Unit-II

Data Analysis

Notes Compiled By : Deepak Gupta

Data Analysis

- "***Data Analysis*** *is the process of systematically applying statistical and/or logical techniques to describe and illustrate, condense and recap, and evaluate data.*"

Data analysis is a process of inspecting, cleansing, transforming and modelling data with the goal of discovering useful information, informing conclusions and supporting decision-making.

Data analysis has multiple facets and approaches, encompassing diverse techniques under a variety of names, and is used in different business, science, and social science domains.

In today's business world, data analysis plays a role in making decisions more scientific and helping businesses operate more effectively.

Univariate Analysis

➢***Univariate analysis is the simplest form of data analysis where the data being analyzed contains only one variable.***

➢Since it's a single variable it doesn't deal with causes or relationships.

➢The key objective of Univariate analysis is to simply describe the data to find patterns within the data. This is be done by looking into the mean, median, mode, dispersion, variance, range, standard deviation etc.

➢Univariate analysis is conducted through several ways which are mostly descriptive in nature:

❑Frequency Distribution Tables

❑Histograms

❑Frequency Polygons

❑Pie Charts

❑Bar Charts

Variable = X	Number = n
Boys	38
Girls	45

➢**Example:** For instance, in a survey of a class room, someone may be looking to count the number of boys and girls. In this instance, the data would simply reflect the number, i.e. a single variable.

Bivariate Analysis

- ***Bivariate analysis is the simultaneous analysis of two variables (attributes). Bivariate analysis is used to find out if there is a relationship between <u>two</u> different variables.***
- Bivariate analysis is conducted through several ways :

- **<u>Correlation coefficients</u>**

  Correlations is a statistical association technique where strength of relationship between two variables are observed. It shows the strength as strong or weak correlations and are rated on a scale of –1 to 1, where 1 is a perfect direct correlation, –1 is a perfect inverse correlation, and 0 is no correlation.

- **<u>Regression analysis</u>**

  Regression analysis is used for estimating the relationships between two different variables. It includes techniques for modelling and analysing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables.

- **<u>Example:</u>**
- Relationship between caloric intake and weight.
- Ice cream sales compared to the temperature that day.
- Traffic accidents along with the weather on a particular day.

Caloric Intake X	Weight Y
3500	250lbs
2000	225lbs
1500	110lbs
2250	145lbs
4500	380lbs

Multivariate Analysis

- ***Multivariate analysis is the simultaneous analysis of three or more variables.***

- Multivariate data analysis is a set of statistical models that examine patterns in multidimensional data by considering, at once, several data variables.

- It is an expansion of bivariate data analysis, which considers only two variables in its models.

- As multivariate models consider more variables, they can examine more complex phenomena and find data patterns that more accurately represent the real world.

- **Example:**

- A doctor has collected data on cholesterol, blood pressure, and weight. She also collected data on the eating habits of the subjects (e.g., how many ounces of red meat, fish, dairy products, and chocolate consumed per week). She wants to investigate the relationship between the three measures of health and eating habits?

- In this instance, a multivariate analysis would be required to understand the relationship of each variable with each other.

Multivariate Analysis

- Commonly used multivariate analysis technique include –
  - Factor Analysis
  - Cluster Analysis
  - Variance Analysis
  - Discriminant Analysis
  - Multidimensional Scaling
  - Principal Component Analysis
  - Redundancy Analysis

Multivariate Multiple Regression:
Multivariate regression is a technique that estimates a single regression model with more than one outcome variable. When there is more than one predictor variable in a multivariate regression model, the model is a multivariate multiple regression.

Linear Regression

- **Linear regression** is a linear approach to modelling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables).
- **A linear regression is a linear approximation of a casual relationship between two or more variables.**
- Regression allows you to estimate how a dependent variable changes as the independent variable(s) change.
- **Regression models** describe the relationship between variables by fitting a line to the observed data.

Simple Linear Regression:

Simple Linear regression is a linear approach to modelling the relationship between a scalar response (or dependent variable) and one explanatory variable (or independent variable). i.e. **Simple linear regression** is used to estimate the relationship between **two quantitative variables.**

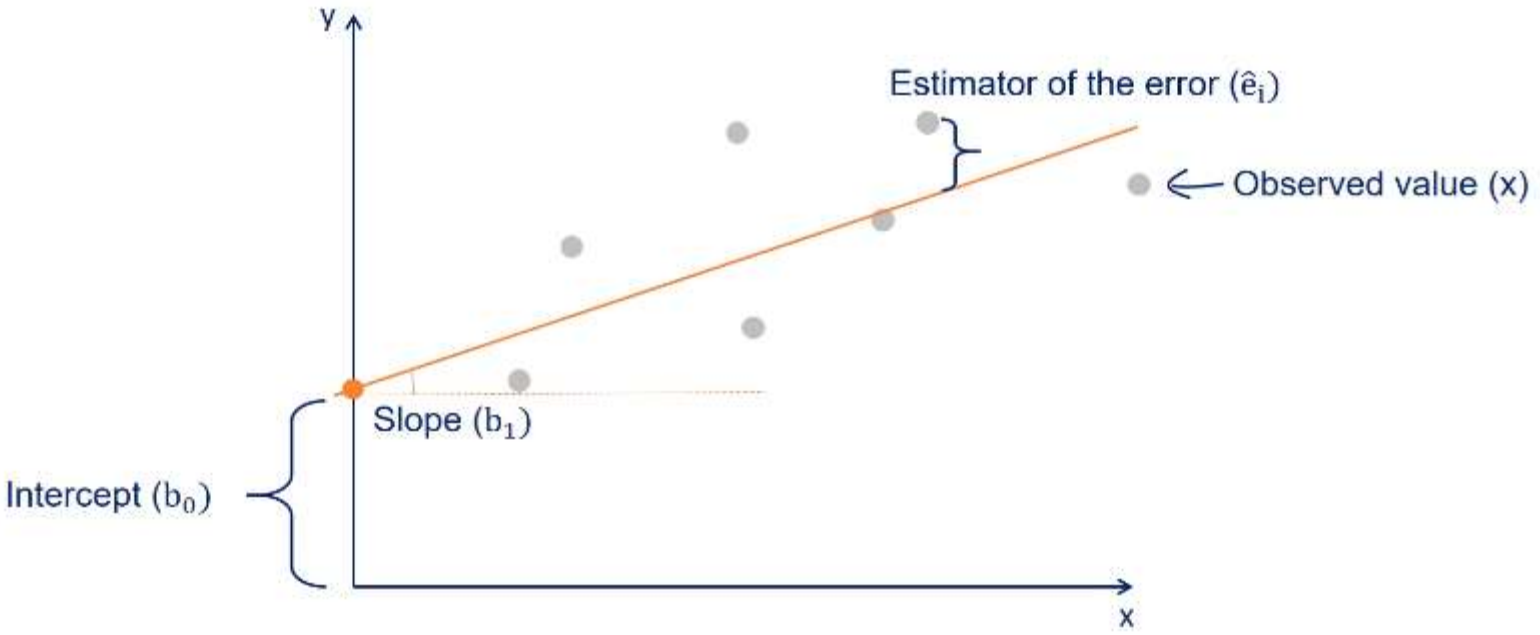
Example: If you are a social researcher interested in the relationship between income and happiness. You survey 500 people whose incomes range from $15k to $75k and ask them to rank their happiness on a scale from 1 to 10.

Your independent variable (income) and dependent variable (happiness) are both quantitative, so you can do a regression analysis to see if there is a linear relationship between them.

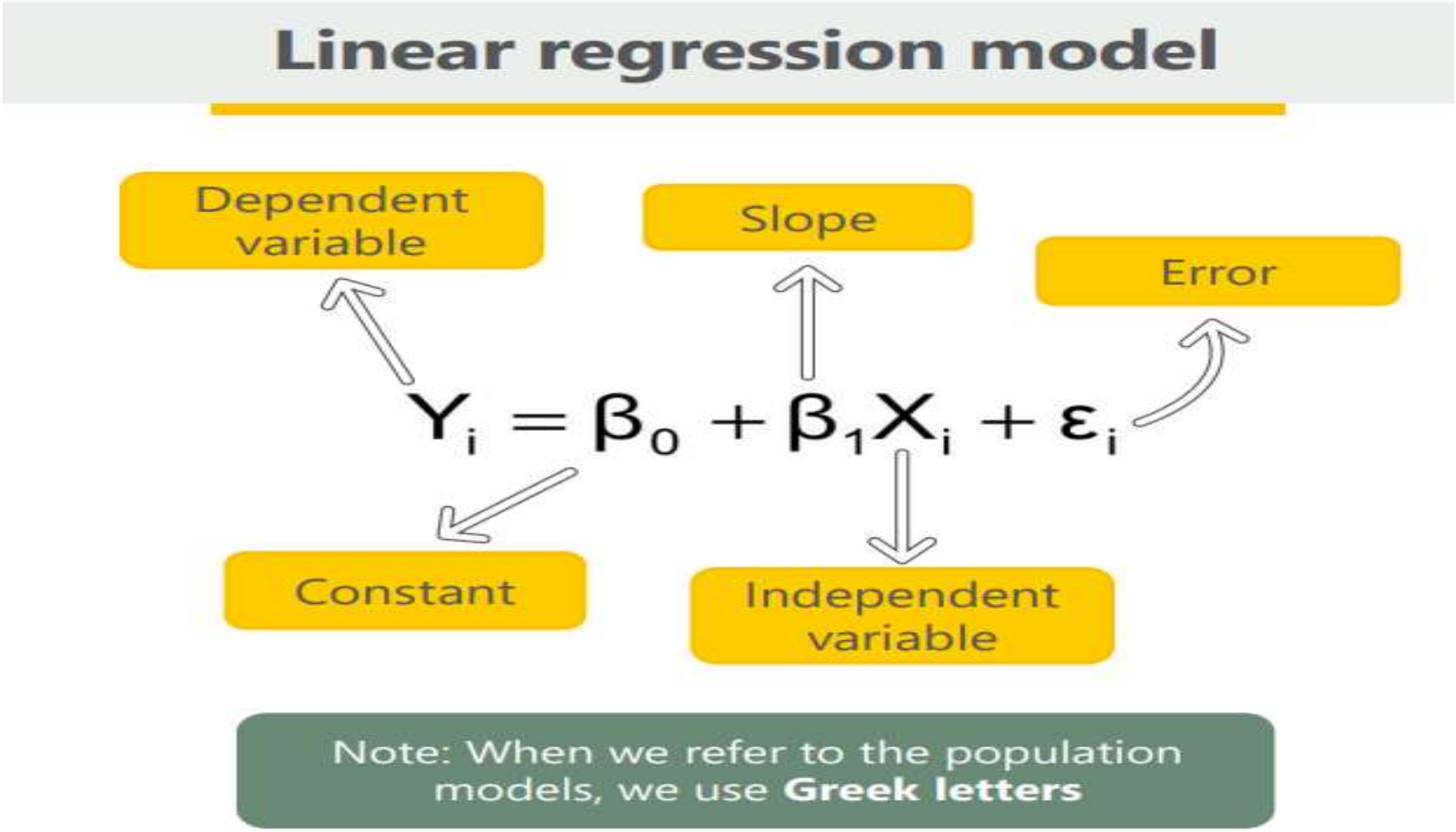
Simple Linear Regression

Linear regression model. Geometrical representation

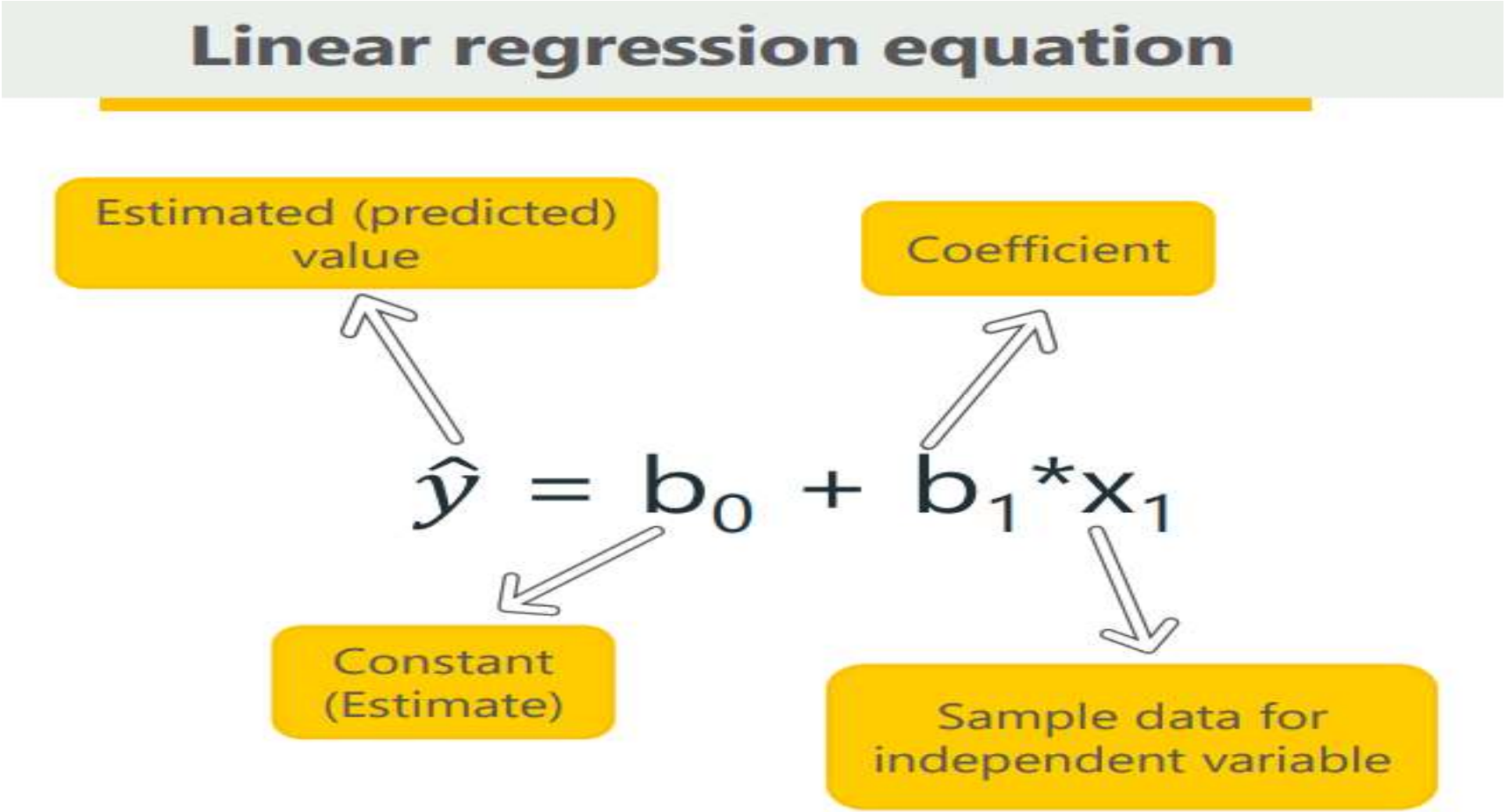
$$\hat{y}_i = b_0 + b_1 x_i$$



Simple Linear Regression



Simple Linear Regression



Multiple Linear Regression

Multiple Linear regression is a linear approach to modelling the relationship between a scalar response (or dependent variable) and more than one explanatory variables (or independent variables).

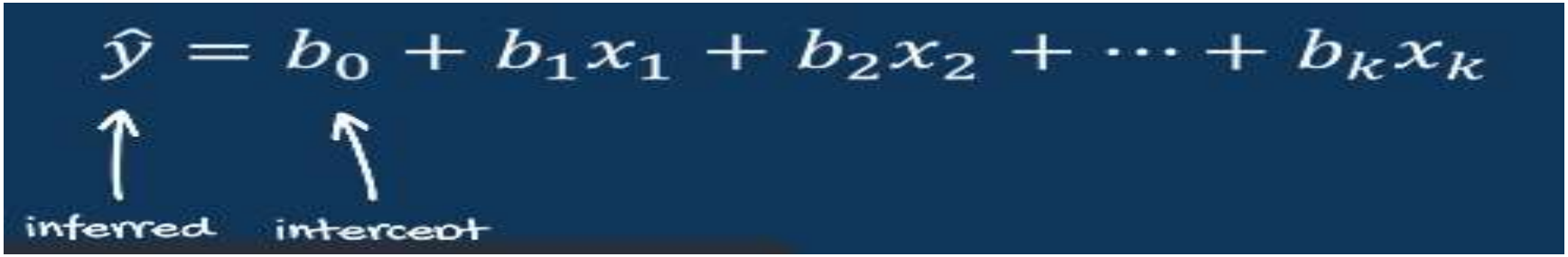
Multiple Linear Regression Model

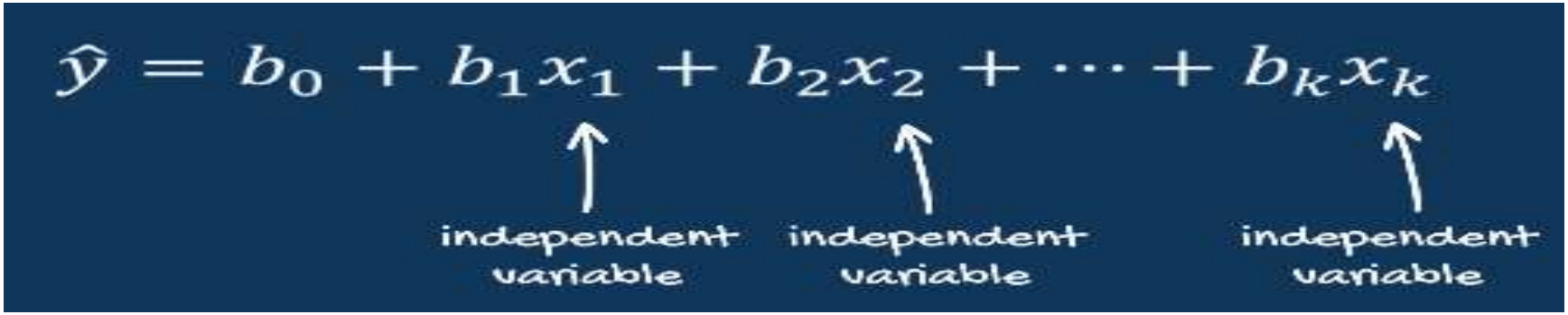
$$\gamma = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \varepsilon$$

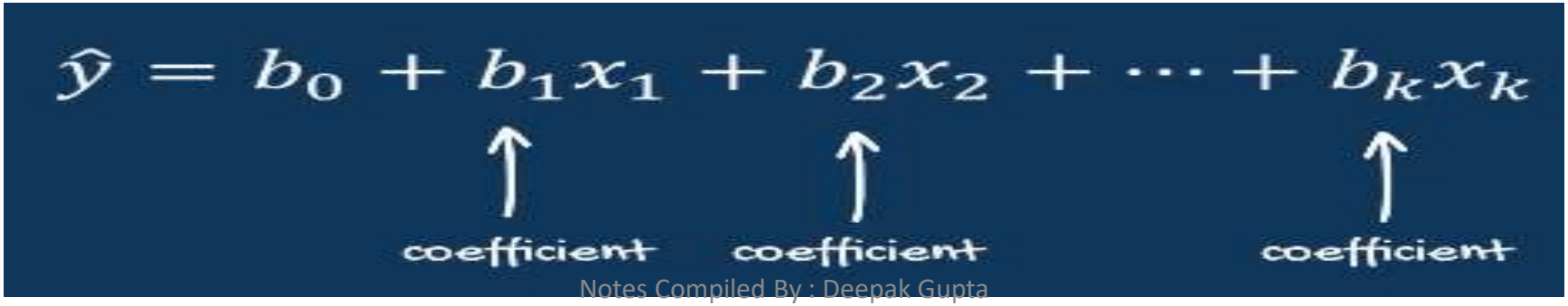
Multiple Linear Regression Equation

$$\hat{y} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_k x_k$$

Multiple Linear Regression







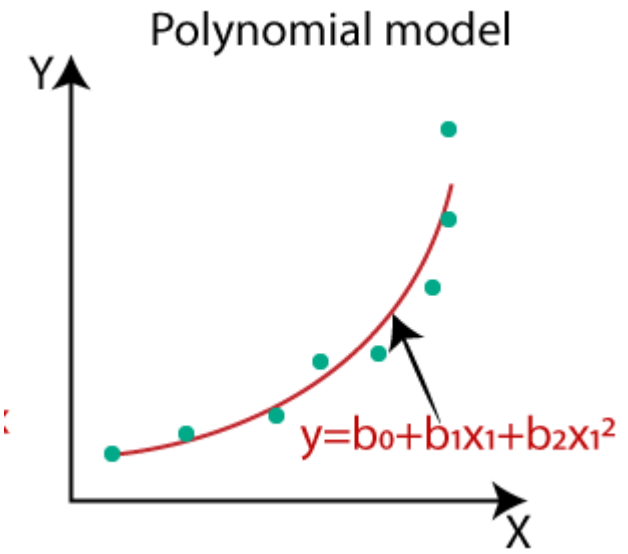
Polynomial Regression

Polynomial regression is a special case of linear regression where we fit a polynomial equation on the data with a curvilinear relationship between the target variable and the independent variables.

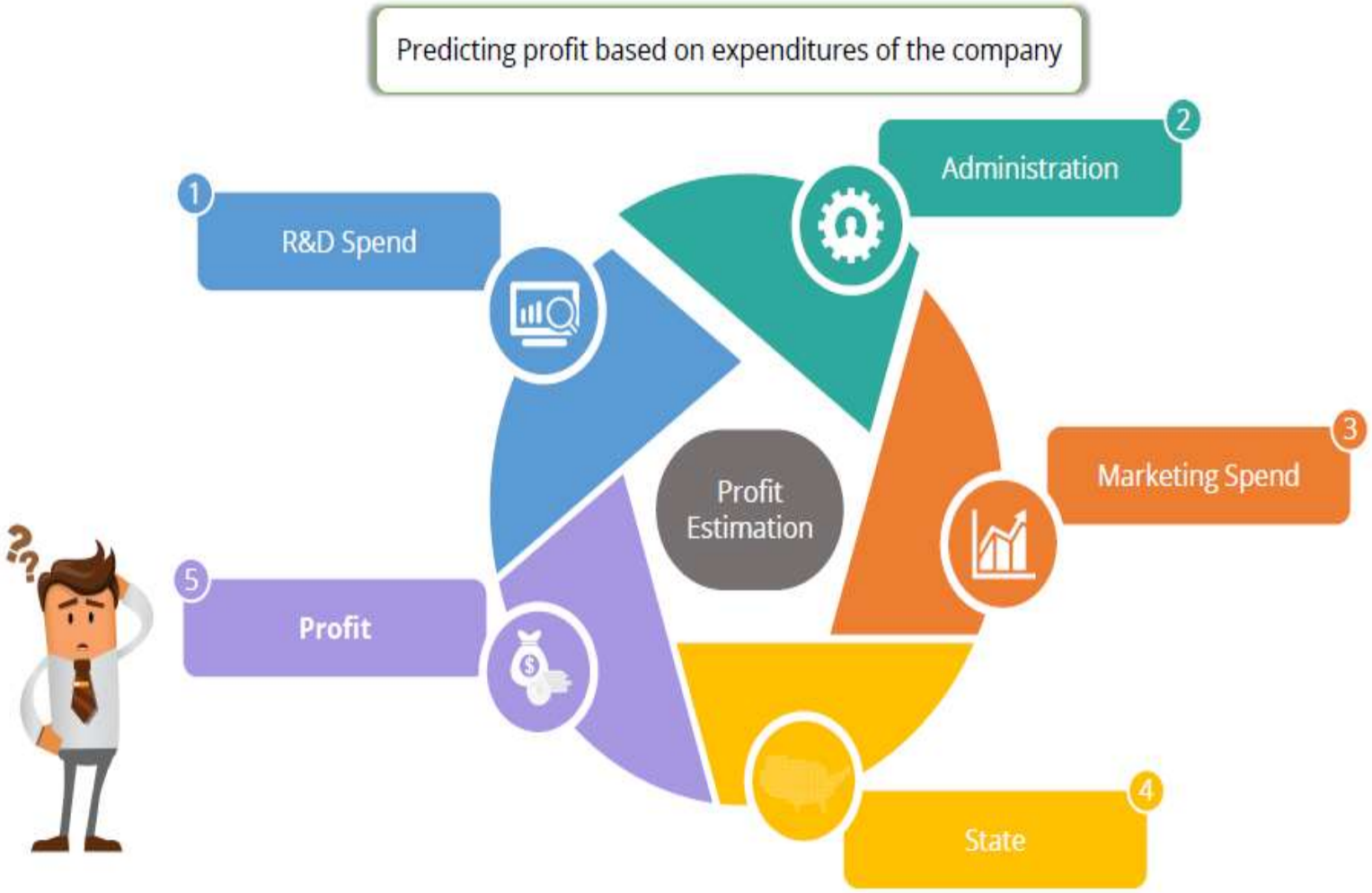
Polynomial Model Equation:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_n x^n$$

For example, if we are modelling the yield of a chemical synthesis in terms of the temperature at which the synthesis takes place, we may find that the yield improves by increasing amounts for each unit increase in temperature.



Regression Use Case



Correlation vs Regression

Correlation	vs	Regression
Represents the relationship between two variables		Represents the relationship between two or more variables
Shows that two variables move together (no matter in which direction)		Shows cause and effect (one variable is affected by the other)
Symmetrical w.r.t. the two variables: $\rho(x,y) = \rho(y,x)$		One way – there is always only one variable that is causally dependent
A single point (a number)		A line (in 2D space)

Multivariate regression

- As the name implies, multivariate regression is a technique that estimates a single regression model with more than one outcome variable. When there is more than one predictor variable in a multivariate regression model, the model is a multivariate multiple regression.

- Example: A doctor has collected data on cholesterol, blood pressure, and weight. She also collected data on the eating habits of the subjects (e.g., how many ounces of red meat, fish, dairy products, and chocolate consumed per week). She wants to investigate the relationship between the three measures of health and eating habits.

- Example: A researcher has collected data on three psychological variables, four academic variables (standardized test scores), and the type of educational program the student is in for 600 high school students. She is interested in how the set of psychological variables is related to the academic variables and the type of program the student is in.

Bayesian Modeling

- A model describes the data that one could observe from the system.
- If we use the mathematics of probability theory to express all the forms of uncertainty and noise associated with our model, Bayes' rule allows us to infer unknown quantities , adapt our model, make predictions and learn from data.
- Baye's theorem, in its basic form, is an intuitive process that we use everyday.
- Baye's rule tells us how to do inference about hypothesis from data, learning and predictions can be seen as forms of inference.
- The theorem states that –***Suppose we have an initial belief-when we get new information, we have a new, updated belief.***
- Baye's theorem provides a way to revise existing predictions or beliefs(update probabilities) given new or additional evidence. i.e. Bayes' theorem allows you to update predicted probabilities of an event by incorporating new information.

Bayesian Modeling

Important Terms: (if A and B are independent events)

- **Joint Probability:** Joint probability is the likelihood of **more than one event** occurring at the **same time** P(A and B) or **p(A ∩ B)**..
- A joint probability can be visually represented through a Venn diagram. As Shown in the Venn diagram above, the joint probability is where both circles overlap each other. It is called the "intersection of two events."

Joint Probability = P(A ∩ B) = P(A) x P(B)

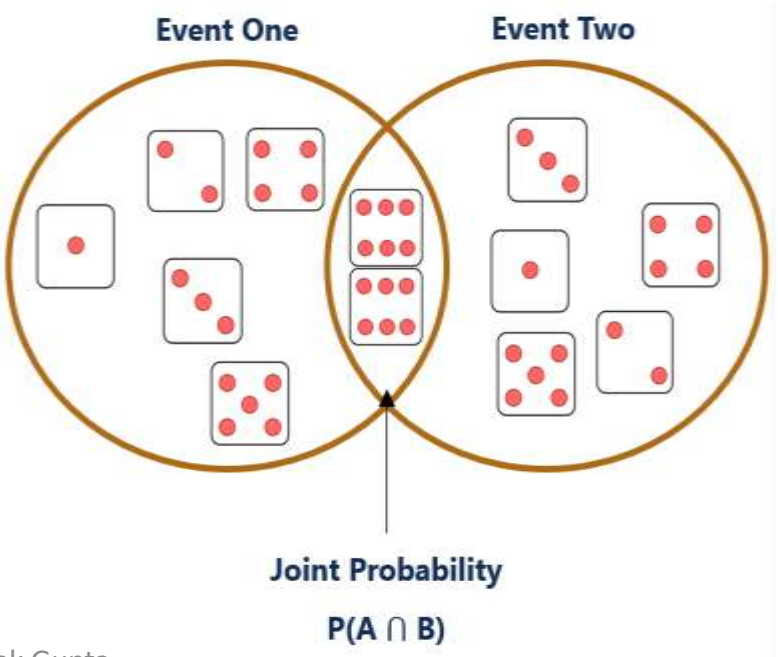
- **Example:** *Throwing two dice simultaneously.*

What is the joint probability of rolling two 6's in a fair six-sided dice?

Event "A" = The probability of rolling a 6 in the first roll is 1/6 = 0.1666.

Event "B" = The probability of rolling a 6 in the second roll is 1/6 = 0.1666.

Therefore, the joint probability of event "A" and "B" is P(1/6) x P(1/6) = 0.02777 = **2.8%**.



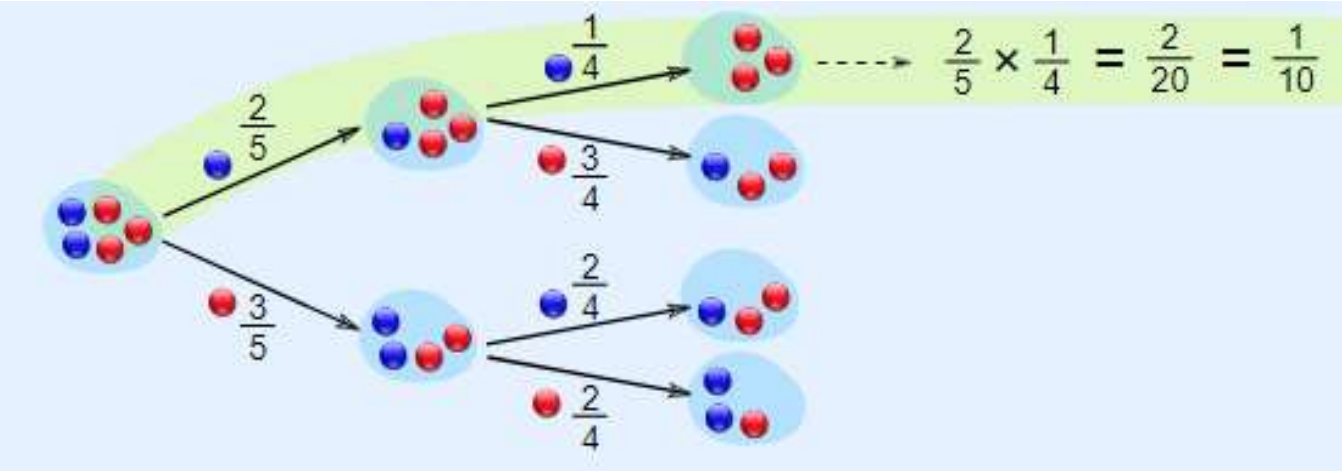
Bayesian Modeling

Important Terms: (if A and B are dependent events)

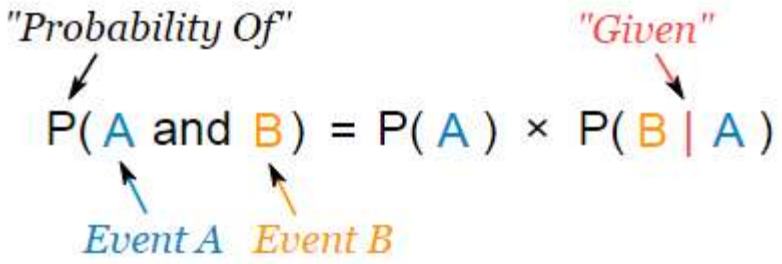
- **Conditional Probability:** The conditional probability of an **event *A*** is the probability that the event will occur given the knowledge that an **event *B* has already occurred**. It is denoted by **P(A|B).**

- Example: Two marbles are drawn from a bag having 2 blue and 3 red marbles. What is the probability of getting two blue marble?

- Solution: It is a **2/5 chance** followed by a **1/4 chance i.e. 1/10.**



Bayesian Modeling



*Thus, Probability of **event A and event B** equals the probability of **event A** times the probability of **event B given event A**"*

So Now we can conclude,

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}, \text{ if } P(B) \neq 0,$$

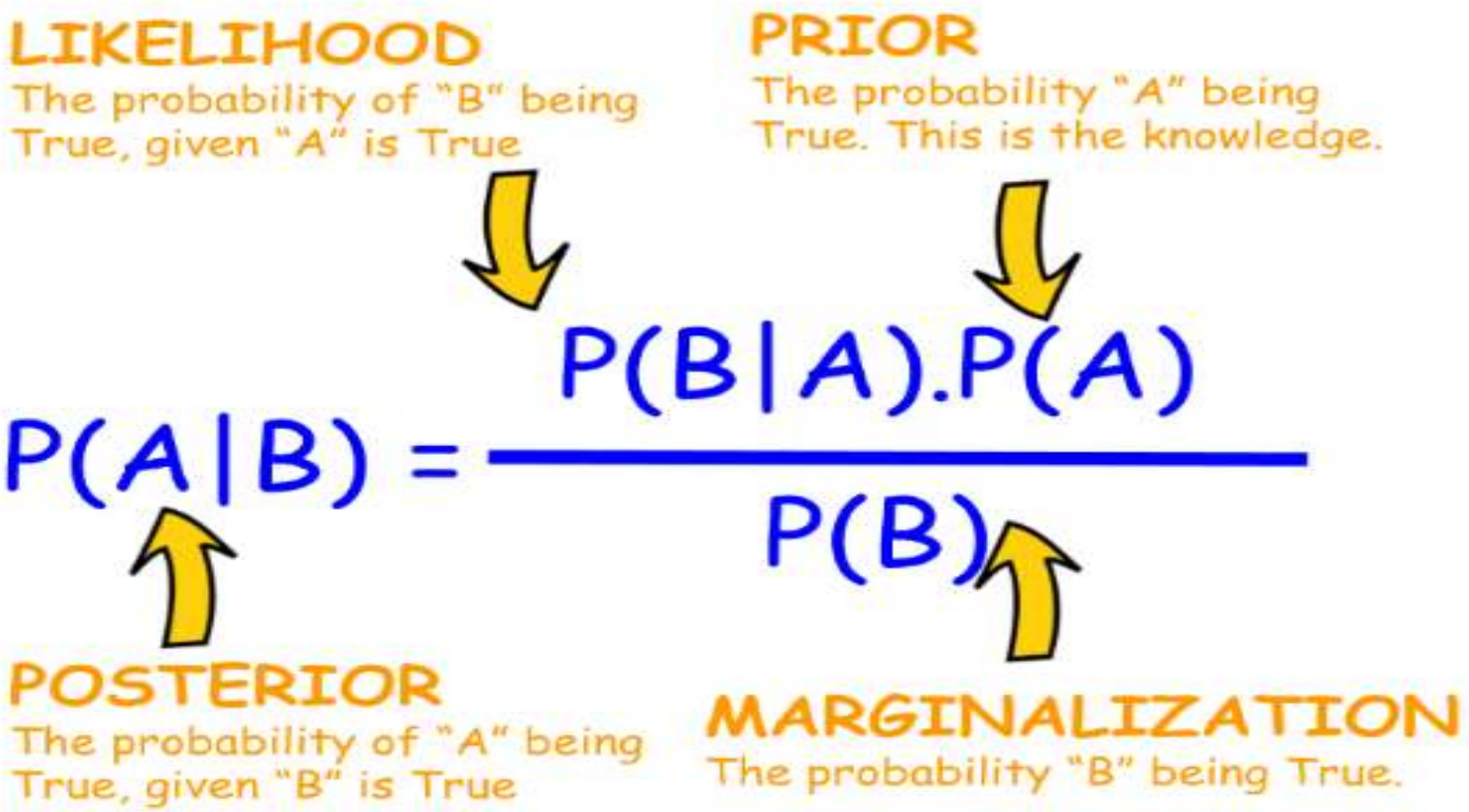
$$P(B \mid A) = \frac{P(B \cap A)}{P(A)}, \text{ if } P(A) \neq 0,$$

$$\Rightarrow P(A \cap B) = P(A \mid B) \times P(B) = P(B \mid A) \times P(A),$$

$$\Rightarrow P(A \mid B) = \frac{P(B \mid A) \times P(A)}{P(B)}, \text{ if } P(B) \neq 0.$$

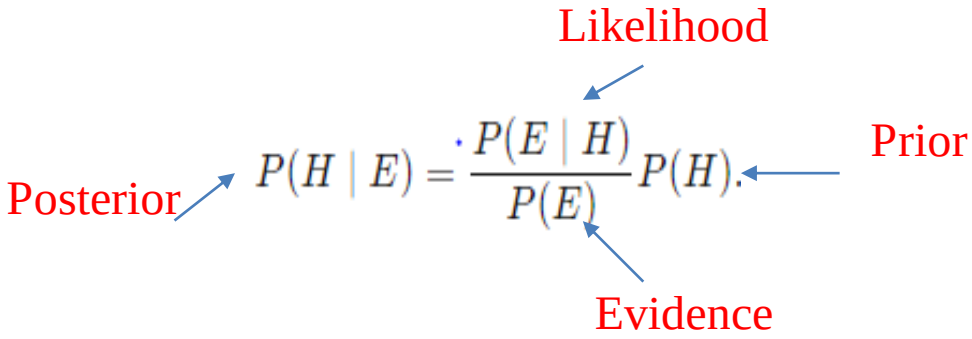
Baye's Theorem

Baye's Theorem states that **"*The posterior probability equals the prior probability times the likelihood ratio*".**



Bayes Theorem in Machine Learning

In machine learning, Given a hypothesis H and evidence E, Bayes' theorem states that the relationship between the probability of the hypothesis before getting the evidence P(H) and the probability of the hypothesis after getting the evidence $P(H|E)$ is



- **Posterior probability: How probable is our hypothesis after observing the evidence?**
- Prior probability: How probable was the hypothesis before observing the evidence?
- The **hypothesis** is your "guess" at what will occur. It is a testable assertion.
- **Evidence (Marginal): How probable is the new evidence under all possible hypothesis**?
- **Likelihood**: How probable is the evidence given that our hypothesis is true?
- You can think of **posterior probability** as an **adjustment** on the **prior probability**

Posterior = (Likelihood * Prior) / Evidence

Bayesian Modeling Example

Example: Picnic Day

You are planning a picnic today, but the morning is cloudy

- Oh no! 50% of all rainy days start off cloudy!
- But cloudy mornings are common (about 40% of days start cloudy)
- And this is usually a dry month (only 3 of 30 days tend to be rainy, or 10%)

What is the chance of rain during the day?

We will use Rain to mean rain during the day, and Cloud to mean cloudy morning.

The chance of Rain given Cloud is written P(Rain|Cloud)

So let's put that in the formula:

$$P(Rain|Cloud) = \frac{P(Rain)\ P(Cloud|Rain)}{P(Cloud)}$$

- P(Rain) is Probability of Rain = 10%
- P(Cloud|Rain) is Probability of Cloud, given that Rain happens = 50%
- P(Cloud) is Probability of Cloud = 40%

$$P(Rain|Cloud) = \frac{0.1 \times 0.5}{0.4} = .125$$

Or a 12.5% chance of rain. Not too bad, let's have a picnic!

Bayesian Modelling Example

Mach1: 30 wrenches / hr
Mach2: 20 wrenches / hr

-> P(Mach1) = 30/50 = 0.6

-> P(Mach2) = 20/50 = 0.4

Out of all produced parts:
We can SEE that 1% are defective

-> P(Defect) = 1%

Out of all defective parts:
We can SEE that 50% came from mach1
And 50% came from mach2

-> P(Mach1 | Defect) = 50%

-> P(Mach2 | Defect) = 50%

-> P(Defect | Mach2) = ?

Question:
What is the probability that a part produced by mach2 is defective = ?

Bayesian Modeling Example

-> P(Mach2) = 20/50 = 0.4
-> P(Defect) = 1%
-> P(Mach2 | Defect) = 50%
-> P(Defect | Mach2) = ?

$$P(\text{Defect} \mid \text{Mach2}) = \frac{P(\text{Mach2} \mid \text{Defect}) * P(\text{Defect})}{P(\text{Mach2})}$$

$$P(\text{Defect} \mid \text{Mach2}) = \frac{0.5 * 0.01}{0.4} = 0.0125$$

$$P(\text{Defect} \mid \text{Mach2}) = \frac{P(\text{Mach2} \mid \text{Defect}) * P(\text{Defect})}{P(\text{Mach2})} = 1.25\%$$

Bayesian Modeling Example

$$P(\text{Defect} \mid \text{Mach1}) = ?$$

Bayesian Inference

- **Inference** is the process of deducing properties about a population or probability distribution from data.
- **Bayesian inference** is therefore just the process of deducing properties about a population or probability distribution from data *using Bayes' theorem.*
- **Bayesian inference** is a method of statistical inference in which Bayes' theorem is used to update the probability for a hypothesis as more evidence or information becomes available.
- Bayesian inference has found application in a wide range of activities, including science, engineering, philosophy, medicine, sport, and law. In the philosophy of decision theory, Bayesian inference is closely related to subjective probability, often called "Bayesian probability".

Bayesian Inference

- One of the great things about Bayesian inference is that you don't need lots of data to use it. one observation is enough to update the prior. In fact, the Bayesian framework allows you to update your beliefs iteratively in real time as data comes in.

- It works as follows: you have a prior belief about something (e.g. the value of a parameter) and then you receive some data. You can update your beliefs by calculating the posterior distribution like we did above. Afterwards, we get even more data come in. So our posterior becomes the new prior.

- We can update the new prior with the likelihood derived from the new data and again we get a new posterior. This cycle can continue indefinitely so you're continuously updating your beliefs.

Bayesian Network

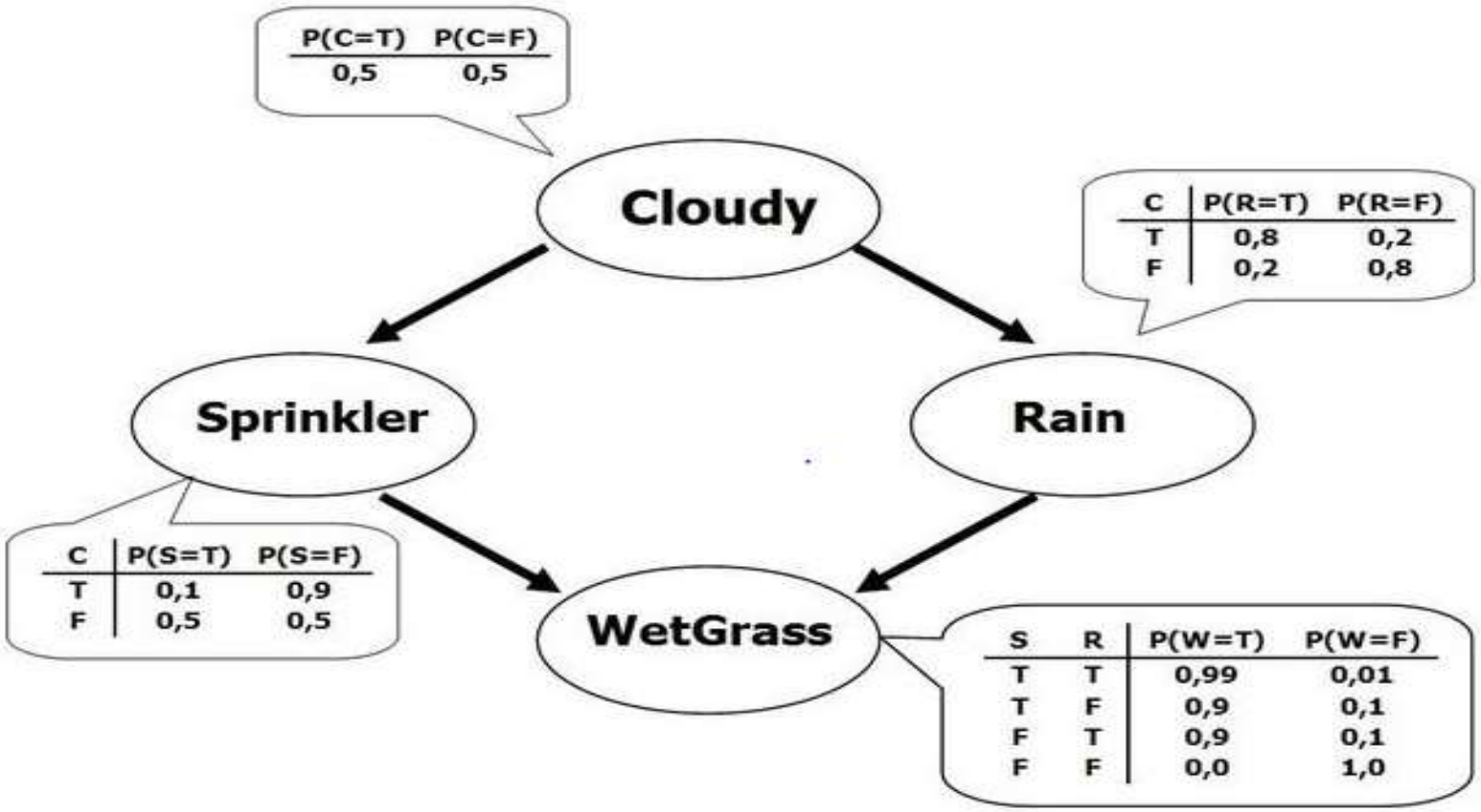
A Bayesian network is a directed graph in which each node is annotated with quantitative probability information. The full specification is as follows:

1. A set of random variables makes up the nodes of the network. Variables may be discrete or continuous.
2. A set of directed links or arrows connects pairs of nodes. If there is an arrow from node X to node Y, X is said to be apparent of Y.
3. Each node X_i has a conditional probability distribution $P(X_i|Parents(X_i))$ that quantifies the effect of the parents on the node.
4. The graph has no directed cycles (and hence is a directed, acyclic graph, or DAG).

Bayesian Network

- ***A Bayesian belief network describes the joint probability distribution for a set of variables.***
- **Bayesian networks** are a type of probabilistic graphical model that uses Bayesian inference for probability computations.
- Bayesian networks are a type of probabilistic graphical model comprised of nodes and directed edges.
- Bayesian network models capture both conditionally dependent and conditionally independent relationships between random variables.
- Models can be prepared by experts or learned from data, then used for inference to estimate the probabilities for causal or subsequent events.
- Designing a Bayesian Network requires defining at least three things:
  - **Random Variables**. What are the random variables in the problem?
  - **Conditional Relationships**. What are the conditional relationships between the variables?
  - **Probability Distributions**. What are the probability distributions for each variable?
- *All missing connections define the conditional independencies in the model.*

Bayesian Network Example



In the above example, since Rain has an edge going into WetGrass, it means that **P(WetGrass|Rain)** will be a factor, whose probability values are specified next to the WetGrass node in a conditional probability table.

Bayesian Network

- Bayesian networks satisfy the **local Markov property**, which states that a node is conditionally independent of its non-descendants given its parents.

- In the previous example, this means that

  P(Sprinkler|Cloudy, Rain) = P(Sprinkler|Cloudy)

  since Sprinkler is conditionally independent of its non-descendant, Rain, given Cloudy.

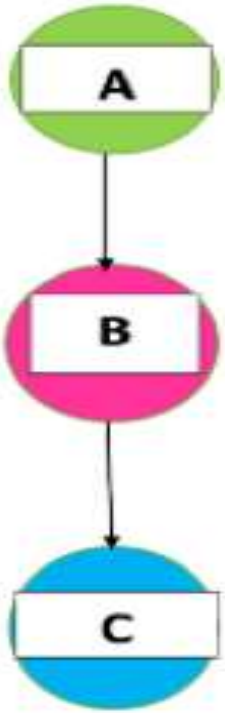
- This property allows us to simplify the joint distribution, obtained in the previous section using the chain rule, to a smaller form.

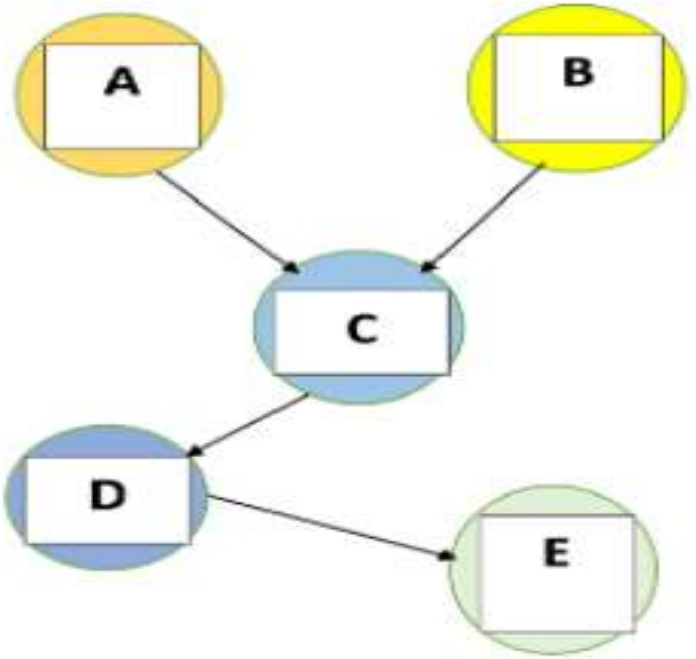
- We can evaluate the joint probability of a particular assignment of values for each variable (or a subset) in the network. For this, we already have a factorized form of the joint distribution, so we simply evaluate that product using the provided conditional probabilities. If we only care about a subset of variables, we will need to marginalize out the ones we are not interested in.

Bayesian Network

The probability of a random variable depends on his parents. Therefore, we can formulate Bayesian Networks as:

$$P(X_1, \dots, X_n) = \Pi_{i=1}^{n}\, p(X_i \mid Parents(X_i))$$





P(A,B,C) = P(C|B).P(B|A).P(A)

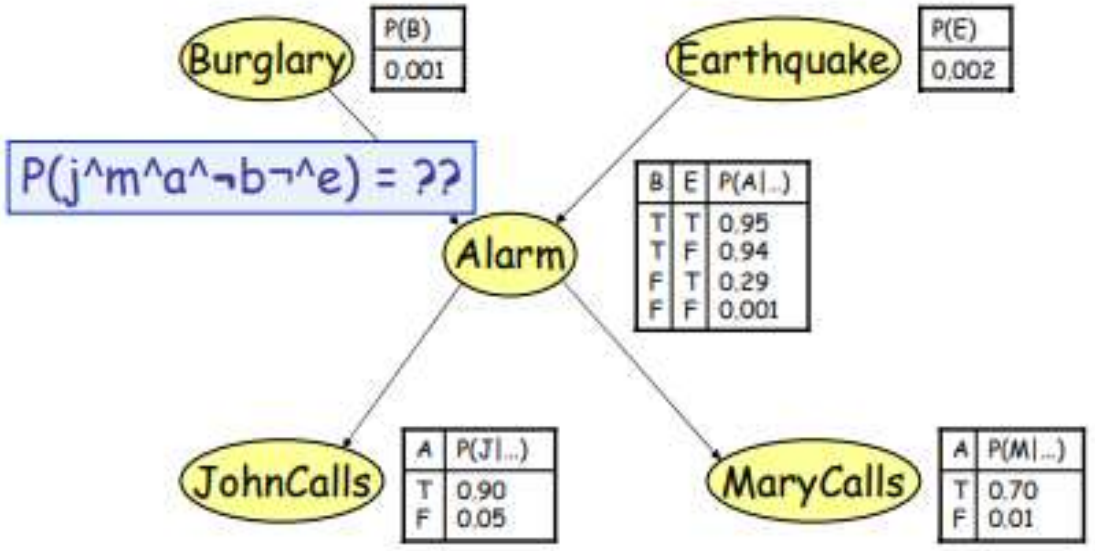
P(A,B,C,D,E) = P(E|D).P(D|C).P(C|A,B).P(B).P(A)

Bayesian Network

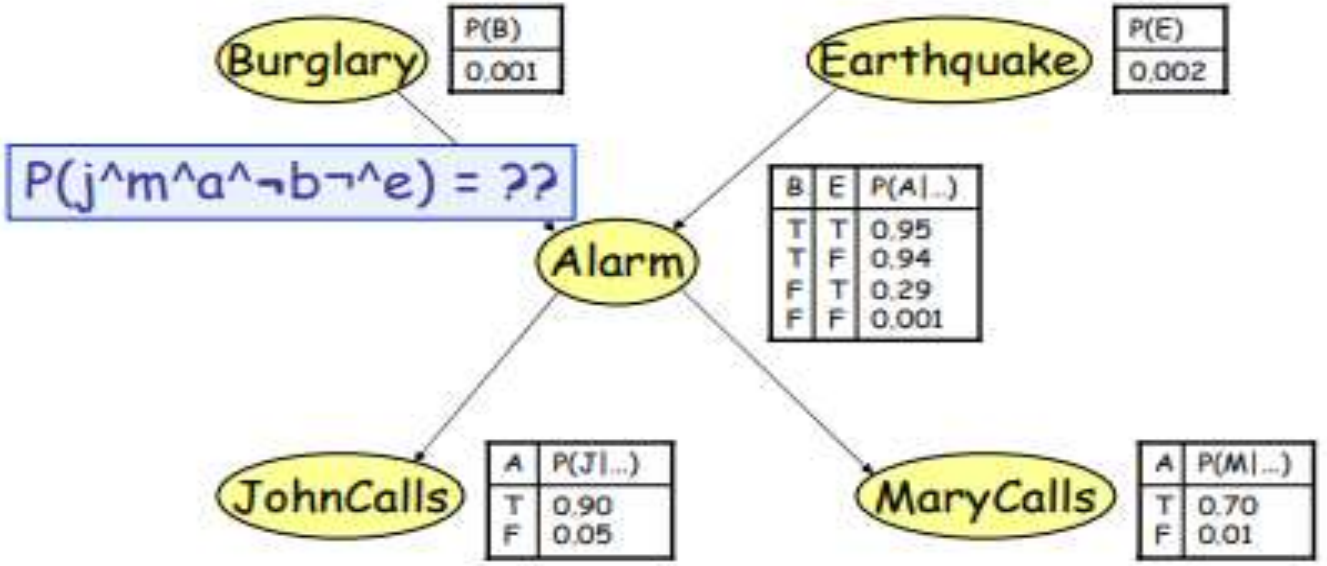
Consider the following scenario with 5 binary variables:

- B = a burglary occurs at house
- E = an earthquake occurs at house
- A = the alarm goes off
- J = John calls to report the alarm
- M = Mary calls to report the alarm

Calculate the probability that alarm has sound, neither a burglary nor an earthquake has occurred, and both john and Mary call.

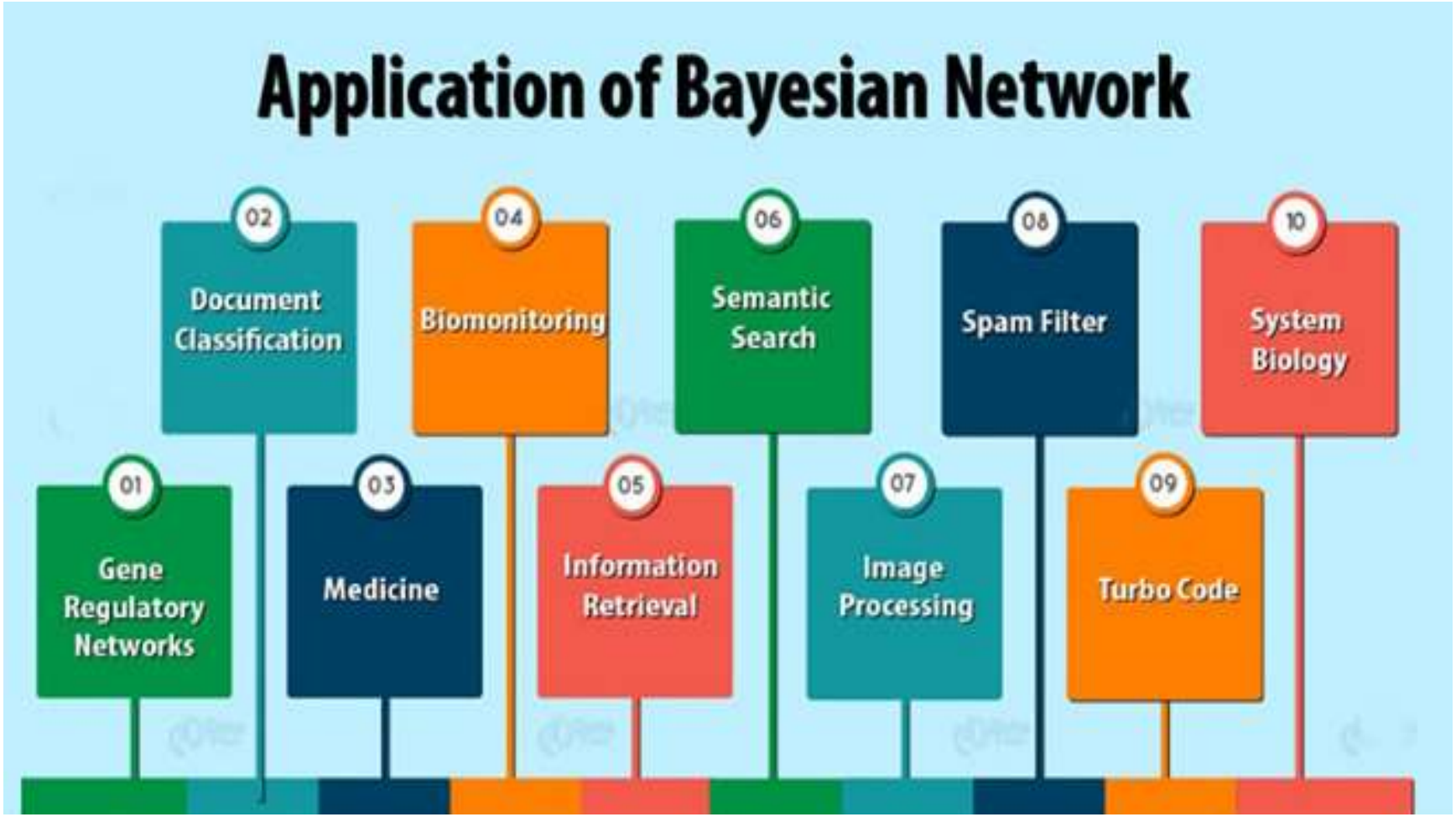


Bayesian Network



$$P(J \wedge M \wedge A \wedge \neg B \wedge \neg E)$$
$$= P(J|A)P(M|A)P(A|\neg B, \neg E)P(\neg B)P(\neg E)$$
$$= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998$$
$$= 0.00062$$

Applications of Bayesian Networks



Support Vector Machines

- Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection. But, it is widely used in classification objectives.

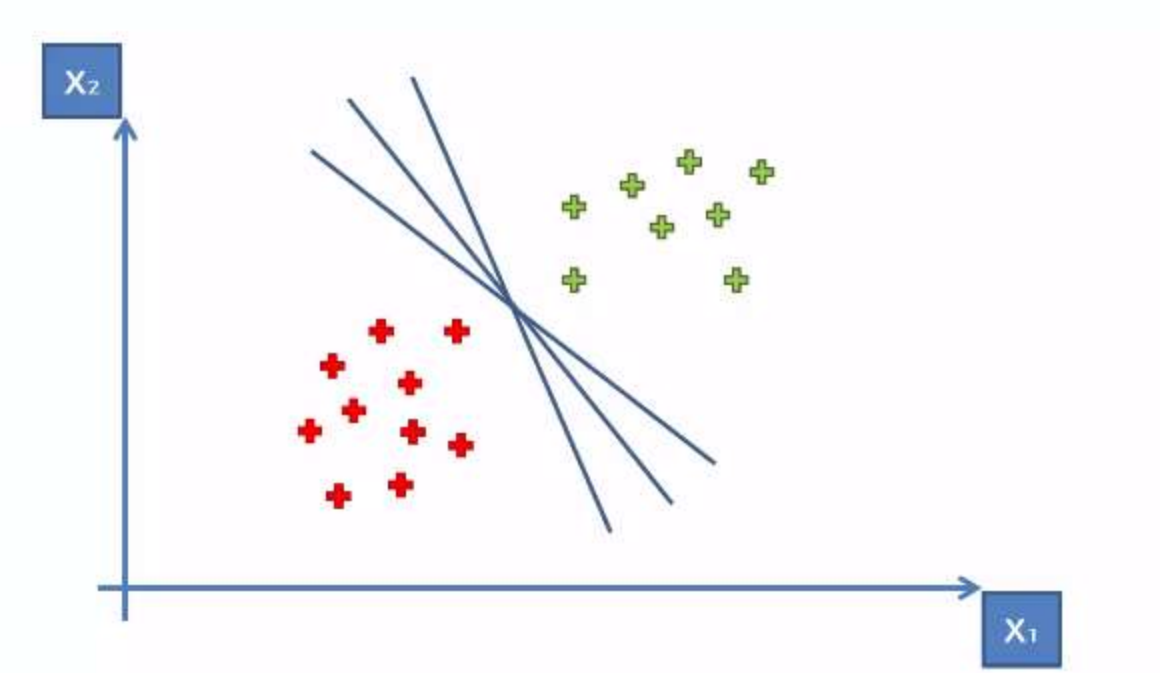
- The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.

- To separate the two classes of data points, there are many possible hyperplanes that could be chosen.

- Our objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes.

- Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence. In general the larger the margin, the lower the generalization error of the classifier.

Support Vector Machines (SVM)



It's difficult to evaluate the optimal separator.

Support Vector Machines

- Hyperplanes are decision boundaries that help classify the data points.
- Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3.

- For 2D dataset, the equation of hyperplane can be written as:

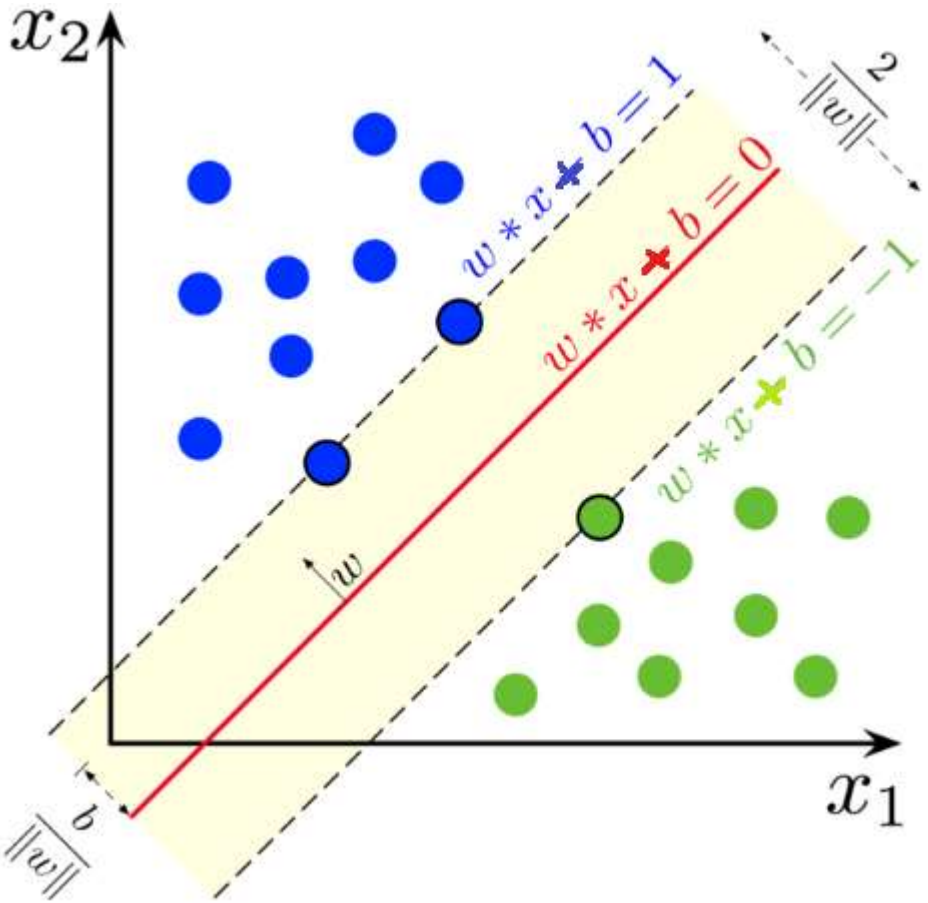
$$\mathbf{w.x+b=0}$$

- If $\mathbf{w.x+b= 1}$
  then we get (+)class hyperplane for all positive(x) points satisfy this rule ($\mathbf{w.x+b \geq 1}$)

- If $\mathbf{w.x+b= -1}$

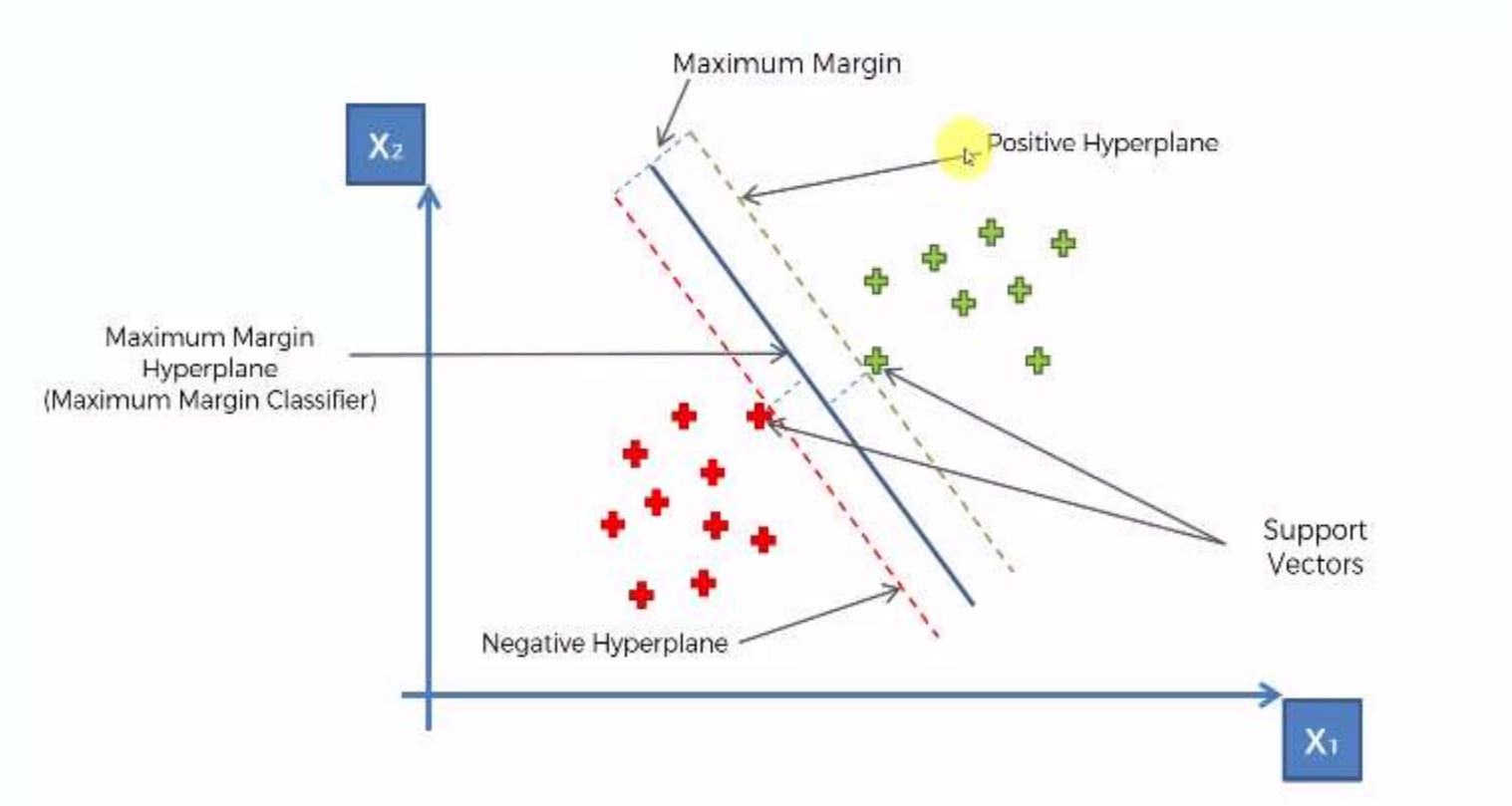
then we get (-)class hyperplane for all negative(x) points satisfy this rule $\mathbf{(w.x+b \leq -1)}$

Support Vector Machines



Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.

Support Vector Machines (SVM)

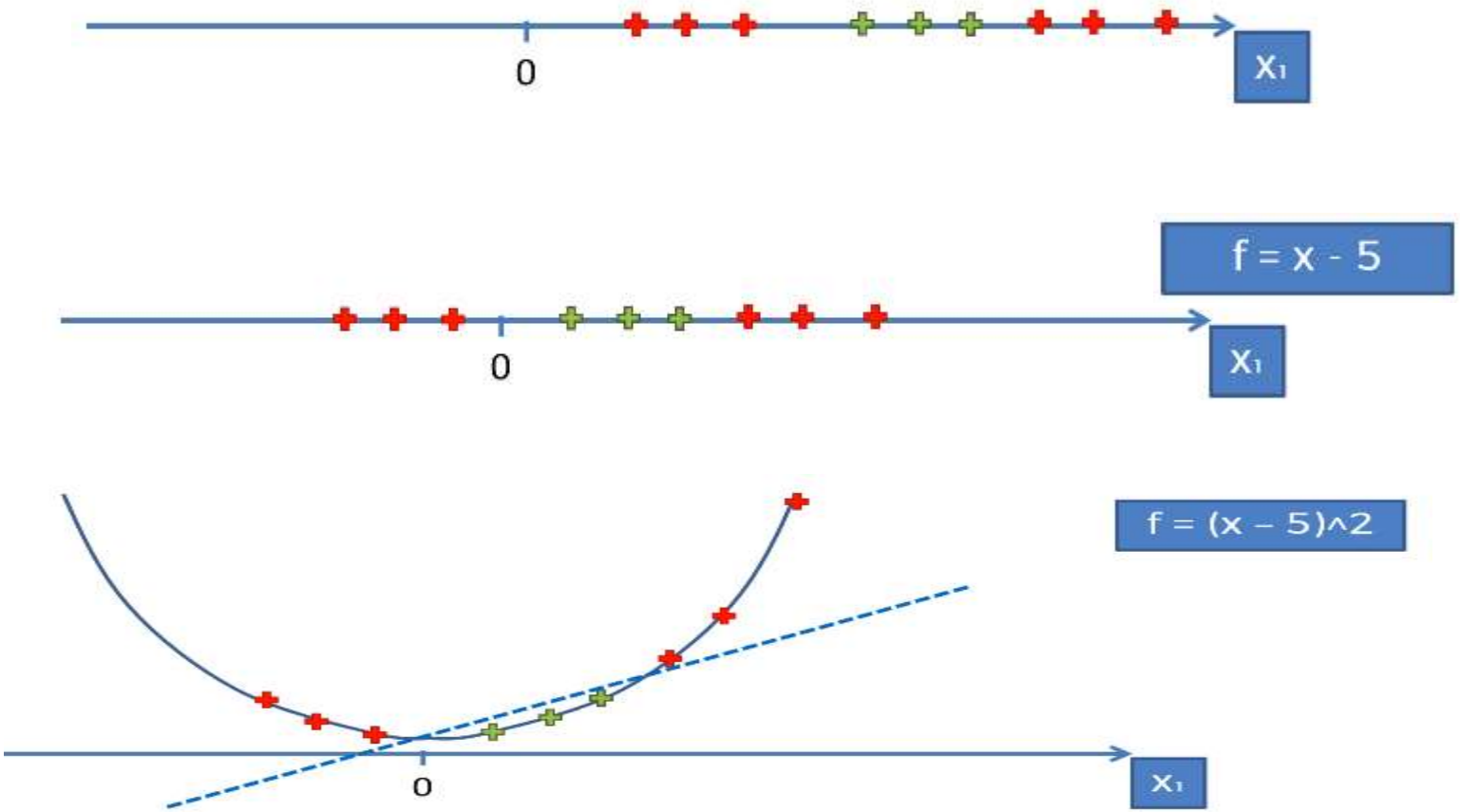


Support Vector Machines

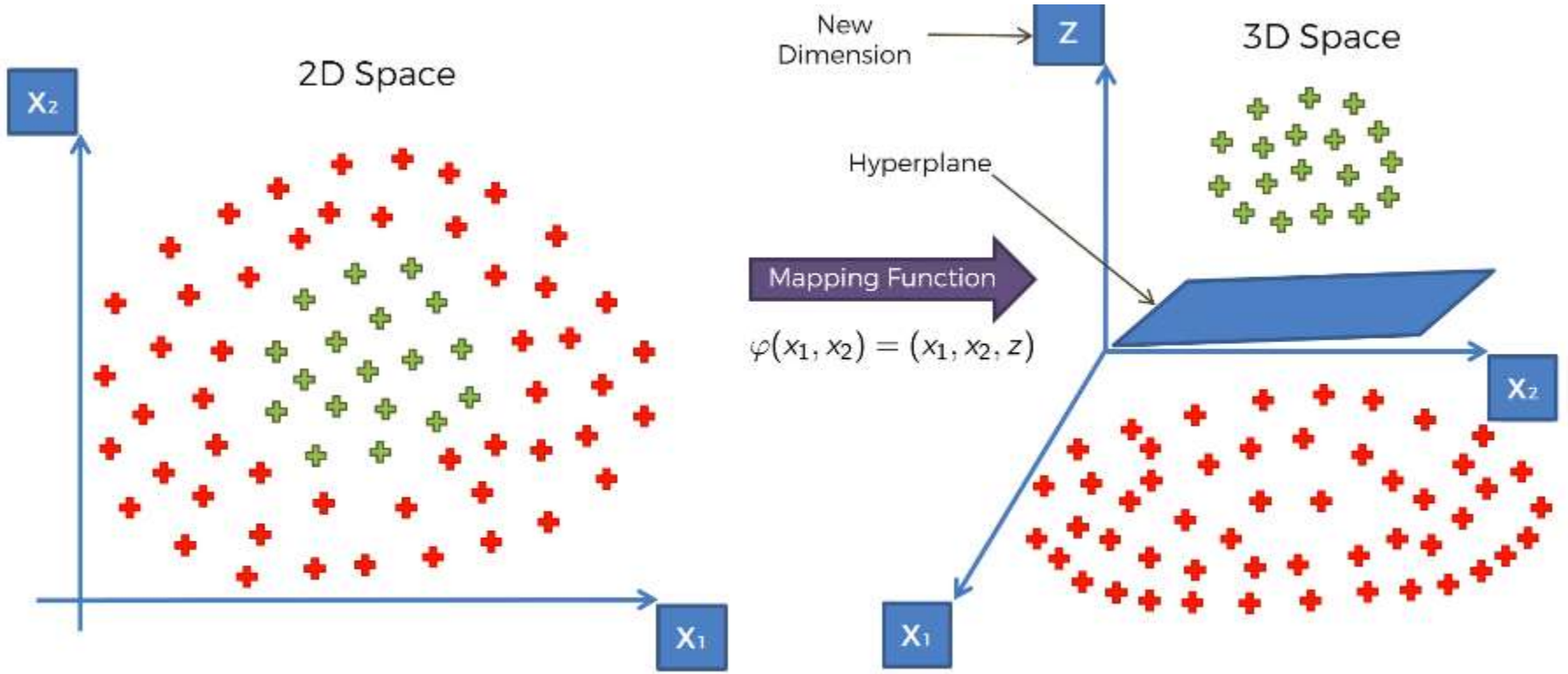
- The advantages of support vector machines are:
  - Effective in high dimensional spaces.
  - Still effective in cases where number of dimensions is greater than the number of samples.
  - Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
  - Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.
- The disadvantages of support vector machines include:
  - If the number of features is much greater than the number of samples, avoid over-fitting in choosing Kernel functions and regularization term is crucial.
  - SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation (see Scores and probabilities, below).

Non-Linearly Separable Data

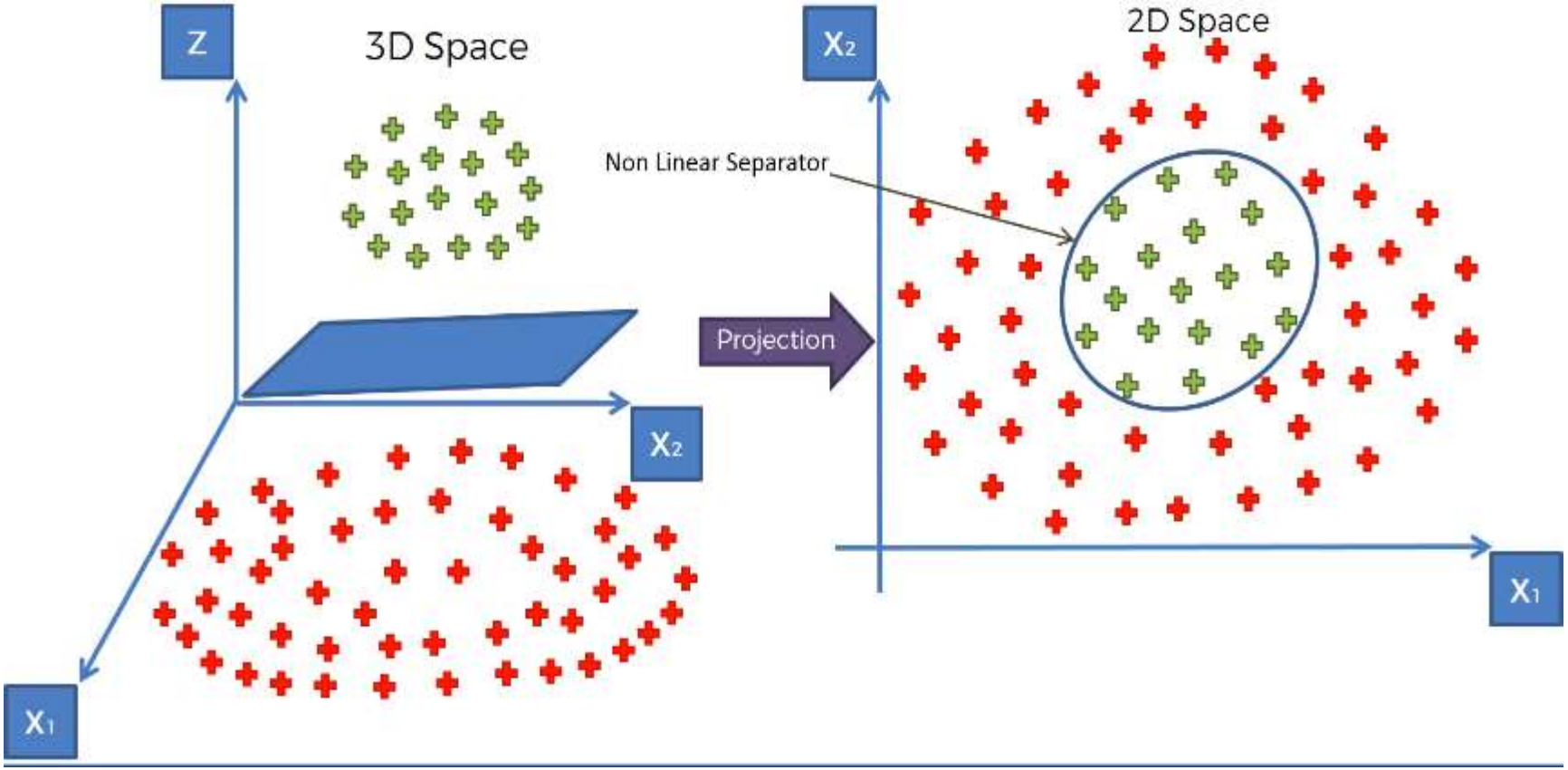
Solution: Mapping 1D data into higher dimensions.



Non-Linearly Separable Data



Projecting Back to 2D Space



Problem in Previous Approach

Mapping to a Higher Dimensional Space can be highly compute-intensive

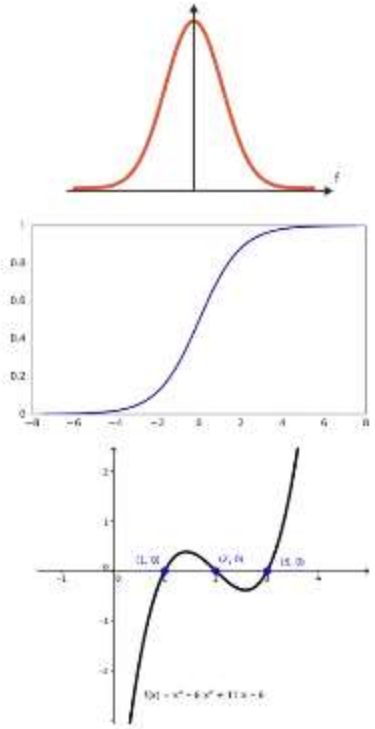
Then,

What is Solution of this problem?

Solution: Kernel Trick

- SVM algorithms use a set of mathematical functions that are defined as the kernel.
- The function of kernel is to take data as input and transform it into the required form.
- Different SVM algorithms use different types of kernel functions. These functions can be different types. For example ***linear, nonlinear, polynomial, radial basis function(RBF), and sigmoid.***
- Introduce Kernel functions for sequence data, graphs, text, images, as well as vectors. The most used type of kernel function is **RBF.** Because it has localized and finite response along the entire x-axis.
- The kernel functions return the inner product between two points in a suitable feature space. Thus by defining a notion of similarity, with little computational cost even in very high-dimensional spaces.

Types of Kernel Functions



Gaussian RBF Kernel

$$K\left(\vec{x}, \vec{l}^{i}\right) = e^{-\frac{\left\|\vec{x}-\vec{l}^{i}\right\|^{2}}{2\sigma^{2}}}$$

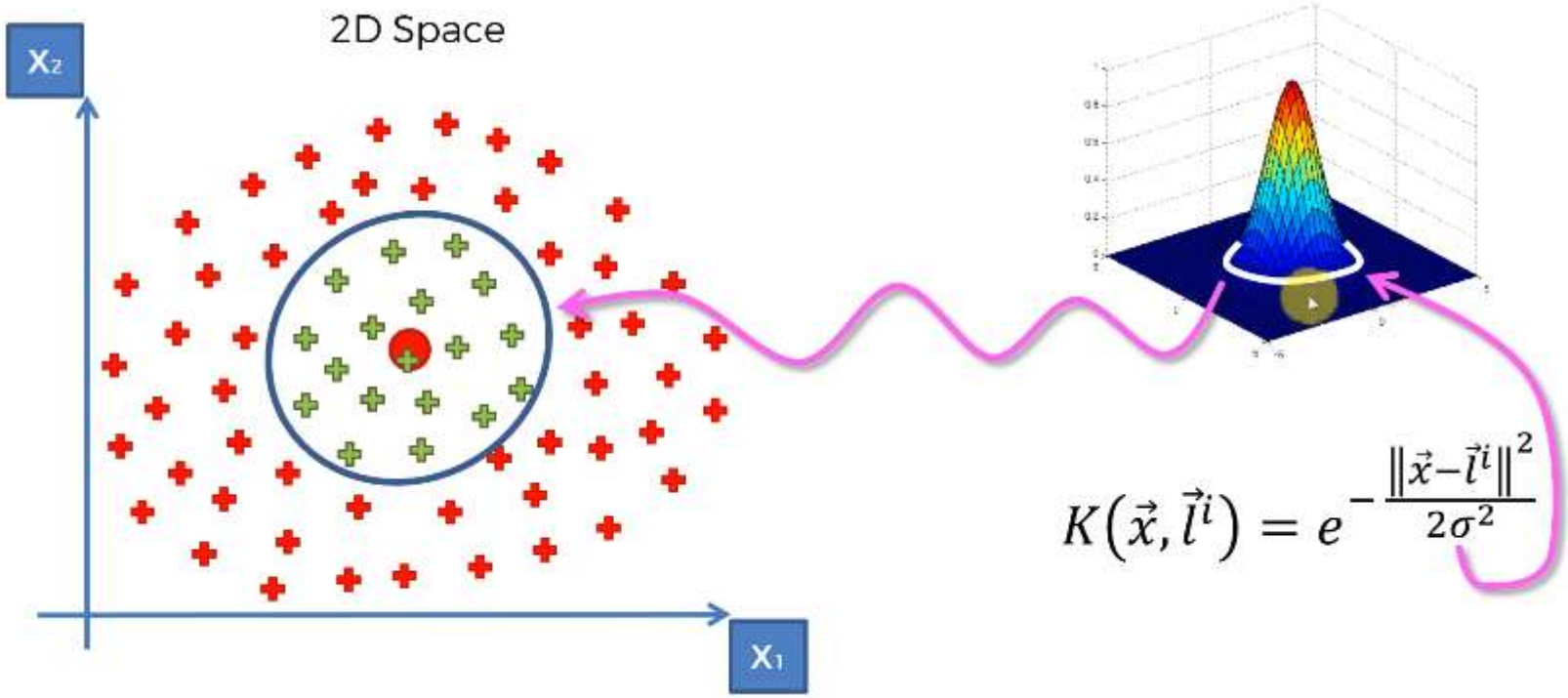
Sigmoid Kernel

$$K(X, Y) = \tanh(\gamma \cdot X^{T}Y + r)$$

Polynomial Kernel

$$K(X, Y) = (\gamma \cdot X^{T}Y + r)^{d}, \gamma > 0$$

SVM using Gaussian RBF Kernel



Time Series Analysis

A **time series** is a sequence of information that attaches a time period to each value.

"Set of observations on a quantitative variable collected over time."

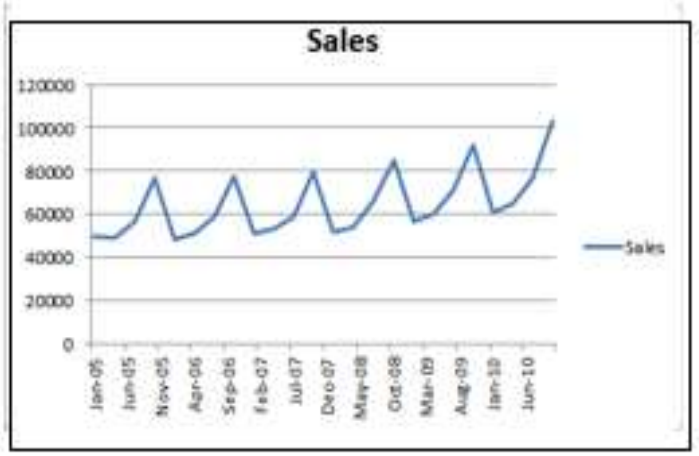
"A sequence of continuous, real-valued elements, is known as a time-series."

Time Series can be defined as a set of measurements of certain variable made at **regular time intervals**.

Time acts as an independent variable for estimation

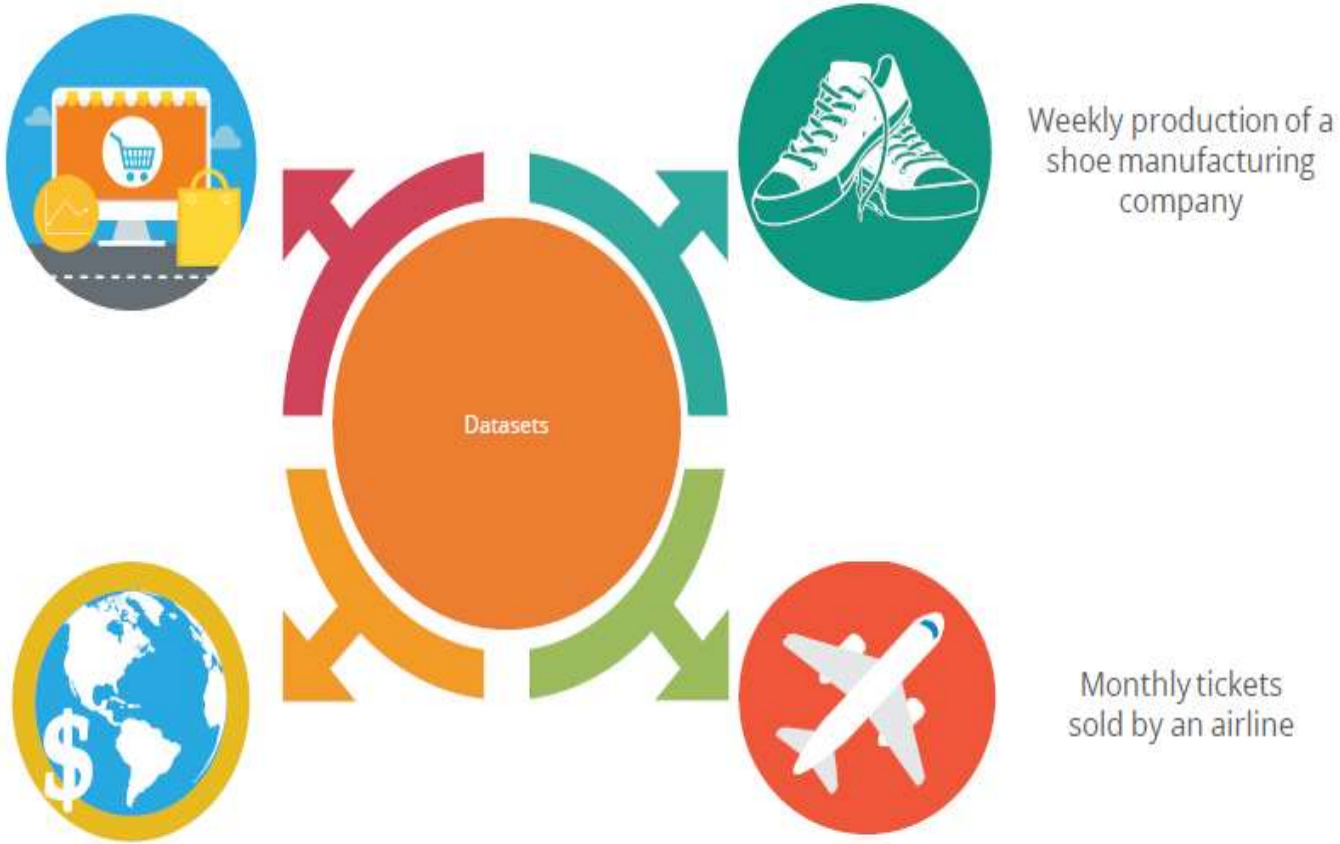
A time series defined by the values Y1, Y2.. of a variable Y at times t1, t2, t3.. is given by :

$$Y = F(t)$$

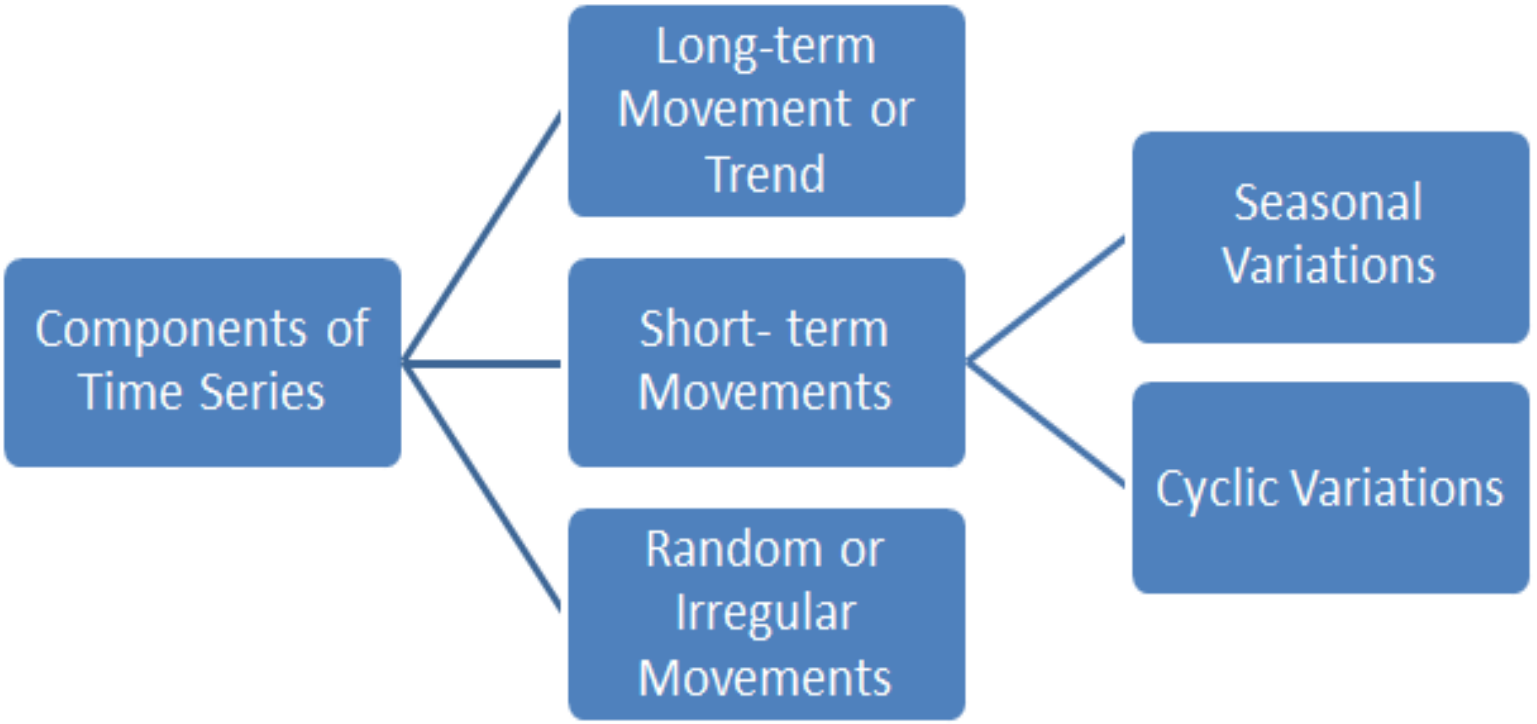


Series of monthly sales data

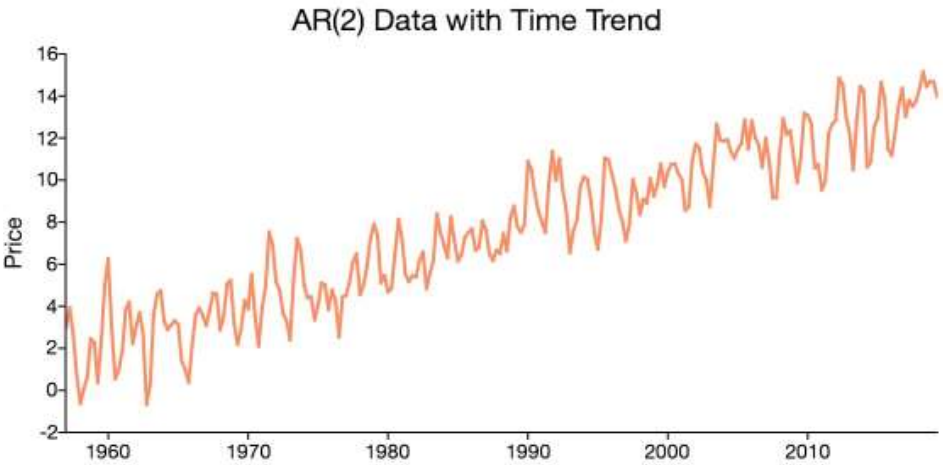
Applications



Components of Time Series



Time Series Pattern Types



Uptrend

Smartphone sales

Downtrend

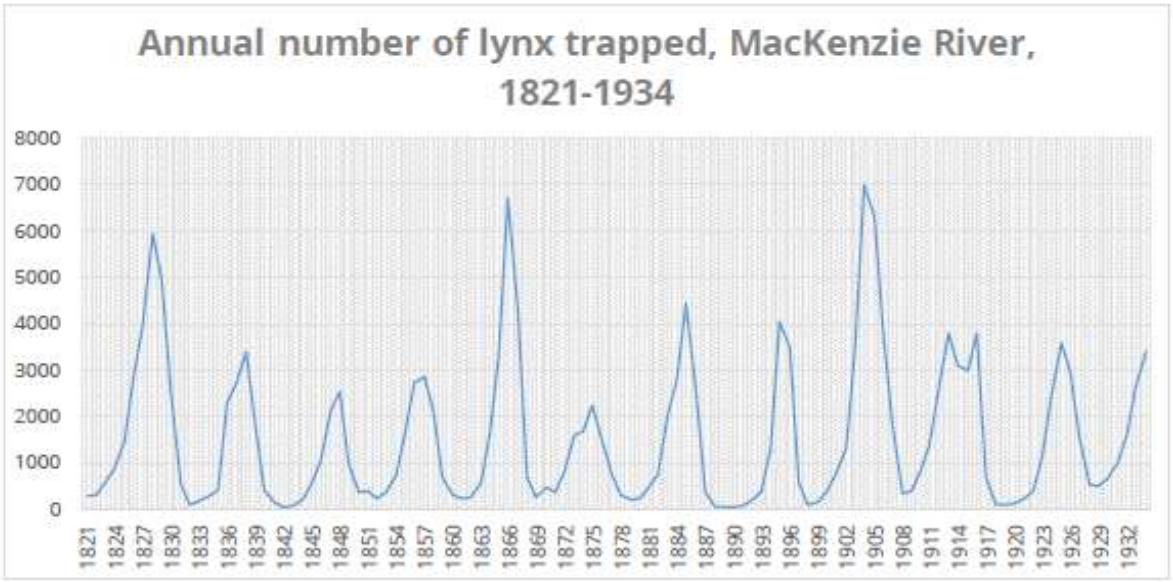
Stock Market price for a wall street company

Trend

A trend is a long-term increase or decrease in time series data

- A *trend* exists when there is a long-term increase or decrease in the data. It does not have to be linear.
- A trend is a smooth, general, long-term, average tendency. It is not always necessary that the increase or decrease is in the same direction throughout the given period of time.
- It is observable that the tendencies may increase, decrease or are stable in different sections of time. But the overall trend must be upward, downward or stable. The population, agricultural production, items manufactured, number of births and deaths, number of industry or any factory, number of schools or colleges are some of its example showing some kind of tendencies of movement.

Time Series Pattern Types (Contd.)

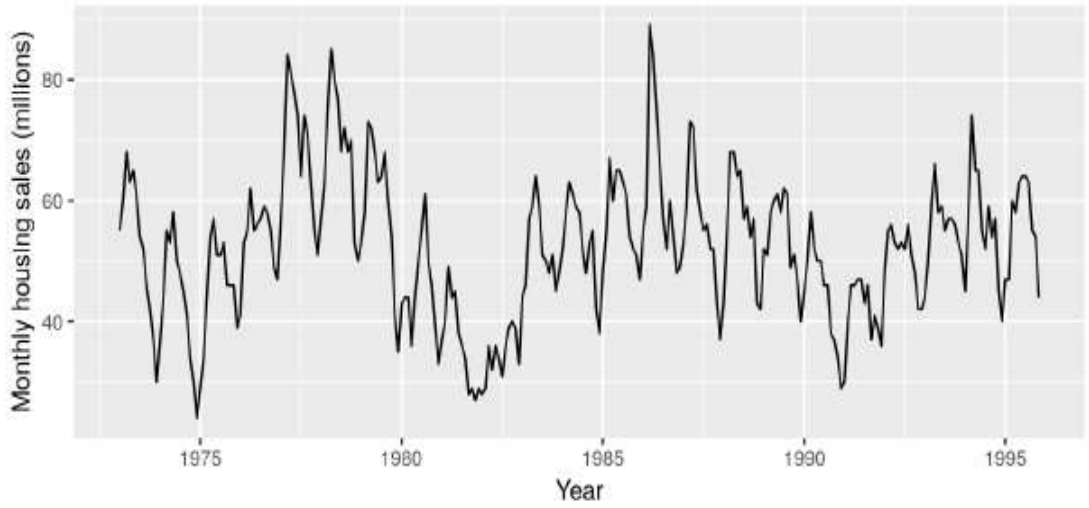


Seasonal

- When factors such as the time of the year or the day of the week affect the dependent variable, repetitive patterns are observed in the time series
- Seasonality is always of a fixed and known frequency

- These are the rhythmic forces which operate in a regular and periodic manner over a span of less than a year. They have the same or almost the same pattern during a period of 12 months. This variation will be present in a time series if the data are recorded hourly, daily, weekly, quarterly, or monthly.
- Such as production of crops depends on seasons, the sale of umbrella and raincoats in the rainy season, and the sale of electric fans and A.C. shoots up in summer seasons.
- An upswing in a season should not be taken as an indicator of better business conditions.

Time Series Pattern Types (Contd.)

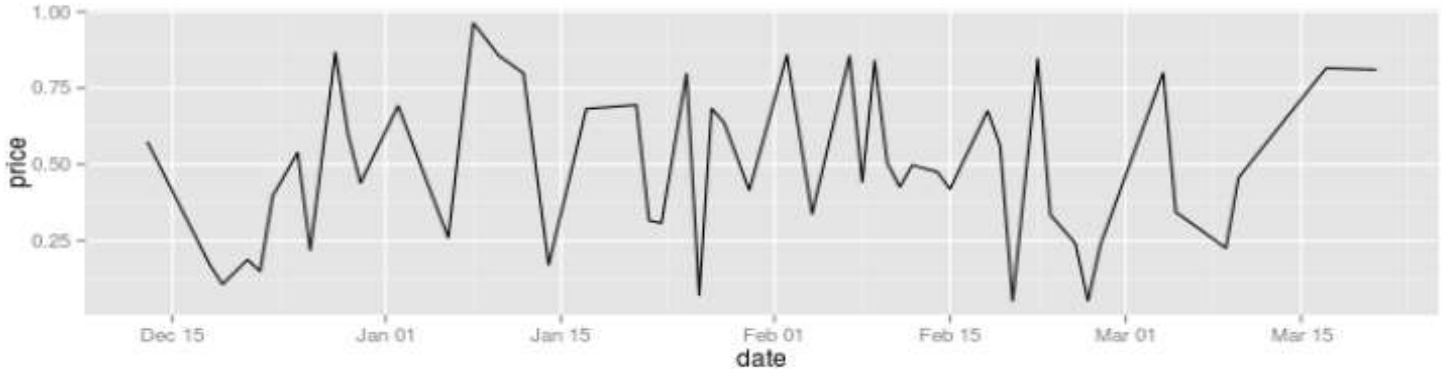


Cyclic

- Unlike seasonal patterns, cyclic patterns exhibit rise and fall that are not of fixed period
- Duration is at least 2 years

- A *cycle* occurs when the data exhibit rises and falls that are not of a fixed frequency.
- These fluctuations are usually due to economic conditions, and are often related to the "business cycle". The duration of these fluctuations is usually at least 2 years.

Time Series Pattern Types (Contd.)



Irregular

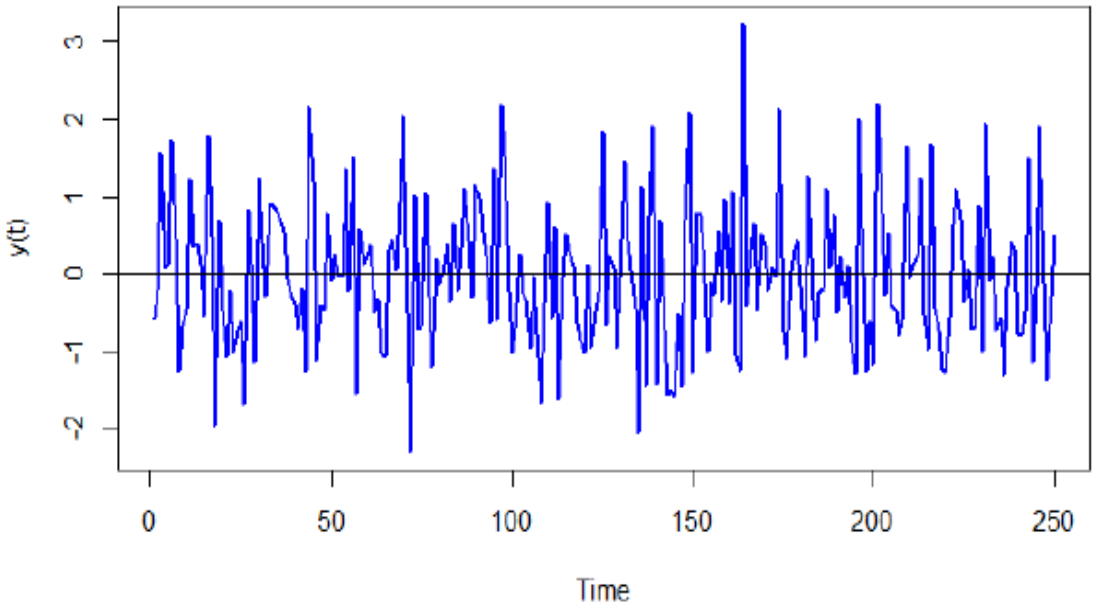
- Irregular patterns might occur due to random or unforeseen events
- They are often of short duration and non-repeating

Random or Irregular:

There is another factor which causes the variation in the variable under study. They are not regular variations and are purely random or irregular. These fluctuations are unforeseen, uncontrollable, unpredictable, and are erratic. These forces are earthquakes, wars, flood, famines, and any other disasters.

White Noise

A white noise series is one with a zero mean, a constant variance, and no correlation between its values at different times.



Since values are uncorrelated, the adjacent values do not help to forecast future values

Time series that show no autocorrelation are called **white noise**.

Just as correlation measures the extent of a linear relationship between two variables, autocorrelation measures the linear relationship between *lagged values* of a time series.

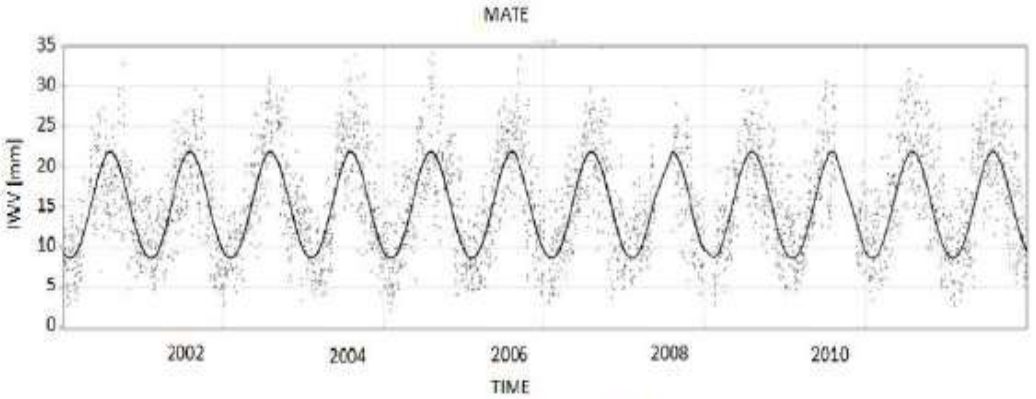
Stationarity

- Stationarity is rather a intuitive term, it means that the statistical properties of the process such as mean, variance and autocorrelation don't changes over time.
- A stationary time series is one whose properties do not depend on the time at which the series is observed.
- Thus, time series with trends, or with seasonality, are not stationary — the trend and seasonality will affect the value of the time series at different times.
- On the other hand, a white noise series is stationary — it does not matter when you observe it, it should look much the same at any point in time.

Stationarity

Criterion to classify a series as stationary

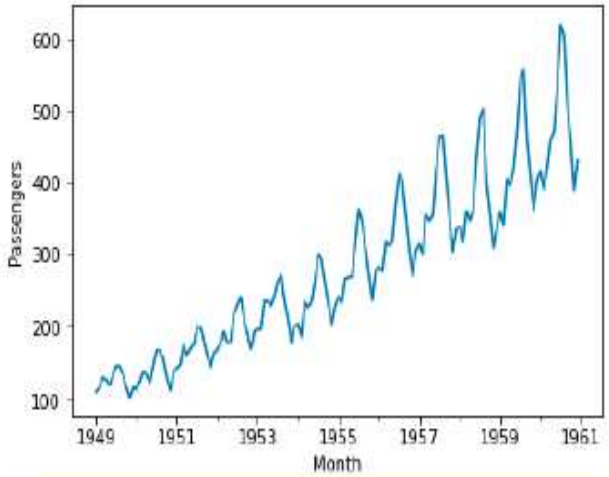
- Time Invariant(constant)
  - Mean
  - Variance
  - Autocorrelation



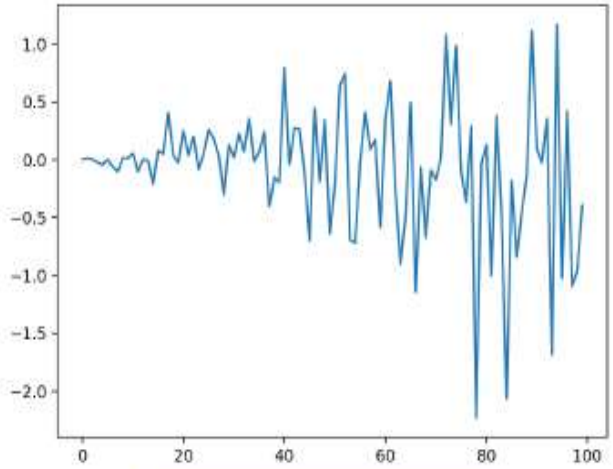
Stationary series

The time series should be stationary to build the model

Non-Stationary Series

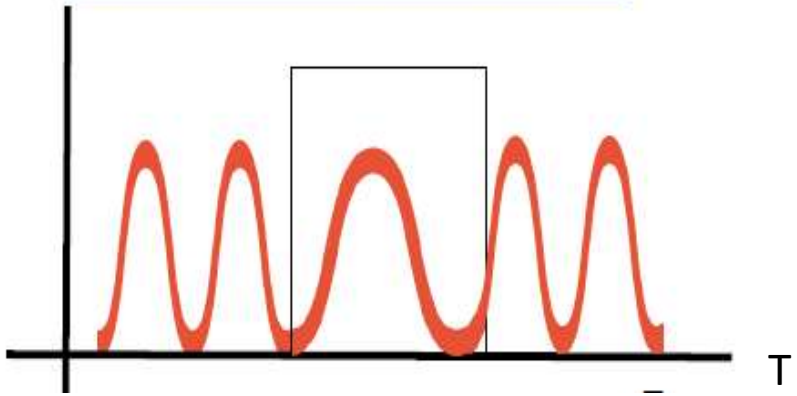


Increasing trend or non-constant **mean**



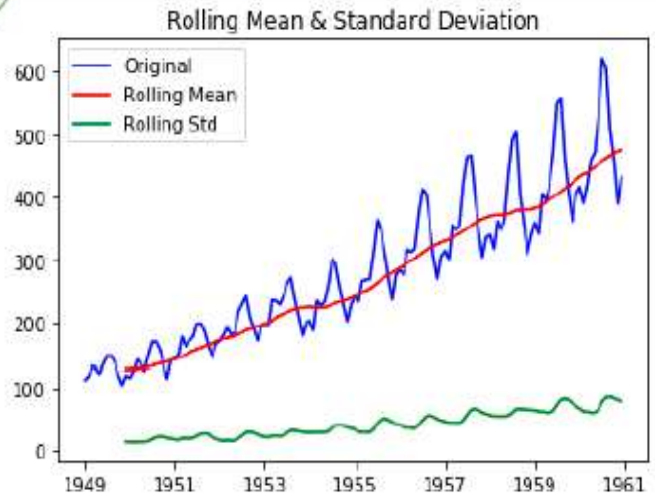
Non-constant **variance**

Co-variance is not constant with time



Stationarity Check

1 Rolling Statistics (Visual)



Plot the moving average or moving variance to check if it varies with time.

Notice the mean and variance **increase** constantly

Dickey Fuller test (Statistical) 2

Test Statistic	0.815369
p-value	0.991880
#Lags Used	13.000000
Number of Observations Used	130.000000
Critical Value (1%)	-3.481682
Critical Value (5%)	-2.884042
Critical Value (10%)	-2.578770
dtype: float64	

Null Hypothesis = TS is non-stationary

If 'Test Statistic' < 'Critical Value',
Reject the null hypothesis

Removal of Non-Stationarity

Differencing

Decomposition

Getting a TS perfectly stationary is desirable but not practical, so it is made as close as possible using these **statistical techniques**

Differencing

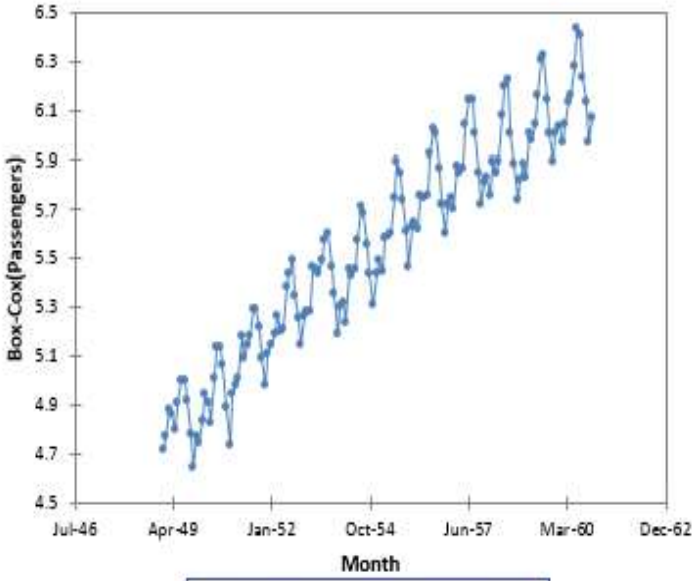
Differencing is performed by subtracting the previous observation from the current observation.

$$\Delta y_t = y_t - y_{t-1}$$

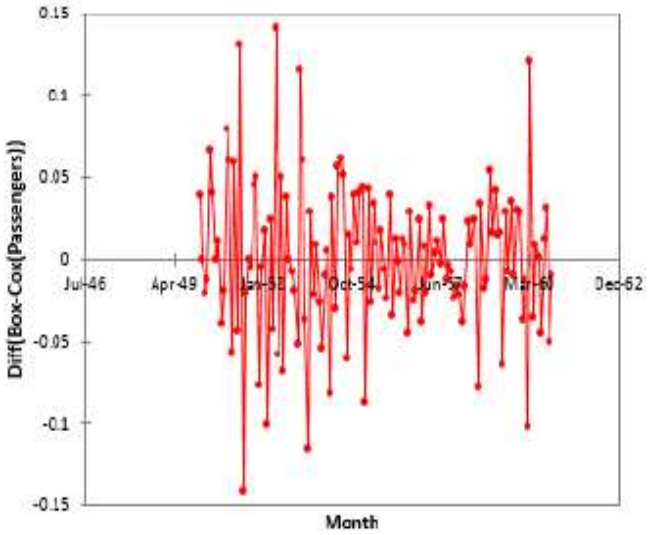
Δy_t is the difference between two successive values

Y_t is the value of y at t and Y_{t-1} is the value preceding Y_t

Differencing can help stabilize the mean of the time series by removing changes in the level of a time series, and so eliminating (or reducing) trend and seasonality.



Non-stationary series



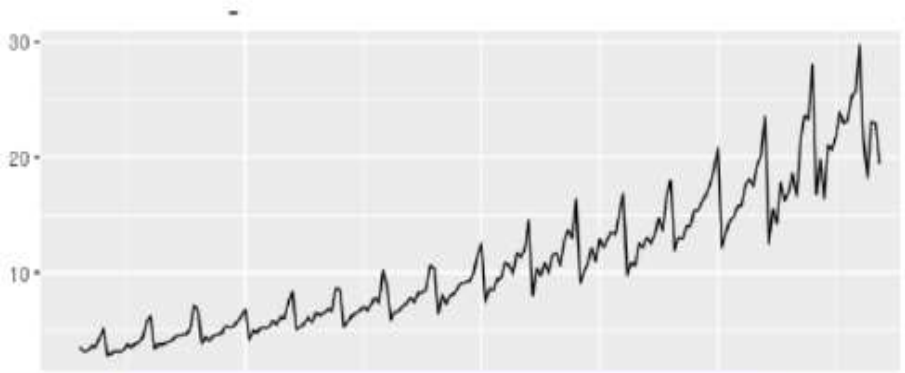
On differencing the series on left

Decomposition

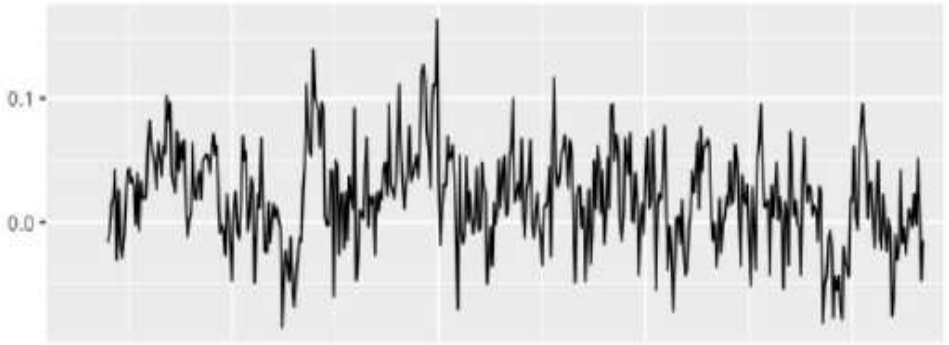
Detrending or de-seasonalizing eliminates the trend and seasonality respectively.

Decomposition is performed on the original series by regressing the series on time and taking the residuals from the regression.

$$y_t = \mu + \beta t + \epsilon_t$$



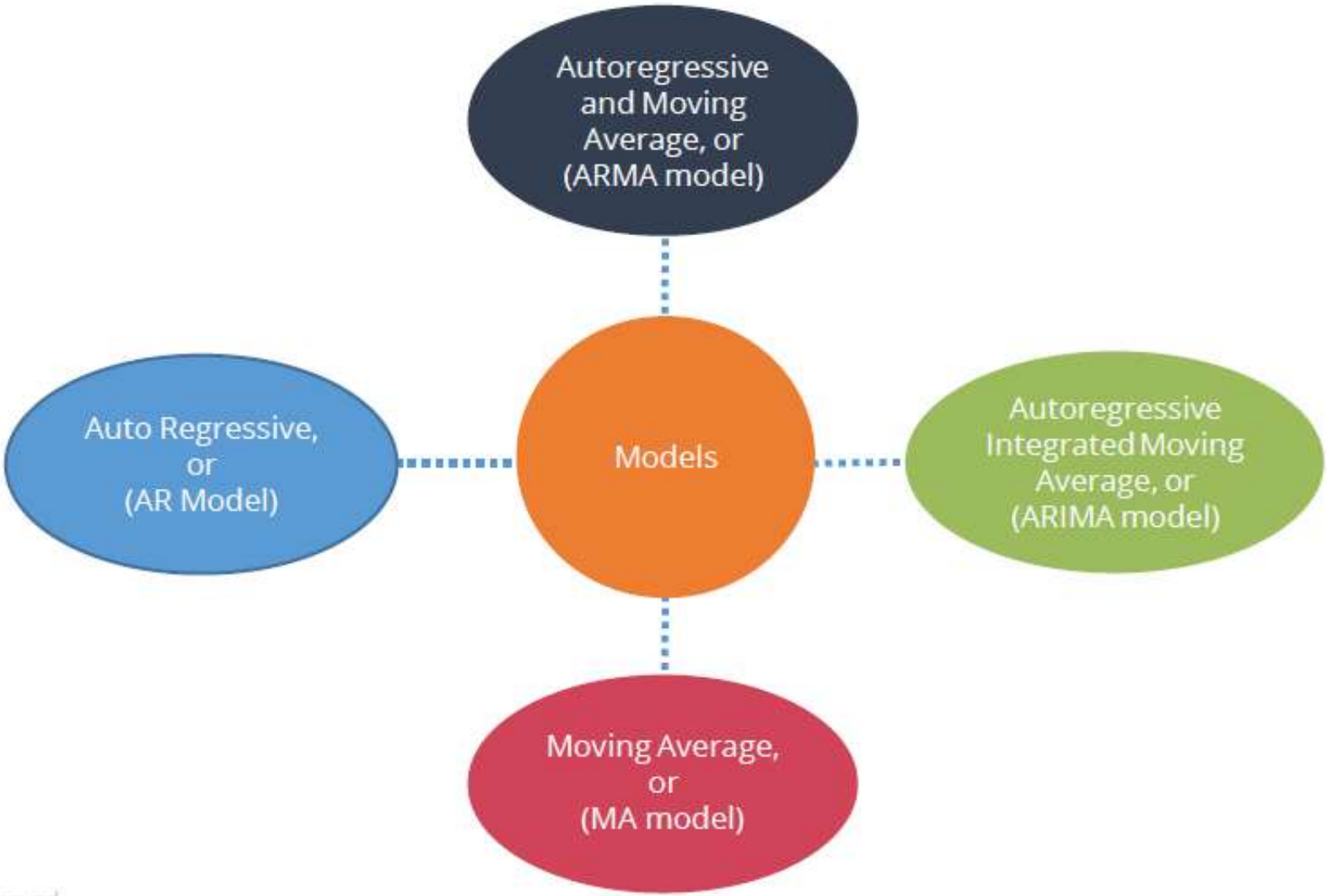
Seasonality with increasing trend



Seasonally decomposed series

i You can also use techniques like **transformation** which penalize higher values more than lower values. Example: square root, cube root, log.

Time Series Models



Auto Regressive(AR) Model

In a multiple regression model, we forecast the variable of interest using a linear combination of predictors. In an autoregression model, we forecast the variable of interest using a linear combination of *past values of the variable*. The term *auto*regression indicates that it is a regression of the variable against itself.

Thus, an autoregressive model of order p can be written as

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t,$$

where ε_t is white noise. This is like a multiple regression but with *lagged values* of y_t as predictors. We refer to this as an **AR(p) model**, an autoregressive model of order p.

The **order** of an auto regression is the number of immediately preceding values in the series that are used to predict the value at the present time.
For AR(1) the Model can be written as: $y_t = c + \phi_1 y_{t-1} + \varepsilon_t,$
For AR(2) the Model can be written as: $y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \varepsilon_t,$

Auto Regressive(AR) Model

For an AR(1) model:

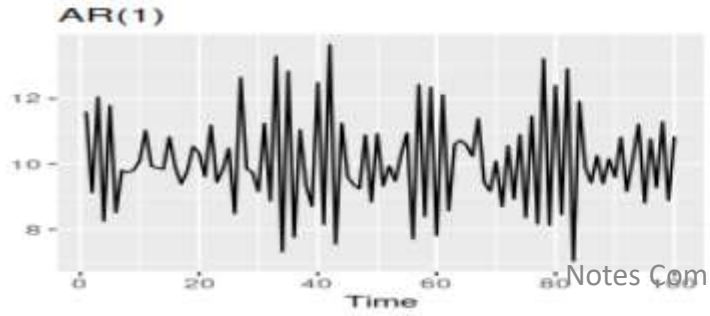
- when $\phi_1 = 0$, y_t is equivalent to white noise;
- when $\phi_1 = 1$ and $c = 0$, y_t is equivalent to a random walk;
- when $\phi_1 = 1$ and $c \neq 0$, y_t is equivalent to a random walk with drift;
- when $\phi_1 < 0$, y_t tends to oscillate around the mean.

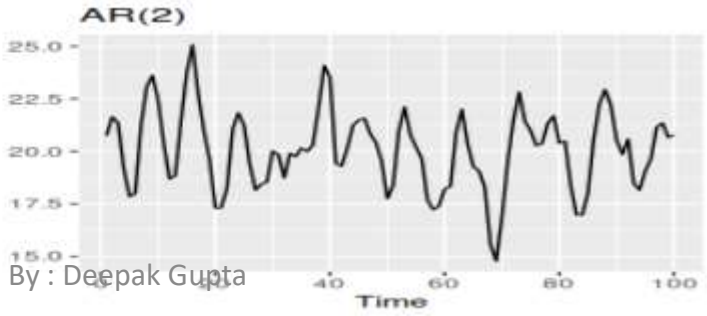
We normally restrict autoregressive models to stationary data, in which case some constraints on the values of the parameters are required.

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

When $p \geq 3$, the restrictions are much more complicated. R takes care of these restrictions when estimating a 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.





Auto Regressive (AR) Model

In an AR model, you predict future values based on a weighted sum of past values.

Equation for the auto regressive model :

$$Y_t = c + \varphi_1 Y_{t-1} + \varphi_2 y_{t-2} + \cdots + \varphi_p Y_{t-p} + e_t$$

Y_t is the function of different past values of the same variable
e_t is the error term
c is a constant
φ_1 to φ_p are the parameters

AR(1) is a model whose current value is based on the preceding value

AR(2) is based on the preceding two values

Day	Price	
1	21	y_{t-p}
2	22	.
3	23	.
4	24	.
5	23	.
6	26	.
7	27	.
8	27	.
9	29	y_{t-3}
10	30	y_{t-2}
11	32	y_{t-1}
12	?	y_t

Moving Average (MA) Model

MA model is used to forecast time series if Y_t depends only on the random error terms.

Equation for the MA model :

$$Y_t = \mu + \varphi_1 E_{t-1} + \varphi_2 E_{t-2} + \cdots + \varphi_p E_{t-p}$$

Y_t is the function of different past error terms
μ is the mean of the series
E_t is the error term
φ_1 to φ_p are the parameters

The error terms here are assumed to be white noise processes with mean zero and constant variance.

Year	Units	Moving Avg.
1994	2	—
1995	5	3
1996	2	3
1997	2	3.67
1998	7	5
1999	6	—

Moving Average(MA) Model

A moving-average process of order q, or MA(q), is a weighted sum of the current random error plus the q most recent errors.

Rather than using past values of the forecast variable in a regression, a moving average model uses past forecast errors in a regression-like model.

$$y_t = \mu + \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \cdots + \theta_q\varepsilon_{t-q},$$

where ε_t is white noise. We refer to this as an **MA(q) model**, a moving average model of order q. Of course, we do not *observe* the values of ε_t, so it is not really a regression in the usual sense.

The following time series is called first order moving average process, denoted by MR(1)

$$y_t = \mu + \varepsilon_t + \theta_1\varepsilon_{t-1}$$

μ is the mean of series & Θ is the constant term.
Here in MR(1),The term moving average comes from the fact that y_t is constructed from a weighted sum of two most recent values of ε.

Moving Average(MA) Model

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.

ARMA Model

ARMA model is used to forecast time series using both the past values and the error terms.

Equation for the ARMA model :

$$Y_t = c + \varphi_1 Y_{t-1} + \varphi_2 y_{t-2} + \cdots + \varphi_p Y_{t-p} + e \; + \; \mu + E_t + \varphi_1 E_{t-1} + \varphi_2 E_{t-2} + \text{---------} + \varphi_p E_{t-p}$$

Autoregressive part ⊕ Moving Average part = ARMA

It is referred as ARMA (p, q), where p is autoregressive terms and q is moving average terms

ACF and PACF

Autocorrelation refers to the way the observations in a time series are related to each other.

Autocorrelation Function (ACF)

ACF is the coefficient of correlation between the value of a point at a current time and its value at lag p, that is, correlation between Y(t) and Y(t-p)

ACF will identify the order of MA process

Partial Autocorrelation Function (PACF)

PACF is similar to ACF, but the intermediate lags between t and t-p are removed, that is, correlation between Y(t) and Y(t-p) with p-1 lags excluded.

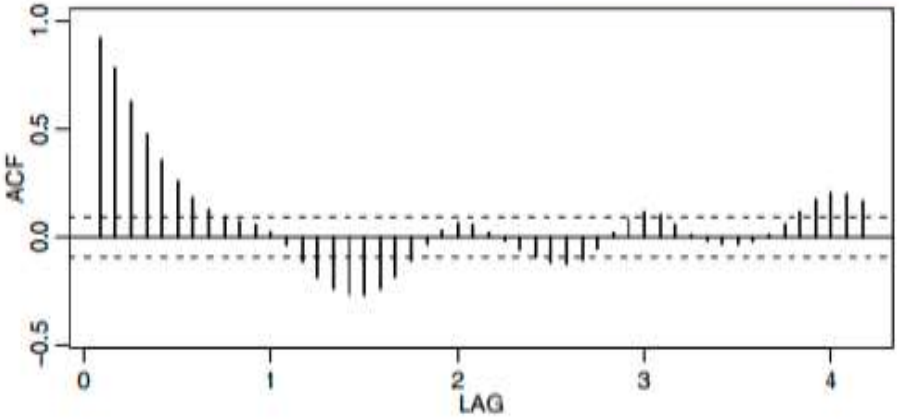
PACF will identify the order of AR process

ACF and PACF are used to determine the value of p and q

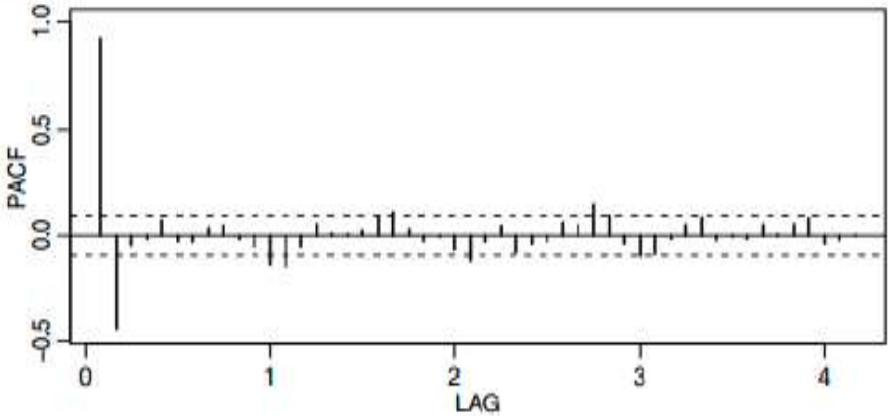
Notes Compiled By : Deepak Gupta

Characteristics of ACF and PACF

MODEL	ACF	PACF
AR(p)	Spikes decay towards zero	Spikes cutoff to zero
MA(q)	Spikes cutoff to zero	Spikes decay towards zero
ARMA(p,q)	Spikes decay towards zero	Spikes decay towards zero



ACF "decays" to zero



PACF "cuts off" to zero after the 2nd lag

Limitations of ARMA Model

- The limitation of ARMA model is the stationarity condition as AR and MA are two widely used linear models on stationary time series.
- Non stationary processes are common in many situations and these would at first appear outside the scope of ARMA model.

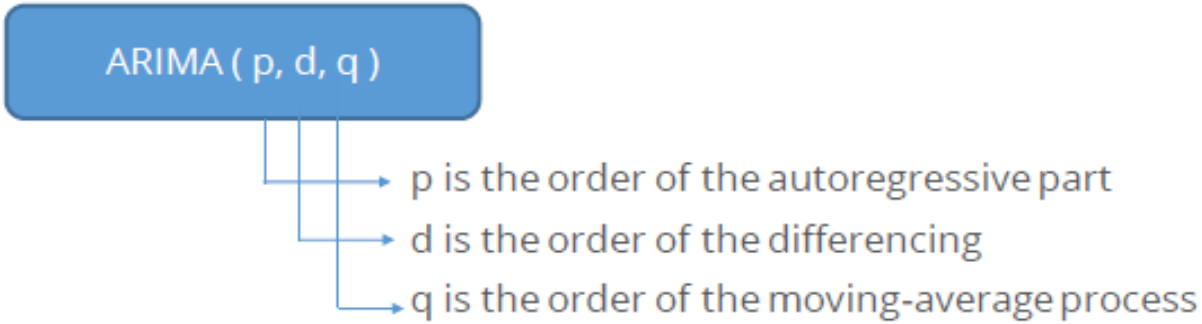
Autoregressive Integrated Moving Average(ARIMA) Model

- An autoregressive integrated moving average, or ARIMA, is a statistical analysis model that uses time series data to either better understand the data set or to predict future trends.

- An ARIMA model can be understood by outlining each of its components as follows:
  - ***Autoregression (AR)*** refers to a model that shows a changing variable that regresses on its own lagged, or prior, values.
  - ***Integrated (I)*** represents the differencing of raw observations to allow for the time series to become stationary, i.e., data values are replaced by the difference between the data values and the previous values.
  - ***Moving average (MA)*** incorporates the dependency between an observation and a residual error from a moving average model applied to lagged observations.
- In an autoregressive integrated moving average model, the data are differenced in order to make it stationary. A model that shows stationarity is one that shows there is constancy to the data over time. Most economic and market data show trends, so the purpose of differencing is to remove any trends or seasonal structures.

ARIMA Model

ARIMA model predicts a value in a response time series as a linear combination of its own past values, past errors, also current and past values of other time series.



The full model can be written 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 y'_t is the differenced series (it may have been differenced more than once). The "predictors" on the right hand side include both lagged values of y_t and lagged errors.

If no differencing is done (d = 0), the models are usually referred to as ARMA(p, q) models

Steps in Time Series Forecasting

Step 01	Visualize the time series – check for trend, seasonality, or random patterns
Step 02	Stationarize the series using decomposition or differencing techniques
Step 03	Plot ACF / PACF and find (p, d, q) parameters
Step 04	Build ARIMA model
Step 05	Make predictions using final ARIMA model

ARIMA vs ARMA

- What sets ARMA and ARIMA apart is *differencing*.
- An ARMA model is a stationary model;
- If your model isn't stationary, then you can achieve stationarity by taking a series of differences. The "I" in the ARIMA model stands for integrated; It is a measure of how many non seasonal differences are needed to achieve stationarity.
- If **no differencing is involved in the model**, then it becomes simply an ARMA.
- A model with a *d*th difference to fit and ARMA(p,q) model is called an ARIMA process of order (p,d,q).

Non linear Dynamics basics

- A dynamical system is something whose behaviour evolves with time: population growth, binary stars, transistor radios, predator-prey populations, differential equations, the air stream past the cowl of a jet engine.

- The dynamics of the system is the set of laws or equations that describes how the state changes over time. Usually this set consists of a system of coupled differential equations.

- A dynamical system is linear if all their equations are linear, otherwise its non linear. If it's linear small cause can have small effects, if it's non linear small cause can have large effects.

- In linear systems, it is often safe to assume, and easy to recognize, that the "important" parts of the signal are lower down on the frequency scale and easily separable from the noise (which is assumed to be high frequency), and it is easy to implement digital filters that remove components of a signal above a specified cut off frequency.

- In nonlinear systems, the important parts of the signal often cover the entire spectrum, making signal separation a difficult proposition. Nonlinearity is even more of a hurdle in system identification.

Non linear Dynamics basics

- The nonlinear dynamics community, relies primarily upon the *state-space* representation, plotting the behaviour on the n-dimensional space, whose axes are the state variables.

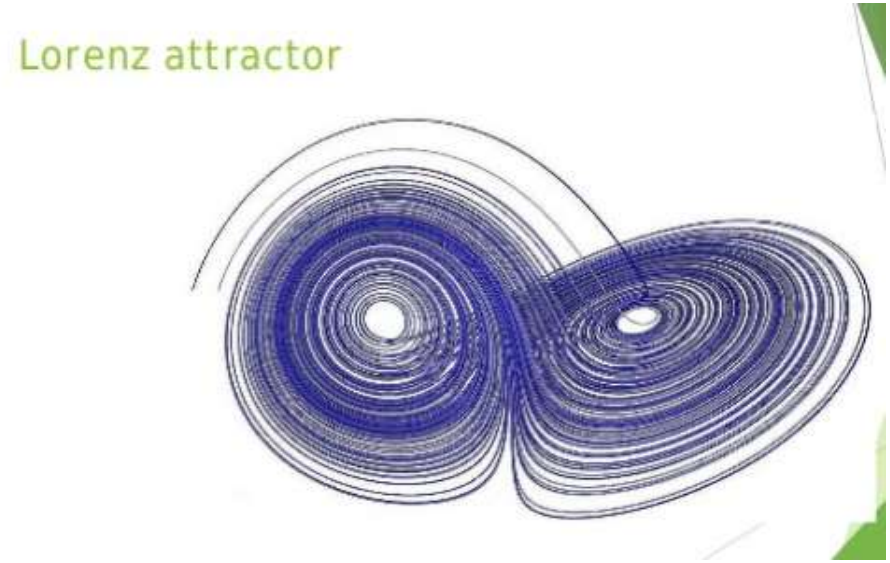
- The *state variables* of a dynamical system are the fundamental quantities needed to fully describe it.

- The focus in nonlinear time series analysis lies therefore not on predicting single trajectories, but on estimating the totality of possible states a system can attain and their statistical properties, i.e., how often the system can be expected to be in a particular state. Of particular importance hereby is the long-term behaviour of the system, the so-called *attractor*, which can roughly be defined as the set of all *recurrent* states of the system.

- In dynamical systems, a trajectory is the set of points in state space that are the future states resulting from a given initial state. In a discrete dynamical system, a trajectory is a set of isolated points in state space. In a continuous dynamical system, a trajectory is a curve in state space.

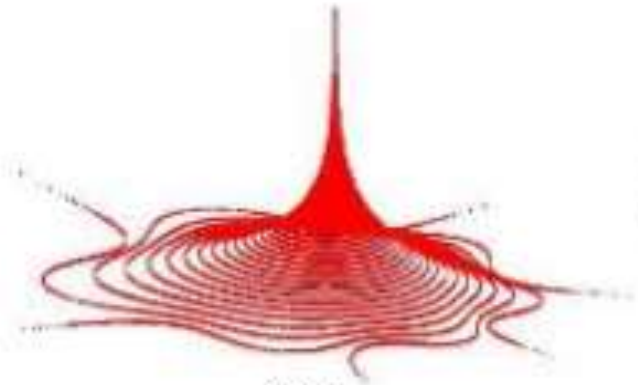
Non linear Dynamics basics

- The number *n* of state variables is known as the *dimension* of the system; a pendulum or a mass on a spring is a two-dimensional system, while a three-capacitor circuit has three dimensions.
- If the number of state variables in the system is infinite—e.g., a moving fluid, whose physics is influenced by the pressure, temperature and velocity at *every point*—the system is called *spatiotemporally extended*, and one must use *partial differential equation* (PDE) models to describe it properly.

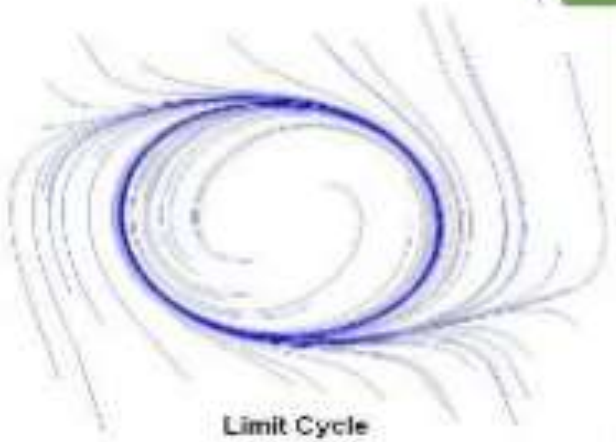


Non linear Dynamics basics

Attractors and their properties



Point



Limit Cycle

Torus Attractor

Strange Attractor

Rule Induction

- Machine learning is concerned with the question of how to construct a computer progr.am that automatically learns new facts and theories from data.
- Rule induction is a special kind of machine learning technique that results from specific cases to general principles that are expressible as if-then rules.
- A number of rule induction systems have been constructed and applied to discover knowledge from data in different applications.

- Since regularities hidden in data are frequently expressed in terms of rules, rule induction is one of the fundamental tools of data mining at the same time. Usually rules are expressions of the form

$$if\ (attribute-1, value-1)\ and\ (attribute-2, value-2)\ and\ \cdots$$
$$and\ (attribute-n, value-n)\ then\ (decision, value).$$

- Data from which rules are induced are usually presented in a form similar to a table in which *cases* (or *examples*) are *labels* (or *names*) for rows and variables are labelled as *attributes* and a *decision*.

Rule Induction

- **Rule induction** is an area of machine learning in which formal **rule**s are extracted from a set of observations. The **rule**s extracted may represent a full scientific model of the data, or merely represent local patterns in the data.

- It is the extraction of useful if-then **rule**s from data based on statistical significance.

- It is common in machine learning to distinguish symbolic and sub-symbolic approaches.

- Symbolic approaches employ some kind of description language in which the learned knowledge is expressed.

- The area of machine learning is called *classification rule induction where we* construct explicit symbolic classification rules that generalise the training cases, and are thus instances of symbolic machine learning.

- The classification rule learning task can be defined as follows: Given a set of training examples (instances for which the classification is known), find a set of classification rules that can be used for prediction or classification of new instances, i.e., cases that haven't been presented to the learner before.

Rule Induction

- A more formal definition of the classification rule learning task has to take into account the restrictions imposed by the language used to describe the data (data description language) and the language used to describe the induced set of rules (hypothesis description language). The *language bias* refers to the restrictions imposed by the languages defining the format and scope of data and knowledge representation.

- A generic classification rule learning task can now be defined, given a binary classification problem of classifying instances into classes named *positive* and *negative.* The task of learning a set of rules defining the class *positive* is defined as follows.

Given:

— a data description language, imposing a bias on the form of data

— training examples, i.e., a set of classified instances described in the data description language

— a hypothesis language, imposing a bias on the form of induced rules

— a coverage function, defining when an instance is covered by a rule

Find:

— a hypothesis as a set of rules described in the hypothesis language, that is

consistent, i.e., does not cover any negative example

complete, i.e., covers all positive examples

Rule Induction

- The objective is to provide an overview of two main approaches to classification rule induction:
  - Propositional rule learning
  - Relational rule learning.

- This simplest setting of classification rule induction is usually called *propositional rule induction* or *attribute-value rule learning*.

- A more elaborate setting, where data is represented in several tables, and the output are relational rules, possibly in the form of Prolog clauses. This approach is usually called *induction of relational rules, multi-relational data mining,* or *inductive logic programming* (ILP).

Rule Induction: Sequential Covering Algorithm

Sequential-Covering(*Target_attribute, Attributes, Examples, Threshold*)

1.Learned_rules ←{ }

2. Rule ← Learn-one-rule(*Target_attribute*, *Attributes*, *Examples*)

3.While Performance(*Rule, Examples*) > *Threshold*, do

Learned_rules ← *Learned_rules* + *Rule*

Examples ← *Examples* -{examples correctly classified by *Rule*}

Rule ← Learn-one-rule(*Target_attribute*, *Attributes*, *Examples*)

4.Learned_rules ← sort *Learned_rules* according to Performance over *Examples*

5.Return *Learned_rules*

Rule Induction: Sequential Covering Algorithm

LEARN-ONE-RULE($Target\_attr, Attrs, Examples$)

- $Pos \leftarrow$ positive $Examples$; $Neg \leftarrow$ negative $Examples$
- If Pos

$NewRule \leftarrow$ most general rule possible; $NewRuleNeg \leftarrow Neg$

While $NewRuleNeg$

1. $Candidate\_literals(CLs) \leftarrow$ generate candidates
2. $Best\_literal \leftarrow argmax_{L \in CLs}$ $PERFORMANCE(Specialize(NewRule, L))$
3. add $Best\_literal$ to $NewRule$ preconditions
4. $NewRuleNeg \leftarrow$ subset of $NewRuleNeg$ that satisfies $NewRule$ preconditions

- Return $NewRule$

Learning Propositional Rules: *Learn-one-rule*

General-to-Specific Search:

1. Consider the most general rule (hypothesis) which matches every instances in the training set.

2. Repeat
   Add the attribute that most improves rule performance measured over the training set.

3.Until the hypothesis reaches an acceptable level of performance.

General-to-specific search is very well suited for learning in the presence of noise because it can easily be guided by heuristics.

General-to-Specific Beam Search (CN2):
Rather than considering a single candidate at each search step, keep track of the ***k*** best candidates.

Rule Induction: Sequential Covering Algorithm

Common Performance Metrics

Entropy: S = examples that match the rule's preconditions.

$$-Entropy(S) \equiv \sum_{i=1}^{c} x_i \log_2 x_i$$

Relative Frequency:

$$\frac{n_c}{n}$$

n = # examples the rule matches
n_c = # examples the rule matches and classifies correctly

---

m estimate:

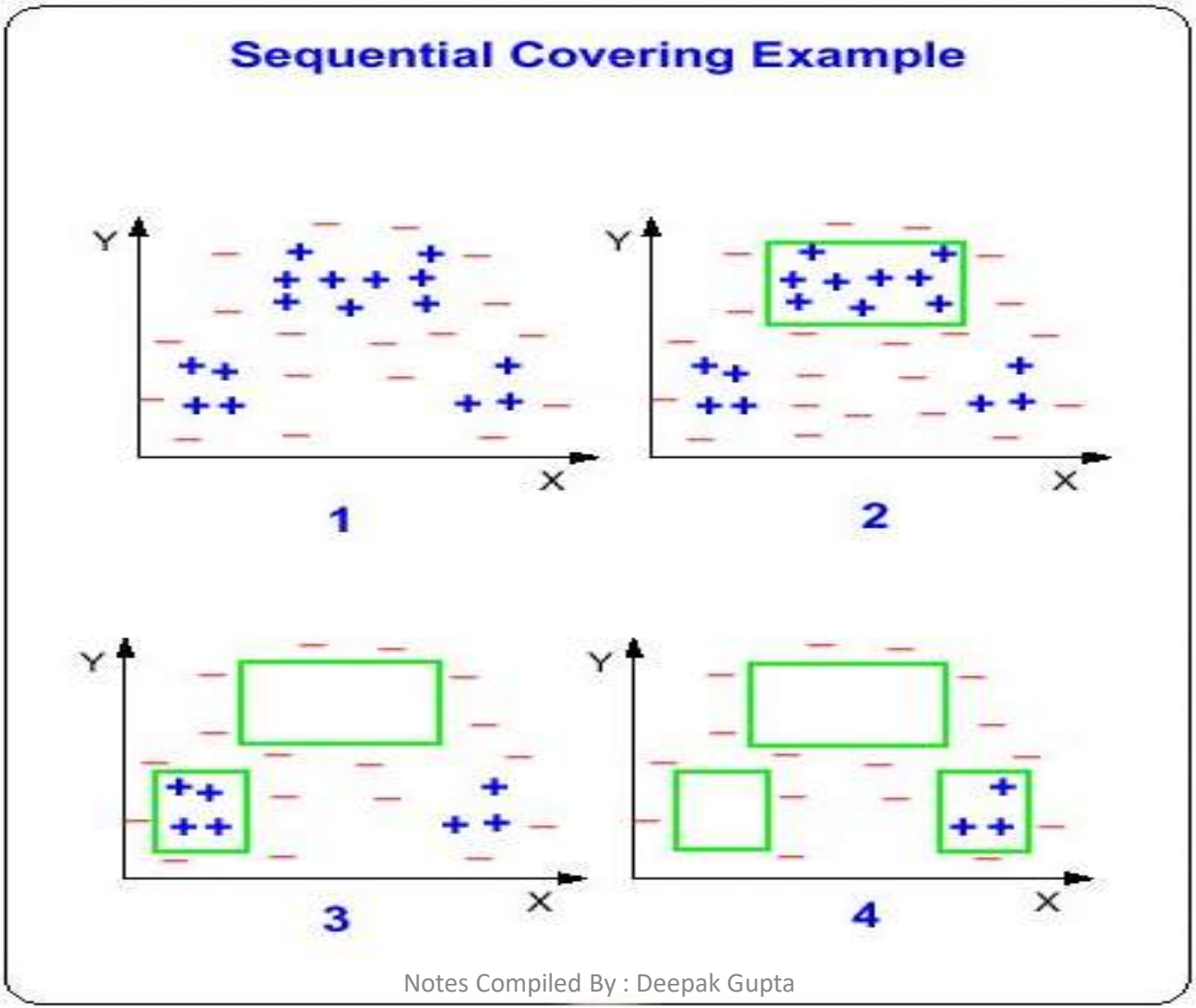
$$\frac{n_c + mp}{n + m}$$

p = prior probability of the class assigned by the rule
m = # examples needed to override the prior

Properties:

One characteristic of this algorithm is it requires **high accuracy**, meaning the prediction should be correct with high probability. Another is that it is possibly **low coverage**, meaning that it does not make prediction for all examples. There might be some examples that do not classified by the algorithm.

Rule Induction: Sequential Covering Algorithm



Rule Induction: Sequential Covering Algorithm

Id	Size	Colour	Shape	Weight	Expensive
1	Big	Red	Square	Heavy	Yes
2	Small	Blue	Triangle	Light	Yes
3	Small	Blue	Square	Light	No
4	Big	Green	Triangle	Heavy	No
5	Big	Blue	Square	Light	No
6	Big	Green	Square	Heavy	Yes
7	Small	Red	Triangle	Light	Yes

The final set of rules is like follow:

Expensive = Yes if:

Colour = Red.	(covers example 1,7)
Or (Colour = Green & Shape = Square).	(covers example 6)
Or (Colour = Blue & Shape = Triangle).	(covers example 2)

Learning First Order Logic

The propositional logic has very limited expressive power. Unfortunately, in propositional logic, we can only represent the facts, which are either true or false. PL is not sufficient to represent the complex sentences or natural language statements.

Consider the following sentence,

- **"Some humans are intelligent", or**
- **"Sachin likes cricket.“**

To represent the above statements, PL logic is not sufficient, so we required some more powerful logic, such as first-order logic.

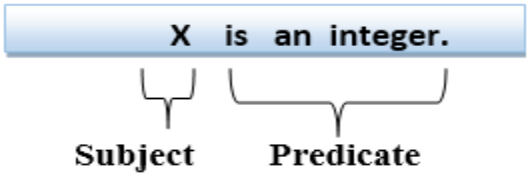
It is an extension to propositional logic. FOL is sufficiently expressive to represent the natural language statements in a concise way.

Elements of First Order Logic

Basic Elements of First-order logic:

Constant	1, 2, A, John, Mumbai, cat,....
Variables	x, y, z, a, b,....
Predicates	Brother, Father, >,....
Function	sqrt, LeftLegOf, ...
Connectives	∧, ∨, ¬, ⇒, ⇔
Equality	==
Quantifier	∀, ∃

First-order logic statements can be divided into two parts:



•**Subject:** Subject is the main part of the statement.

•**Predicate:** A predicate can be defined as a relation, which binds two atoms together in a statement.

Consider the statement: "x is an integer.", it consists of two parts, the first part x is the subject of the statement and second part "is an integer," is known as a predicate.

Elements of First Order Logic

Induction is reasoning from the specific to the general.
For example, consider the following dataset

```
parent(a,b)     parent(a,c)     parent(d,b)
father(a,b)     father(a,c)     mother(d,b)
male(a)         female(c)       female(d)
```

Given the above dataset, we can use inductive reasoning to infer the following rules (or view definitions):

```
father(X,Y) :- parent(X,Y) & male(X)
mother(X,Y) :- parent(X,Y) & female(X)
```

In inductive logic programming, given a dataset, a set of starting view definitions, and a target predicate, we can infer the view definition of the target predicate.

In the example above, we were given a dataset, no starting view definitions, and we inferred the view definition of "Father" and "Mother".
In the context of the inductive logic programming, the dataset is also referred to as a set of positive examples. Set of positive and negative examples taken together is also known as training data.

First Order Inductive Logic

- Inductive logic programming is the subfield of machine learning that uses first-order logic to represent hypotheses and data. Because first-order logic is expressive and declarative, inductive logic programming specifically targets problems involving structured data and background knowledge.
- Inductive logic programming tackles a wide variety of problems in machine learning, including classification, regression, clustering, and reinforcement learning, often using "upgrades" of existing propositional machine learning systems. It relies on logic for knowledge representation and reasoning purposes.
- The general approach used in an inductive learner is to start from the predicate whose definition is to be learned as the head of a rule whose body is initialized to be empty.
- At each step, we add a literal to the body of the rule so that it satisfies several positive examples and none of negative examples. The literal to be added can be one of the predicates from the problem statement, the negation of a predicate from the problem statement, equality between two bound variables and inequality between two bound variables. Recursive literals are allowed if they will not cause an infinite regression.
- For choosing between alternative literals, a heuristic measure is used. An example pseudo code of an inductive learner is shown below.

FOIL: First Order Inductive Logic

```
foil(positive, negative, predicate)
    clauses ← ∅
    repeat
     clause ← learnNewClause(positive, negative, target)
     positive ← positive - all positive examples satisfied by the clause
     clauses ← clauses + clause
    until positive examples = ∅
    return clauses

learnNewClause(positive, negative, predicate)
    clause ← ∅
    repeat
     literal ← choose_literal(clause, positive, negative)
     clause ← clause ∪ literal
     negative ← negative - negative examples satisfied by literal
    until negative examples = ∅
    return clause
```

FOIL: First Order Inductive Logic

To understand the working of this algorithm, let us consider how it learns the rule `father(X,Y) :- parent(X,Y) & male(X)`. Two positive examples, and the relevant background predicates are as listed at the start of this section. There are ten negative examples are listed below.

```
father(a,d)    father(b,a)    father(b,c)
father(b,d)    father(c,a)    father(c,b)
father(c,d)    father(d,a)    father(d,b)
father(d,c)
```

We start a new clause with `father(X,Y)` in the head, and with an empty body. Here are the possible candidates of literals to be added to the body.

```
male(X)        male(Y)        female(X)
female(Y)      parent(X,Y)    father(X,Z)
father(Y,X)    father(Y,Z)    father(Z,X)
father(Z,Y)
```

To ensure the safety of the rules, we have not considered any negated literals, or a literal that checks for equality of inequality of unbound variables. Furthermore, we only consider those literals that share at least one variable with the predicate that we are defining.