.
    - Inform the modeling process based on correlation information.
- **Applications:** ACF aids in assessing:
    - Randomness and stationarity of a time series.
    - Presence of trends and seasonal patterns within the data.
- **Interpretation:** In an ACF plot, each bar represents the size and direction of the correlation between the time series and its lagged values. Significant correlations are typically indicated by bars extending across a designated threshold, often represented by a red line.





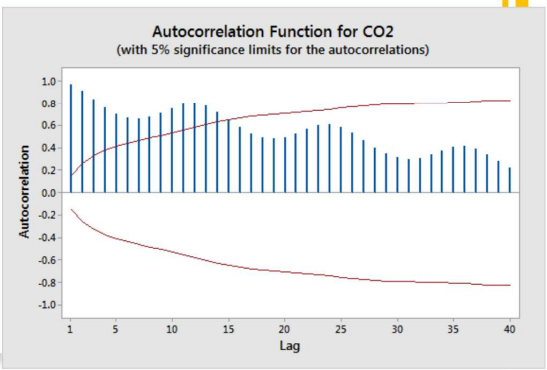
ACF of Trends

- When trends are present in a time series, shorter lags typically have large positive correlations because observations closer in time tend to have similar values. The correlations taper off slowly as the lags increase.



ACF of Seasonality

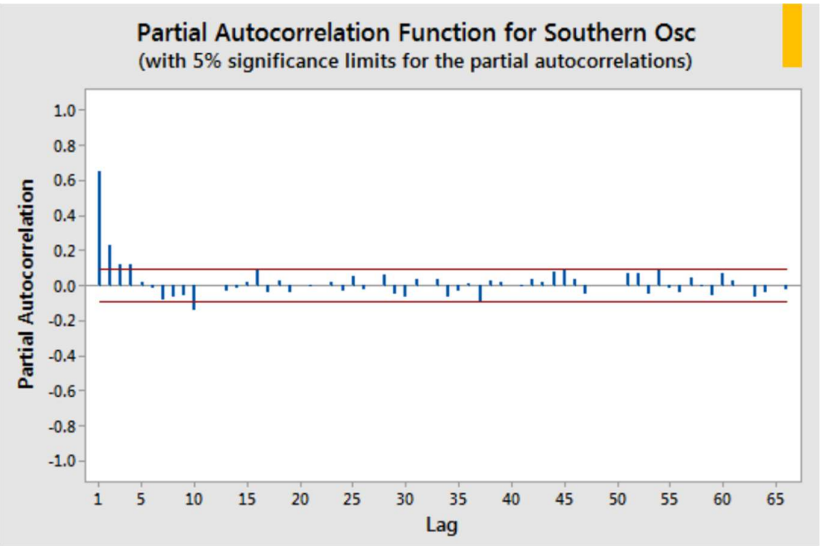
- When seasonal patterns are present, the autocorrelations are larger for lags at multiples of the seasonal frequency than for other lags..



Q84. Explain Partial auto correlation function (PACF).

-
    - The Partial Autocorrelation Function (PACF) is a statistical tool similar to the Autocorrelation Function (ACF). However, PACF specifically displays the correlation between two observations, considering only the influence of shorter lags that fall in between those observations.
- **Purpose:**
    - PACF is employed to identify and analyze the direct correlation between two observations while excluding the influence of intermediate lags. It helps reveal the unique correlation structure between specific time points in a time series.
- **Comparison with ACF:**
    - Unlike ACF, which shows the correlation between any two observations at various lags, PACF focuses on the direct correlation between two observations, discounting the impact of shorter lags.
- **Interpretation:**

- In a PACF plot, each bar represents the partial correlation between two observations at a specific lag. Significant partial correlations may indicate direct relationships that shorter lags do not adequately explain.



Example:

For example, the partial autocorrelation for lag 3 is only the correlation that lags 1 and 2 do not explain. In other words, the partial correlation for each lag is the unique correlation between those two observations after partialing out the intervening correlations.

Formula:

For a time series, the partial autocorrelation between x_t and x_{t-h} is defined as the conditional correlation between x_t and x_{t-h}, conditional on $x_{t-h+1}, \ldots, x_{t-1}$, the set of observations that come between the time points t and $t-h$.

- The 1st order partial autocorrelation will be defined to equal the 1st order autocorrelation.
- The 2nd order (lag) partial autocorrelation is

$$\frac{\text{Covariance}(x_t, x_{t-2}|x_{t-1})}{\sqrt{\text{Variance}(x_t|x_{t-1})\text{Variance}(x_{t-2}|x_{t-1})}}$$

This is the correlation between values two time periods apart conditional on knowledge of the value in between. (By the way, the two variances in the denominator will equal each other in a stationary series.)

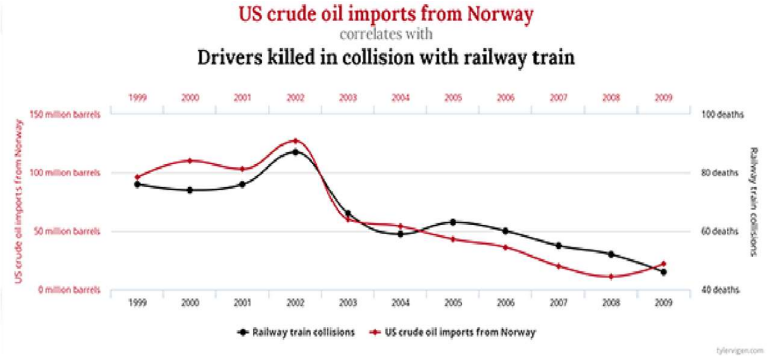
- The 3rd order (lag) partial autocorrelation is

$$\frac{\text{Covariance}(x_t, x_{t-3}|x_{t-1}, x_{t-2})}{\sqrt{\text{Variance}(x_t|x_{t-1}, x_{t-2})\text{Variance}(x_{t-3}|x_{t-1}, x_{t-2})}}$$

And, so on, for any lag.

Q85. What Causes a Spurious Correlation?

A spurious correlation occurs when two variables are correlated but don't have a causal relationship. In other words, it appears like values of one variable cause changes in the other variable, but that's not actually happening.



Spurious correlation that produces a non-zero correlation coefficient and a graph that displays a relationship.

1. Confounding Variables:

- Spurious correlation can arise when a third variable influences both of the correlated variables, creating a misleading correlation between the two. For instance, if there are two positive causal relationships:

- $A \rightarrow B$
- $A \rightarrow C$

- As variable A increases, both B and C might increase simultaneously, giving the false impression of a direct relationship between B and C.

2. Mediating Variables:

- Spurious correlation may also result from a chain of correlations involving mediating variables. For example:
  - $A \rightarrow B \rightarrow C$
- If there are causal relationships between A & B and B & C, it can create a spurious correlation between A and C.

3. Random Sampling Error:

- Random sampling error introduces variability between samples and the population due to chance. This variation can lead to the appearance of effects in the sample that do not truly exist in the broader population. Spurious correlations can be a potential outcome of random sampling error.

Q86. What is visualization in TSA?

Visualizations of varying degrees of complexity:

- **One-Dimensional Visualization:**
  - Utilized to understand the overall temporal distribution of individuals within a given time series.
- **Two-Dimensional Visualization:**
  - Employed to comprehend the typical trajectory of a value over time, especially when dealing with many parallel measurements.
- **Three-Dimensional Visualization:**
  - Involves representations where time can be expressed across as many as two dimensions or as few as none, while still being implicitly present in the visualization.

These visualizations serve to enhance the understanding of temporal patterns and relationships within time series data, catering to the specific needs and complexity of the analysis.

Q87. What is Simulating Time Series Data?

Simulation of time series data is the process of creating artificial time series data that follows a particular pattern. This can be done for a variety of purposes, such as testing forecasting algorithms, evaluating the performance of statistical models, or generating synthetic data for training machine learning models.

Different ways to simulate the time series data:

1. The **random walk model** assumes that the next value in a time series is equal to the last observation plus a random noise term. This model can be used to simulate a variety of different time series patterns, such as trends, seasonality, and cyclicality.
2. bare more complex than random walk models, but they can be used to simulate more realistic

time series patterns. For example, ARIMA models are a type of statistical model that can be used to simulate trends, seasonality, and cyclicality.

3. **Machine learning models** can be trained on real time series data to learn the patterns in the data. Once the model is trained, it can be used to generate synthetic data that follows the same patterns as the real data.

Benefits of Simulating Time Series Data:

- It can be used to test forecasting algorithms.
- It can be used to evaluate the performance of statistical models.
- It can be used to generate synthetic data for training machine learning models.
- It can be used to understand the underlying patterns in time series data.
- Simulations have lower stakes than forecasts; there are no lives and no resources on the line.

Challenges of Simulating Time Series Data:

- It can be difficult to generate realistic time series patterns.
- It can be time-consuming to simulate large amounts of time series data.
- It can be difficult to verify that the simulated data is accurate.

Q88. Difference b/w Defining requirements of Live vs Stored Data.

Factor	Live Data	Stored Data
Latency	More latency-sensitive due to constant updates.	Can tolerate longer latency, not constantly updated.
Accuracy	Typically more accurate, collected in real time.	May be outdated, but more accurate for historical analysis.
Volume	Lower volume, collected for a short period.	Higher volume, collected over a longer period.
Cost	Typically more expensive, requires additional infrastructure.	More cost-effective for long-term storage, less infrastructure.
Security	More sensitive, may contain confidential information.	May be less sensitive, less likely to contain confidential information.
Compliance	May need to comply with specific regulations.	May not need to comply with the same regulations.
Scalability	Needs to be scalable to handle large volumes of data.	May not need to be as scalable, not constantly updated.

Q89. Difference Database vs file solution for storing time series data.

Aspect	Database	Files
Advantages	- Efficient storage and retrieval.	- Simple to use and can be stored on any filesystem.
	- Well-defined schema.	- Flexible schema, allowing storage in any way.
	- Indexing, aggregation, and time series analysis.	- Efficient for large amounts of data.
Disadvantages	- More complex setup and management.	- More difficult to query and analyze.
	- Less flexible schema.	- Can be difficult to scale.
	- Not as efficient for large amounts of data.	

Databases are efficient for structured storage, retrieval, and analysis of time series data, but they require more complex setup and management.

Files, on the other hand, are simple, flexible, and efficient for large amounts of data, but can be more challenging for querying and analyzing, and may face difficulties in scaling. The choice between databases and files depends on the specific needs and characteristics of the application.

Q90. Some Popular Time series database solutions.

Time Series Database	Description	Use Case
InfluxDB	Open-source time series database designed for high-performance, high-volume data ingestion.	Ideal for applications requiring real-time data ingestion and analysis.
Prometheus	Open-source time series database designed for monitoring and alerting.	Suitable for applications needing to collect and store metrics from various sources.
TimescaleDB	Time series database built on top of PostgreSQL, suitable for storing historical data and performing complex queries.	A good choice for applications with requirements for storing historical data and conducting advanced queries.

Q91. Popular Time series File Solutions.

File Format	Description	Use Case
CSV Files	Simple way to store time series data, easy to create and read, but less efficient.	Suitable for basic applications where simplicity is valued over efficiency.
JSON Files	More flexible for storing both structured and unstructured data, less efficient.	Ideal for applications where flexibility in data storage is more crucial than performance.
Parquet Files	Binary format designed for efficient storage and querying of time series data.	Well-suited for applications requiring efficient storage and complex queries on large amounts of time series data.

Q92. What are state space models – plus and minus?

State space models, also known as dynamic linear models, are a class of mathematical models that describe the evolution of a system over time. These models consist of two main components: the state equations and the observation equations.

Advantages (Pluses) of State Space Models:

1. **Flexibility:** State space models are versatile and can be applied to a wide range of dynamic systems, making them suitable for various fields such as economics, finance, engineering, and biology.
2. **Incorporating Uncertainty:** These models naturally accommodate uncertainty by allowing the inclusion of stochastic components in both state and observation equations. This makes them suitable for modeling real-world systems with inherent randomness.
3. **Sequential Estimation:** State space models enable sequential estimation of system states, meaning that the model can be updated in real-time as new observations become available. This is particularly useful for applications where timely information is crucial.
4. **Handling Missing Data:** State space models can handle missing or irregularly sampled data, making them robust in situations where data points are not uniformly collected.
5. **Prediction and Filtering:** The framework of state space models facilitates the prediction of future states based on the available information up to the current time. Filtering algorithms, such as the Kalman filter, are commonly employed for this purpose.
6. **Parameter Estimation:** State space models allow for the estimation of both system parameters and unobserved states, providing a comprehensive understanding of the underlying system dynamics.

Disadvantages (Minuses) of State Space Models:

1. **Complexity:** Building and estimating state space models can be complex, especially when dealing with high-dimensional systems or non-linear dynamics. This complexity may increase computational demands and the risk of overfitting.

2. **Assumption Sensitivity:** Like many statistical models, state space models rely on certain assumptions, such as linearity and Gaussian noise. Deviations from these assumptions may impact the model's performance.
3. **Initialization Issues:** The accuracy of state estimates can be sensitive to the initial values assigned to the system states. Selecting appropriate initial values may require domain knowledge or additional optimization.
4. **Computational Burden:** Estimating parameters and updating state estimates sequentially can be computationally intensive, especially for large datasets. Efficient algorithms, such as the Kalman filter, are often used to mitigate this challenge.
5. **Interpretability:** Interpreting the meaning of individual states in the context of the application domain may be challenging, particularly in complex models with many hidden states.
6. **Model Misspecification:** Choosing an inappropriate model structure or assuming incorrect distributions for the system's noise components may lead to biased parameter estimates and inaccurate predictions.

Q93. Applications of Time Series in Healthcare, Financial Applications, Government Sector, Statistical Dialogue Management

Applications of Time Series Analysis in Various Sectors:

1. **Healthcare:**
    - **Patient Monitoring:** Time series analysis is crucial for monitoring patients' vital signs over time, enabling early detection of anomalies or patterns indicative of health issues.
    - **Disease Surveillance:** Analyzing temporal patterns of diseases helps in identifying outbreaks, monitoring epidemiological trends, and allocating resources effectively.
2. **Financial Applications:**
    - **Stock Market Analysis:** Time series analysis is extensively used in predicting stock prices, identifying trends, and making investment decisions.
    - **Risk Management:** Analyzing historical financial data helps in modeling and predicting risks, such as credit risk, market risk, and operational risk.
3. **Government Sector:**
    - **Economic Forecasting:** Time series models assist in forecasting economic indicators, such as GDP, inflation rates, and employment figures, aiding government policy and planning.
    - **Traffic and Transportation Management:** Analyzing time series data from traffic sensors helps optimize traffic flow, reduce congestion, and plan transportation infrastructure.
4. **Statistical Dialogue Management:**

- **Chatbot Performance Monitoring:** Time series analysis is employed to monitor and analyze the performance of chatbots and virtual assistants over time, improving user experience.
- **User Engagement Analysis:** Understanding patterns in user interactions over time helps enhance dialogue management strategies.

Example Scenario - Financial Sector: *Scenario:* Predicting Stock Prices

- **Data Collection:** Historical stock price data is collected over time.
- **Time Series Analysis:** Techniques such as ARIMA or machine learning models are applied to identify patterns and trends in the stock prices.
- **Prediction:** The model is used to predict future stock prices, aiding investors in decision-making.
- **Risk Management:** Understanding historical volatility and patterns helps in managing investment risks effectively.

Example Scenario - Healthcare: *Scenario:* Patient Monitoring

- **Data Collection:** Continuous monitoring of a patient's vital signs, such as heart rate, blood pressure, and temperature.
- **Time Series Analysis:** Analyzing temporal patterns to detect irregularities or signs of deterioration.
- **Early Intervention:** Identifying abnormal patterns triggers alerts, enabling healthcare professionals to intervene early and provide timely care.

Q94. Measuring Error- How to Test Forecasts

Measuring forecast accuracy is crucial for evaluating the performance of forecasting models. Several metrics are commonly used to test and quantify the accuracy of forecasts. Here are some widely used measures:

1. **Mean Absolute Error (MAE):**
   - **Formula:** $MAE = \frac{1}{n}\sum_{i=1}^{n}|Actual_i - Forecast_i|$
   - **Interpretation:** MAE represents the average absolute difference between the actual and forecast values. It is easy to understand but can be sensitive to outliers.
2. **Mean Squared Error (MSE):**
   - **Formula:** $MSE = \frac{1}{n}\sum_{i=1}^{n}(Actual_i - Forecast_i)^2$
   - **Interpretation:** MSE squares the differences between actual and forecast values, giving more weight to large errors. However, it may be influenced by outliers.
3. **Root Mean Squared Error (RMSE):**
   - **Formula:** $RMSE = \sqrt{MSE}$
   - **Interpretation:** RMSE is the square root of MSE, providing an interpretable measure in the same units as the original data. It penalizes large errors more than MAE.
4. **Mean Absolute Percentage Error (MAPE):**
   - **Formula:** $MAPE = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{|Actual_i - Forecast_i|}{|Actual_i|}\right) \times 100\%$
   - **Interpretation:** MAPE calculates the percentage difference between actual and forecast values. It is useful for understanding the relative accuracy of forecasts.
5. **Theil's U Statistic:**
   - **Formula:** $U = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{(Actual_i - Forecast_i)^2}{Actual_i^2} + \frac{(Actual_{i-1} - Forecast_{i-1})^2}{Actual_{i-1}^2}\right)}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{(Actual_i)^2}{n}\right) + \frac{1}{n}\sum_{i=1}^{n}\left(\frac{(Actual_{i-1})^2}{n}\right)}}$
   - **Interpretation:** Theil's U compares the forecast's performance to a naive forecast, considering both bias and variability.
6. **Forecast Bias:**
   - **Formula:** $Bias = \frac{1}{n}\sum_{i=1}^{n}(Actual_i - Forecast_i)$
   - **Interpretation:** Bias measures the systematic tendency of forecasts to be consistently higher or lower than actual values.