**AI**

In **1956 John McCarthy** (**Father of AI**) coined the term AI.

In **1959 Arthur Samuel** coined the term ML.

In 1961 first Chat bot called Eliza was introduced.

**AI** – The theory and development of computer systems able to perform tasks nominal requiring human intelligence such as visual perception, speech recognition, decision making and translation between languages.

Types 1 –

1. **Artificial narrow intelligence** – Alexa, Sophia, Google Search Engine – Also known as weak AI involves applying AI only to specific tasks.
2. **Artificial General intelligence** – Alpha Go - Also known as strong AI involves machines that possess the ability to perform any intellectual task that a human being can.
3. **Artificial super intelligence** – is a term referring to the time when the capability of computers will surpass Humans.

Types 2 -

1. **Purely Reactive** - Chess-playing programs like IBM's Deep Blue, which analyse possible moves based on a predefined rules and historical data but do not learn or adapt during gameplay.
2. **Limited Memory** -Self-driving cars use limited memory AI to analyse and respond to real-time traffic conditions based on past data and experiences. They can learn and adapt, but their memory is not as extensive as more advanced models.
3. **Theory Of Mind** -This type of AI, often associated with advanced artificial general intelligence (AGI), is hypothetical at this point. It refers to AI systems that have the ability to understand and interpret human emotions, beliefs, intentions, and other mental states.
4. **Self-Aware AI** - would have consciousness and the ability to understand its own existence, emotions, and thoughts. This concept is more theoretical and is not yet realized in practical applications.

Programming Language for AI

**Python** (Supports Procedure-oriented programming and OOP), **Java**,

**R** for Statistical things, **Lisp** (processing symbolic information it has excellent prototyping capability – made by John McCarthy), **SWI Prolog** (Tree based data structuring, Pattern matching – used in Medical Project).

**Machine Learning** is a subfield of artificial intelligence (AI) that focuses on the development of algorithms and statistical models that enable computer systems to perform a task without being explicitly programmed for that task. The key idea behind machine learning is to allow computers to learn from data and improve their performance over time.

**Need for ML**- Increase in Data generation, Improve Decision Making, Uncover Patterns and Trends in Data, Solve Complex Problems.

**ML Algorithms** – A set of rules and statistical techniques used to learn patterns from data.

**ML Model** – A model is trained by using MLA.

**Predictor Variable** – It is a feature of the data that can be used to predict the output. Ex – Predicting the Height of the person with respect to its weight so weight her is PV.

**Response variable** – It is the feature of the output variable that needs to be predicted by using the predictor variable.

The input data is divided into two parts this is called data splicing -

**Training Data** – The machine learning murder is built using the training data.

**Testing data** - The machine learning model is evaluated using the testing data.

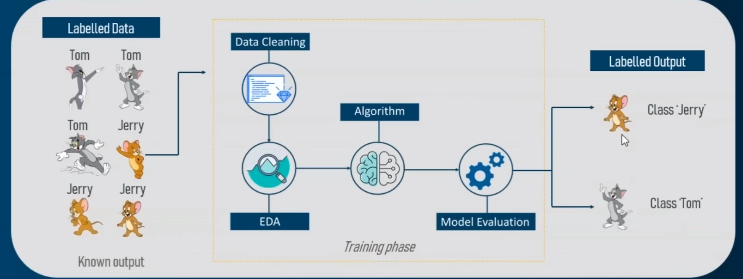
Training Data is more than Testing Data.

The machine learning process involves building a productive model That can be used to find a solution for a problem statement.

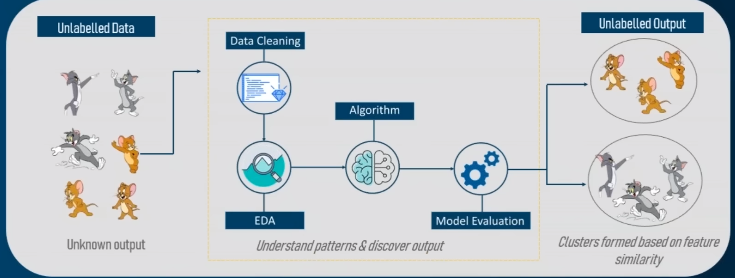
**Process to solve Problem in ML** – Define objective, data gathering, preparing data (cleaning data), data exploration, Building a model (Training Data is used, Predictive Model is built using Algos -Linear regression, Decision Trees), model evaluation (Testing Data is used to test the model), prediction.

**Types of ML** –

**Supervised Learning** – is a technique in which we teach or train the machine using data which is well labelled. **Ex** – Data about Tom and Jerry are given to a machine, as the data is labelled it will recognise them and out them in 2 different classes.

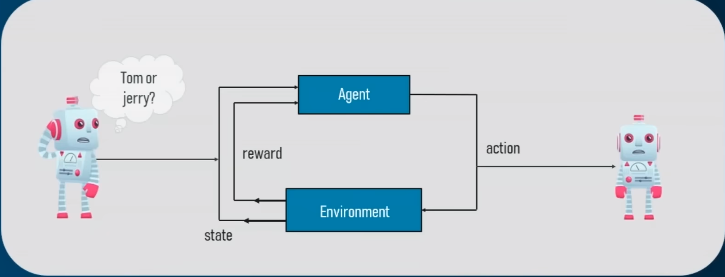


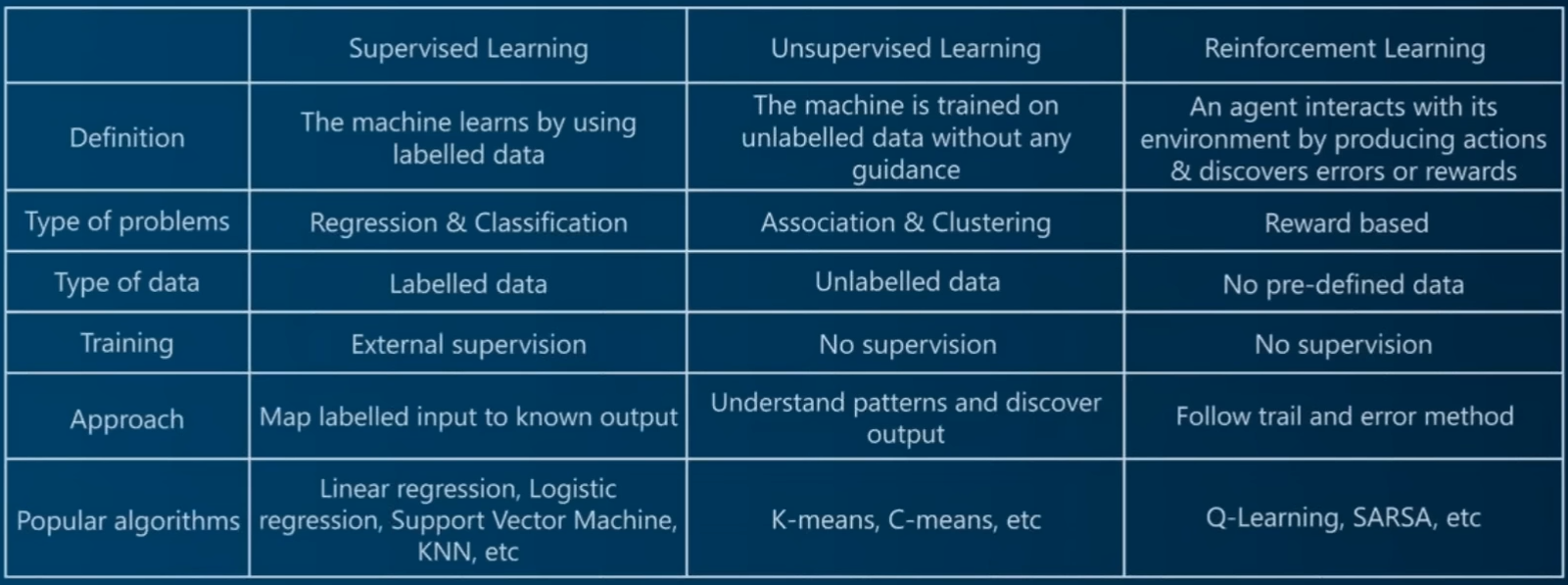
**Unsupervised Learning** – is the training of machine using information that is unlabelled and allowing the algorithm to act on that information without guidance. **Ex** – Machine will be given 2 different picture and now it has to label who is who on its own.



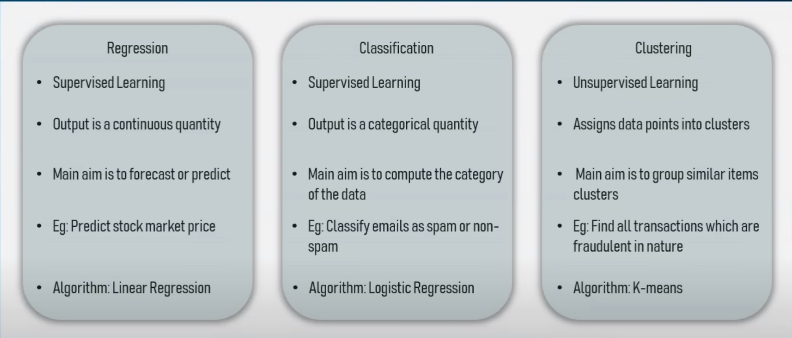
**Reinforcement Learning** – It is a part of ML where an agent is put in an environment and he learns to behave in this environment by performing certain actions and observing the rewards which it gets from those actions.

**Ex** – Going to a deserted Island and learning all the things on our own.



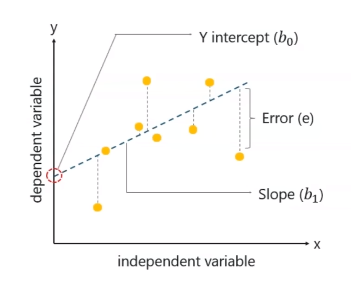
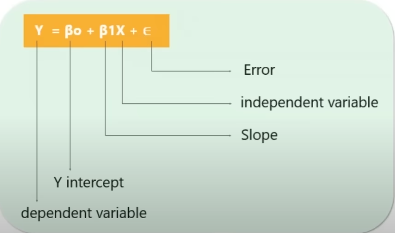


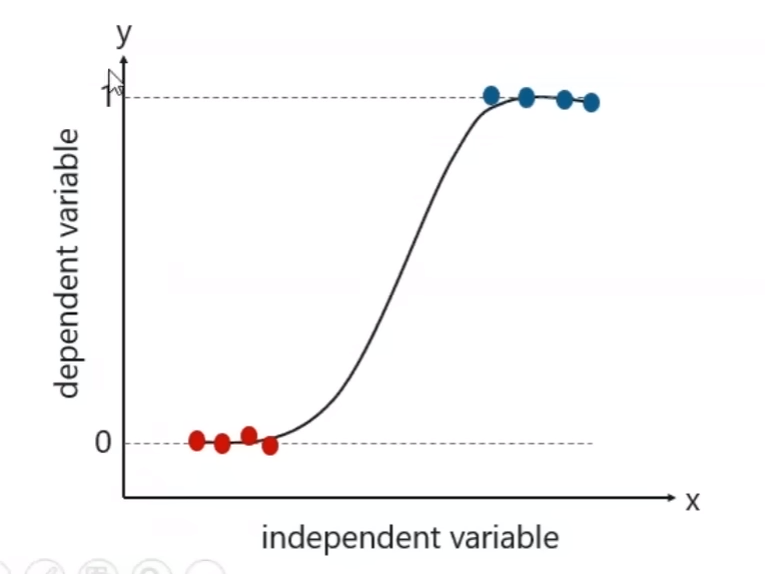
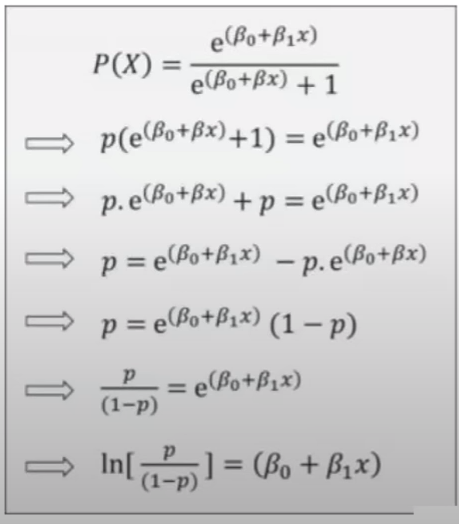
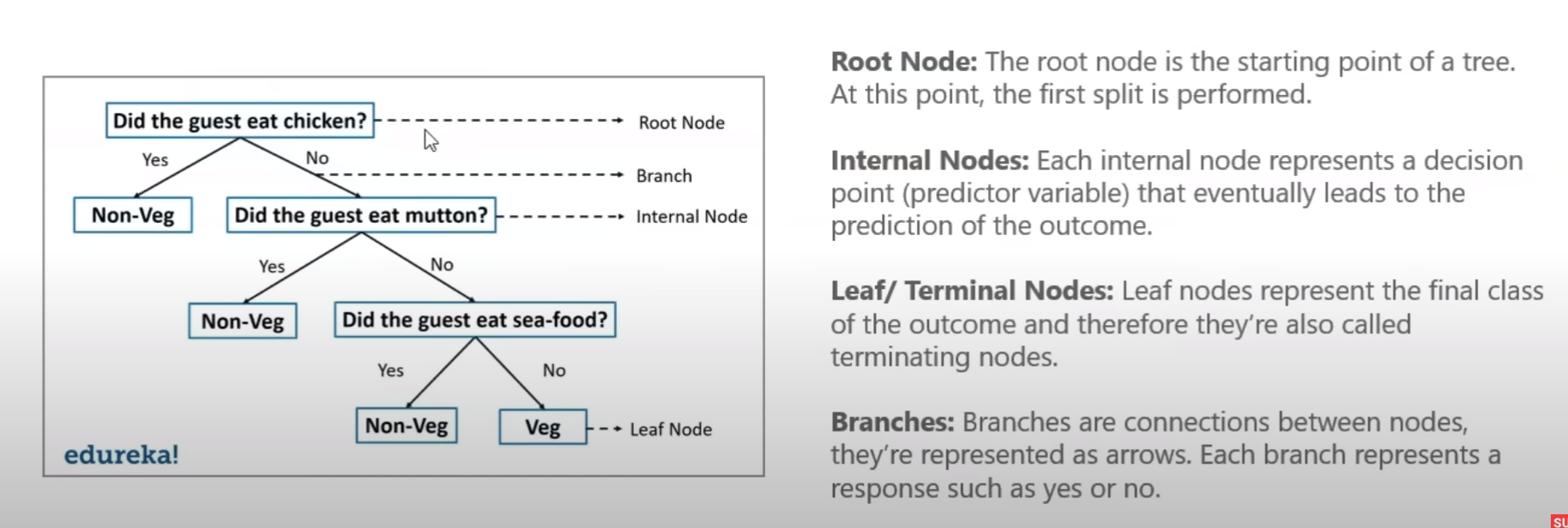
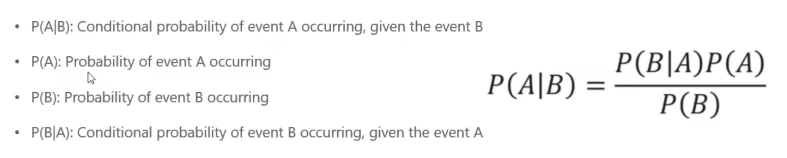
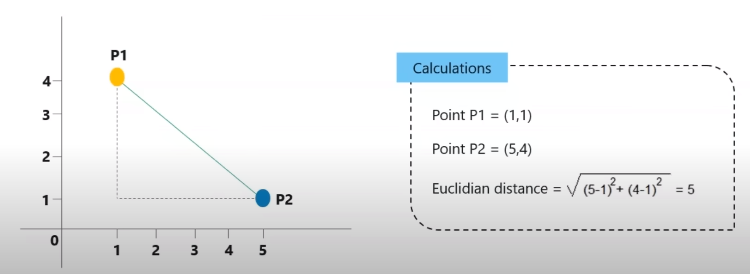
Problems in ML can be put in 1 of these 3 categories –

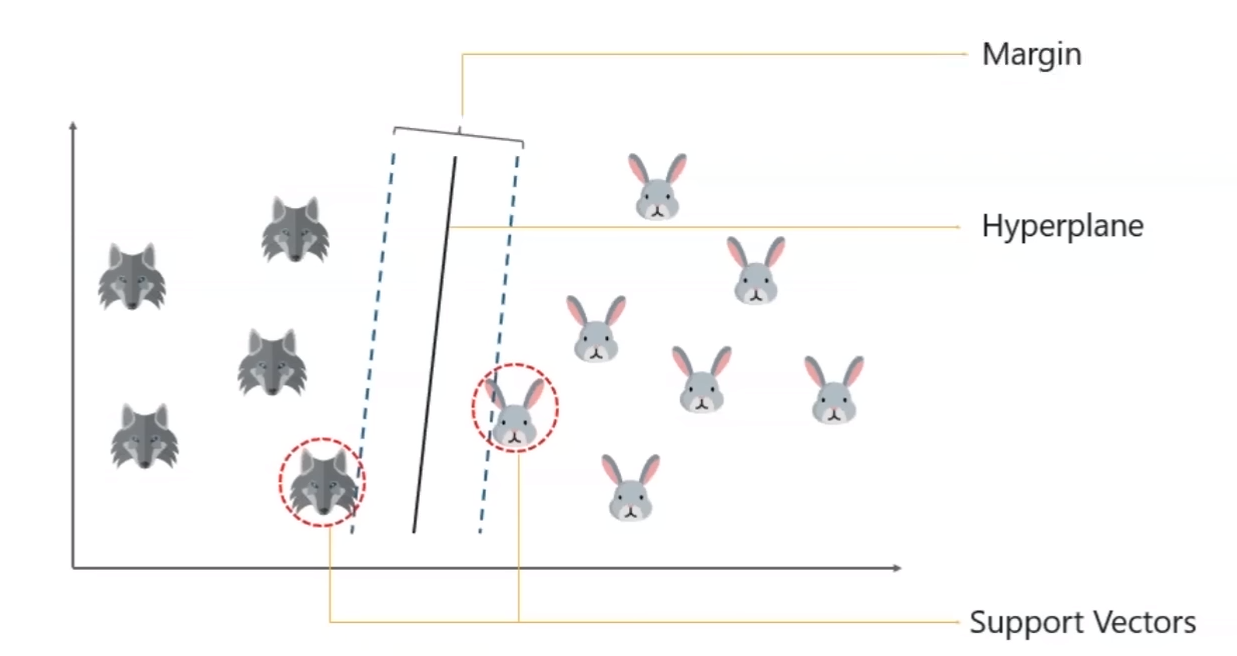


Supervised Learning Algorithms –

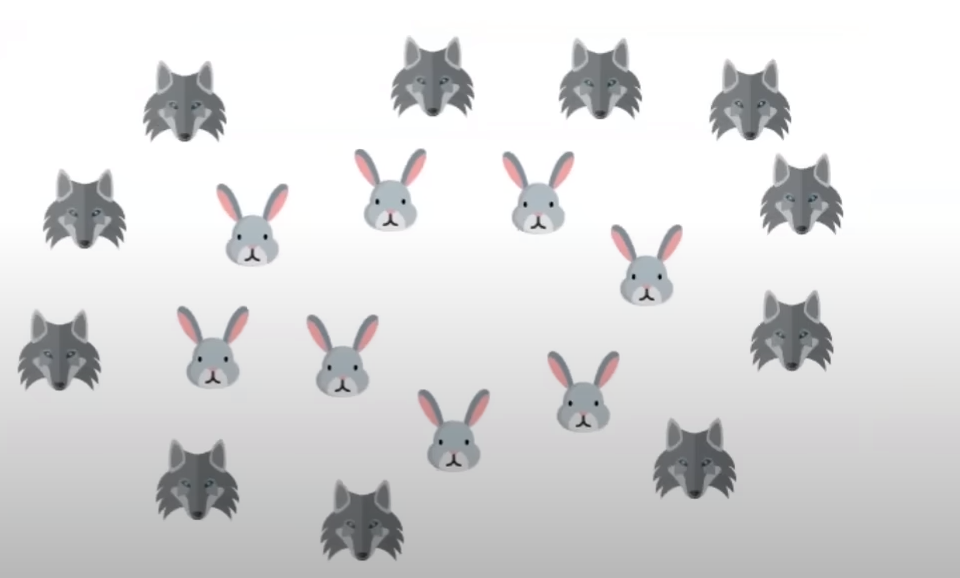
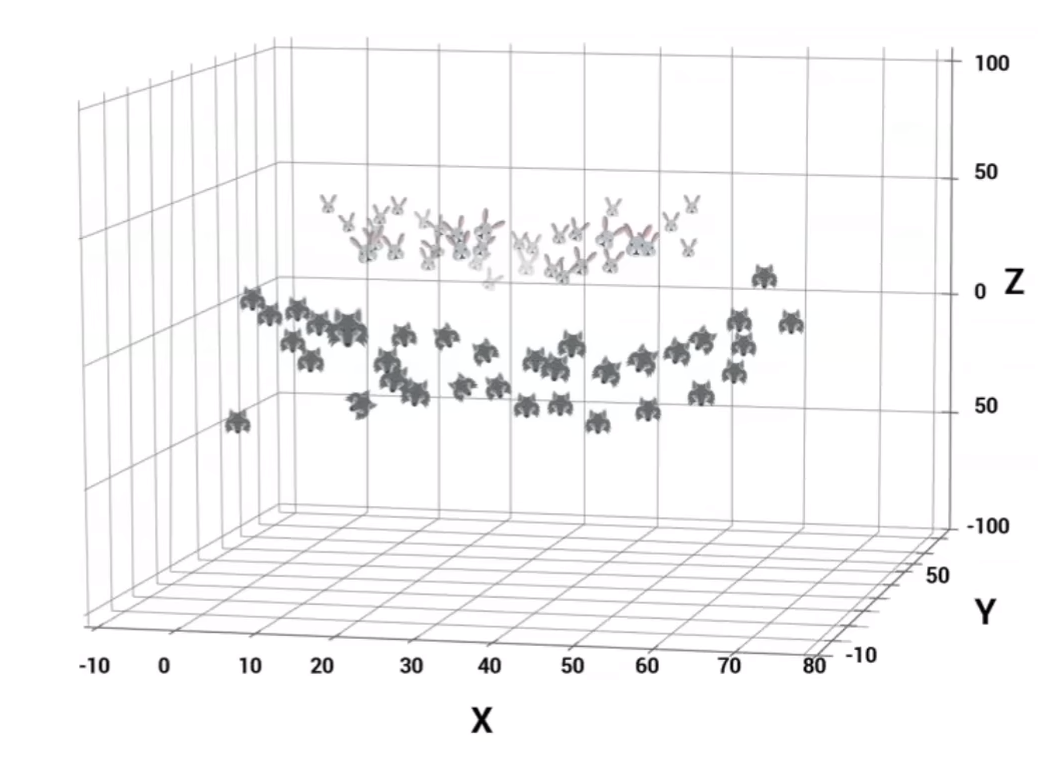
1. **Linear Regression** – It is a **Regression Algorithm**. It is a method to predict dependent (Output) variable Y based on values of independent (Input) variables X. It can be used for the case where we want to predict some continuous quantity. A continuous quantity means a quantity which is infinite, example weight of a person can be anything.  
   Example of linear regression is stock market. Here the dependent variable is stock price and the independent variable is time.  
   Formula used is Y= mx + c.

1. **Logistic Regression** – It is a method you should predict a dependent variable, given a set of independent variables, such that a dependent variable is categorical.  
   Ex – Checking your Score from Nursery to class 12th to check if you can get admission in Harvard. Linear regression work on Continuous quantity but Logistic Regression works on Categorical Quantity (Holds binary values, Yes or No, True or False). It is called Regression though it is **Classification Algorithm** because it is use to predict a dependent variable which is categorical in nature.  
       
   Value Can be 0, 1 or in between.  
   Logistic regression is used for classification and the output variable The output variable will be categorical variable. The relation between the variables is denoted as a S curve or Sigmoid Curve.
2. **Decision Tree** – It is a **Classification Algorithm**. It is a supervised machine learning algorithm which looks like an inverted tree wherein each node represents a predictor variable the link between the nodes represents a decision and each leaf node represents an outcome (response variable).  
     
   ID3 Algorithm is used to build Decision Tree.  
   Best attribute is the variable that splits the data most effectively. It uses information gain and Entropy.   
   Entropy Measure the impurity or uncertainty present in the data.  
   Information gain indicates how much information a particular feature/variable gives us about the final outcome.  
   A variable with highest Information gain will be used to split the data.
3. **Random Forest** – It is a **Classification Algorithm**. It builds multiple decision tree called as forest and glues them together to get a more accurate and stable prediction. It is more accurate than decision tree. Random forest Avoids Overfitting (When our model learns, so when new data is given it won’t give the outcome accurately). Bagging means to reduce the variation in the prediction by combining result of various decision trees on different samples of the data set. Having a variety of Decision trees in Random Forest is the reason it is more affective.
4. **Naïve Bayes Classifier** – It is a **Classification Algorithm**. It is based on Bayes theorem that is used to solve classification problems by following a probabilistic approach. It is based on the idea that predictor variables in a machine learning model are independent of each other. It is a parametric algorithm because it assumes that all the independent variable are in no way related to each other.  
   Ex – Fly, Beak, Bird, Run, Tail, Dog – if you give this data to the machine and ask it what animal can fly and have beaks, it will give you bird as the output.  
   
5. K Nearest Neighbour - It is a **Classification Algorithm but can be used as Regression Algorithm**. K means number of nearest neighbour. It is a supervised learning algorithm that classifies a new data point into the target class depending on the features of its neighbouring data points. For Ex - if you train the machine with cat pics and dog pics. When you show it a cat image, it will recognise it with Cat’s feature in it. It is a lazy Algorithm because it memories the data instead of learning from it.  
   It uses Euclidean Distance to find the nearest distance between data points.  
   
6. Support Vector Machines - It is a **Classification Algorithm and a Regression Algorithm**. It is a supervised classification method that separates data using hyperplanes. SVM is used to classify data using a hyperplane such the distance between the support vector is maximum.



When you can’t draw a Hyperplane use a Non-Linear SVM. It transforms 2D Data set to a 3D Data Set.

**Unsupervised Algorithms** –

1. K-Means Clustering – K means number of clusters. The process by which objects are classified into predefined number of groups so that they are as much as dissimilar as possible from 1 group to another group but as much similar as possible within each group.

**Reinforcement Learning** –

**Agent** – The reinforcement learning algorithm that learns from trial and error.

**Environment** - the world through which the agent moves.

**Action** - all the possible steps that the agent can take.

**State** - current condition returned by the environment.

**Reward** - an instant return from the environment to appraise the last action. **Policy** - the approach that agent uses to determine the next action based on the current state

**Value** - the expected long-term return with discount as opposed to the short term reward.

**Action value** - value this is similar to value except it takes an extra parameter the current action

The Basic Aim of RL Agent is to maximise Reward.

The **Reward Maximisation Theory** states that RL agent must be trained in such a way that he takes the best action so that the reward is maximum.

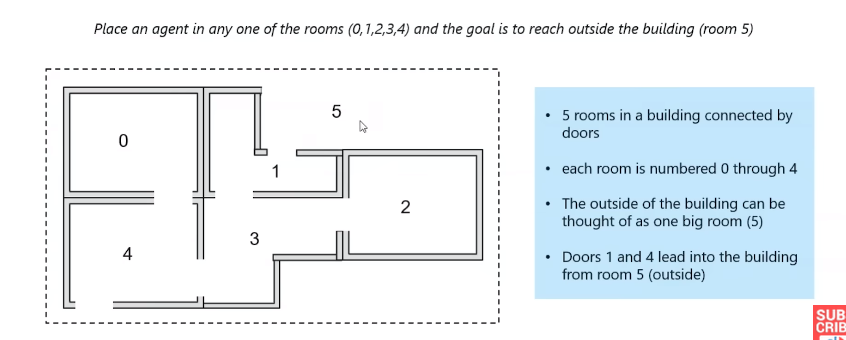
**Exploration** - is about exploring and capturing more information about An environment. A Fox is exploring the environment and finding the biggest flesh to eat.

**Exploitation** - is about using the already known exploited information to heighten the rewards. A Fox eating small fleshes near him instead of going for bigger flesh which is far away from him.

**Markov’s Decision Process** – The mathematical approach for mapping a solution in reinforcement learning is called Markov decision process.

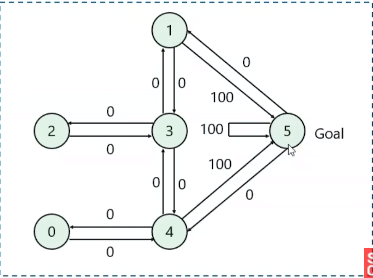
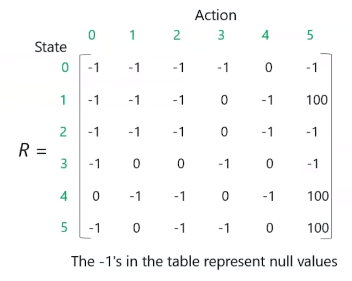
RL Algorithm –

1. Q-Learning – We have to find the best way so that the reward is maximum.

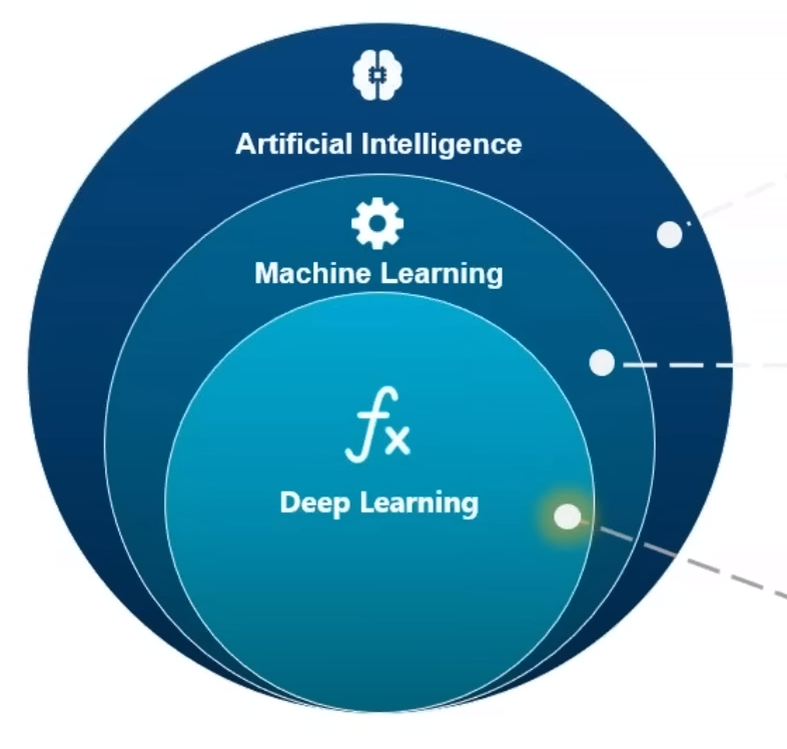


Representation in the form of graph –

Directly connected room will have 100 as reward.

-1 - 0 to 1 has no direct connection, 0 - 0 and 4 are connected but has no reward.



**Artificial intelligence** is a technique which enables machines to benefit human behaviour.

**Machine learning** is a subset of AI techniques which use statistical methods to enable machines to improve with experience.

**Deep Learning** is a subset of machine learning which makes the computation of multi-layer neural networks feasible.

**Limitations of ML –**

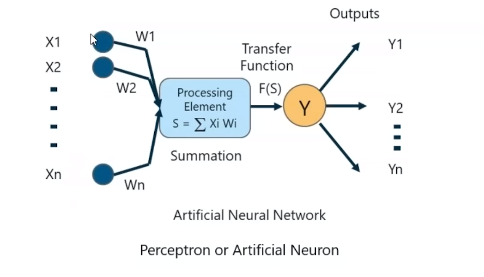
1. Not able to handle High Dimensional Data.
2. Restrictive in Image Recognition.
3. Feature Extraction – Challenge in object or handwriting recognition.

**Deep Learning** – The idea behind deep learning is to build learning algorithms that mimic brain. Deep learning models are capable to focus on the right features by themselves, requiring little guidance from the programmer. Models also partially solve the dimensionality problem.

Deep learning is a form of machine learning that uses a model of computing that’s very much inspired by the structure of the brain.

Deep learning is a collection pf statistical machine learning used to learn feature hierarchies based on the concept of artificial neural networks.

**Single Layer Perceptron** – An Artificial neuron or perceptron is a linear model used for Binary Classification. Hotels in New Rome which has a set of Inputs each of which is given a specific weight. The neuron computes some function on these weighted inputs and gives the output.



**A multi- layer perceptron** has the same structure of a single layer perceptron but with one or more hidden layers and thus considered a deep neural network. It uses the Feed Forward Network. **A feedforward**[**neural network**](https://deepai.org/machine-learning-glossary-and-terms/neural-network) is one of the simplest types of artificial neural networks devised. In this network, the information moves in only one direction—forward—from the input nodes, through the hidden nodes (if any), and to the output nodes. There are no cycles or loops in the network. Feedforward neural networks were the first type of artificial neural network invented and are simpler than their counterparts like [**recurrent neural networks**](https://deepai.org/machine-learning-glossary-and-terms/recurrent-neural-network) and[**convolutional neural networks**](https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network).

The most common deep learning algorithm for supervised training of the multi-layer perceptron is known as **back propagation**. Calculating the weighted sum of inputs and passing them through the activation function we propagate backwards and update the weight to reduce the error.

Limitations of Feed Forward Network - A trained feed forward network can be exposed to any random collection of photographs and a first photograph It is exposed to will not necessarily alter how it classifies the second one.

In feed forward network the output T+1 has nothing to do with the output T. It will have no relation with the previous output. Solution to this is –

**Recurrent networks** are a type of artificial neural network designed to recognise pattern sequence of data such as text genomes etc. The output of T depends on T-1 and all the previous output.

[**Convolutional Neural Networks**](https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network) – In this the neuron in a layer will only be connected to a small region of the layer before it instead of all the neurons in a fully connected manner.

**Natural Language Processing** – It is the overall goal to turn text into data for analysis. Auto complete is an application of NLP, Sentimental Analysis, Chatbot, Speech Recognition, Machine Translation.

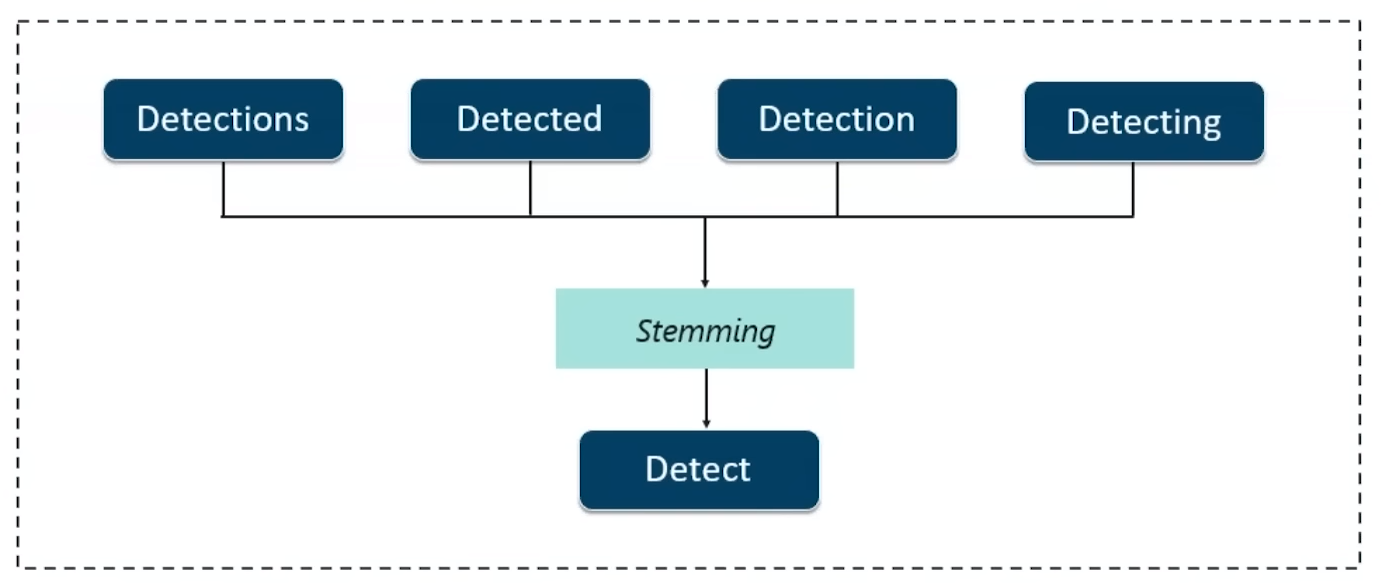
**Text Mining** – It is the process of deriving meaningful information from natural language text.

Text mining is a vast field that uses NLP for text analysis and text mining

**Terminologies in NLP** -

**Tokenization** - It is the process of splitting the whole data into smaller chunks.

**Stemming** – Normalize words into its base form or root form.



**Lemmitization** – It is upgrade version of Stemming. It will give proper word.

**Stop Word** – During web search removing not so important word and focusing on key words to give the best search result.

**Document Term Matrix** – It is used to find whether a document contains each of the given word during search.

Neural Networks: Neural networks are computational models inspired by the structure and functioning of the human brain. They consist of interconnected nodes, also known as neurons, organized into layers. The three main types of layers in a neural network are the input layer, hidden layer(s), and output layer. Each connection between neurons has a weight associated with it, and the network learns by adjusting these weights during training. Neural networks are used for various tasks, such as pattern recognition, classification, regression, and more.

Fuzzy Logic: Fuzzy logic is a mathematical framework that deals with uncertainty and imprecision. Unlike classical (or crisp) logic, which relies on binary values (true or false), fuzzy logic allows for degrees of truth between 0 and 1. This approach is particularly useful when dealing with vague or ambiguous information. Fuzzy logic is employed in control systems, decision-making processes, and artificial intelligence applications where precise, deterministic rules may be inadequate. It uses linguistic variables and fuzzy sets to represent and manipulate uncertain information.

Genetic Algorithm: Genetic algorithms are optimization algorithms inspired by the process of natural selection and genetics. They are used to find approximate solutions to optimization and search problems. The algorithm works by maintaining a population of potential solutions, called individuals, which are represented as strings of parameters (genetic representation). These individuals undergo evolution through processes like selection, crossover (recombination), and mutation, mimicking the biological evolution of species. Genetic algorithms are employed in various fields, including optimization, machine learning, and artificial intelligence, to find optimal or near-optimal solutions to complex problems.

**Big Data**

Big Data was coined by Roger Moauglas from O’reilly media in 2005

**Big Data Concept:**

**Definition:** Big Data refers to large and complex datasets that exceed the capabilities of traditional data processing methods. It encompasses the analysis, processing, and interpretation of vast amounts of data to reveal patterns, trends, and insights.

**Characteristics of Big Data:**

1. **Volume:**
   * Enormous amounts of data generated, stored, and processed.
2. **Velocity:**
   * Rapid speed at which data is generated, collected, and processed in real-time.
3. **Variety:**
   * Diverse types of data including structured (e.g., databases), semi-structured (e.g., XML, JSON), and unstructured (e.g., text, images).
4. **Veracity:** Refers to the accuracy and trustworthiness of the data.
5. **Value:**
   * Extracting meaningful insights and value from the data for informed decision-making.

### ****Batch Processing:****

**Definition:** Batch processing involves collecting, processing, and analyzing a large volume of data at a scheduled time. Data is typically stored and processed in chunks or batches. Batch processing is suitable for scenarios where data latency is not critical, and insights can be derived from a comprehensive analysis of historical data.

**Examples:**

* Traditional data warehousing, nightly ETL (Extract, Transform, Load) processes.

### ****Stream Processing:****

**Definition:** Stream processing involves the continuous and real-time analysis of data as it is generated. Data is processed as it flows through the system, enabling near-instantaneous insights and responses. Stream processing is crucial in scenarios where low-latency decision-making is essential.

### ****Micro-Batch Processing:****

**Definition:** Micro-batch processing is an approach that combines aspects of both batch and stream processing. Instead of processing data in large batches or in a pure real-time manner, micro-batch processing involves dividing the data into small, manageable chunks and processing them at short, regular intervals.

**Examples:**

* Apache Spark Streaming using micro-batch processing.

**Use Cases:**

* **Batch Processing:**
  + Historical trend analysis, large-scale data reporting.
* **Stream Processing:**
  + Fraud detection, real-time monitoring, live social media analytics.
* **Micro-Batch Processing:**
  + Near-real-time analytics, where low-latency is important but immediate processing is not mandatory.

**1. Structured Data:**

**Definition:**

* **Characteristics:**
  + Data that is organized in a tabular format with a clear and well-defined schema.
  + Typically fits into traditional relational databases.
  + Examples include data in rows and columns, like that found in SQL databases.
* **Use Cases:**
  + Financial records, customer information, transaction data.

**Advantages:**

* Easy to query and analyze using SQL.
* Well-suited for relational databases and traditional data processing methods.

**Disadvantages:**

* May not accommodate all types of data, especially those with complex relationships or varying structures.

**2. Unstructured Data:**

**Definition:**

* **Characteristics:**
  + Data that lacks a predefined data model or does not fit neatly into relational databases.
  + Varied formats and types, such as text, images, audio, and video files.
* **Use Cases:**
  + Social media posts, emails, multimedia content, sensor data.

**Advantages:**

* Captures a wide range of information sources.
* Allows for the storage and analysis of non-textual data.

**Disadvantages:**

* Challenging to analyze using traditional database techniques.
* Requires specialized tools for processing and analysis.

**3. Semi-Structured Data:**

**Definition:**

* **Characteristics:**
  + Data that has some structure but does not conform to the rigidity of a traditional relational database.
  + Typically represented in formats like JSON (JavaScript Object Notation) or XML (eXtensible Markup Language).
* **Use Cases:**
  + Configuration files, log files, data exchanged between systems.

**Advantages:**

* Retains some level of structure, making it more flexible than unstructured data.
* Allows for easier handling of hierarchical relationships.

**Disadvantages:**

* Still requires specialized tools for processing compared to fully structured data.
* Not as rigidly organized as structured data.

**Use Cases for Each Type:**

* **Structured Data:**
  + Ideal for scenarios where data follows a consistent, organized pattern, such as financial transactions or customer records.
* **Unstructured Data:**
  + Suited for handling diverse data sources like social media posts, images, or video content.
* **Semi-Structured Data:**
  + Offers a middle ground, providing some structure while accommodating variability. Commonly used in scenarios where data relationships are more complex than structured data allows.

RDBMS – It is a database system based on relational model introduced by Edgar F CODD.

In relational model data is represented in terms of tuples(rows).

RDBMS is used to manage relational database.

**Relational Database Management System (RDBMS):**

**Definition:**

* A Relational Database Management System (RDBMS) is a type of database management system that organizes data into tables (relations) with a predefined schema, establishing relationships between the tables.

**Key Concepts:**

1. **Tables:**
   * Basic storage structure, where data is organized in rows and columns.
   * Each table has a unique name and consists of attributes (columns) and tuples (rows).
2. **Schema:**
   * Defines the structure of the database, including tables, fields, and relationships.
   * Enforces data integrity and ensures consistency.
3. **Keys:**
   * **Primary Key:**
     + Uniquely identifies each record in a table.
   * **Foreign Key:**
     + Establishes a link between two tables, ensuring referential integrity.
4. **Normalization:**
   * Process of organizing data to reduce redundancy and dependency.
   * Achieved through the division of large tables into smaller ones linked by relationships.
5. **ACID Properties:**
   * **Atomicity:**
     + Transactions are treated as a single, indivisible unit of work.
   * **Consistency:**
     + Database remains in a consistent state before and after the transaction.
   * **Isolation:**
     + Transactions do not interfere with each other.
   * **Durability:**
     + Once a transaction is committed, its changes are permanent.
6. **SQL (Structured Query Language):**
   * Standard language for managing and manipulating relational databases.
   * Enables users to perform tasks such as querying data, updating records, and defining database structures.

**Components of RDBMS:**

1. **Database Engine:**
   * Core component responsible for data storage, retrieval, and management.
   * Executes SQL queries and ensures data integrity.
2. **Query Processor:**
   * Interprets and optimizes SQL queries for efficient execution.
   * Generates query execution plans.
3. **Storage Engine:**
   * Manages the storage of data on disk.
   * Responsible for indexing, caching, and retrieving data efficiently.
4. **Transaction Manager:**
   * Ensures the ACID properties of transactions.
   * Manages the execution of multiple transactions concurrently.

**Examples of RDBMS:**

* MySQL, PostgreSQL, Oracle Database, Microsoft SQL Server.

**Examples of DBMS:**

* Microsoft Access, FoxPro, dBASE, IMS (Information Management System).

**ACID in the Context of Big Data:**

ACID (Atomicity, Consistency, Isolation, Durability) is a set of properties that guarantee the reliability of transactions in a database system. These properties are crucial for maintaining the integrity and consistency of data, especially in scenarios where multiple transactions may be processed concurrently. While ACID principles are fundamental in traditional relational databases, their application in big data systems can vary.

**1. Atomicity:**

**Definition:**

* **Atomicity ensures that a transaction is treated as a single, indivisible unit of work.** Either all the changes made by the transaction are committed to the database, or none of them are.

**Big Data Implications:**

* In distributed big data systems, achieving true atomicity across distributed nodes can be challenging. Many distributed databases implement strategies to emulate atomicity, but complete atomic transactions may not always be guaranteed.

**2. Consistency:**

**Definition:**

* **Consistency ensures that a transaction brings the database from one valid state to another.** If a transaction violates the database's integrity constraints, it is rolled back, leaving the database in its original state.

**Big Data Implications:**

* Maintaining consistency is challenging in distributed systems, where nodes may temporarily operate independently. Some big data systems prioritize eventual consistency over immediate consistency to handle distributed scenarios.

**3. Isolation:**

**Definition:**

* **Isolation ensures that the execution of transactions is independent of each other, even when executed concurrently.** The result of a transaction should be as if it were executed in isolation.

**Big Data Implications:**

* Achieving full isolation in distributed systems can be complex. Different isolation levels may be implemented, and trade-offs may be made between performance and strict isolation.

**4. Durability:**

**Definition:**

* **Durability ensures that once a transaction is committed, its changes are permanent and survive subsequent failures.** The changes are stored in non-volatile memory.

**Big Data Implications:**

* Durability is crucial in big data systems, especially in distributed environments where nodes may fail independently. Replication and distributed storage systems play a role in ensuring the durability of data.

SQL was developed At IBM by Donard D Chamberlain and Raymond F Boyce in 1970s.

SQL (Structured Query Language) is a domain-specific language used for managing and manipulating relational databases. The theoretical foundation of SQL is rooted in the principles of relational database management systems (RDBMS). Here are key concepts related to the theory of SQL databases:

**Normalization:**

* The process of organizing data to reduce redundancy and dependency.
* Achieved through the division of large tables into smaller ones linked by relationships.

**SQL Statements:**

* **DDL (Data Definition Language):**
  + Used to define and manage database structures.
  + Includes statements like **CREATE TABLE**, **ALTER TABLE**, **DROP TABLE**.
* **DML (Data Manipulation Language):**
  + Used to manipulate and query data.
  + Includes statements like **SELECT**, **INSERT**, **UPDATE**, **DELETE**.
* **DCL (Data Control Language):**
  + Manages access to data.
  + Includes statements like **GRANT** and **REVOKE**.

**Normalization Forms:**

* **First Normal Form (1NF):**
  + Eliminates duplicate columns in a table.
* **Second Normal Form (2NF):**
  + 1NF plus all non-prime attributes are fully functionally dependent on the primary key.
* **Third Normal Form (3NF):**
  + 2NF plus no transitive dependencies.

NoSQL -It provides mechanism for storage and retrieval of data that is modelled in means other than tabular relations used in relational databases

NoSQL databases, often referred to as "Not Only SQL," represent a class of database systems that diverge from traditional relational databases. These databases are designed to handle vast amounts of unstructured, semi-structured, or structured data.

**Types of NoSQL Databases:**

**a. Document-Oriented Databases:**

* **Example:** MongoDB, CouchDB
* **Model:** Data is stored in flexible, semi-structured documents (e.g., JSON or BSON). Each document can have a different structure.

**b. Key-Value Stores:**

* **Example:** Redis, Amazon DynamoDB
* **Model:** Simple key-value pairs. Fast and efficient for basic read and write operations.

**c. Column-Family Stores:**

* **Example:** Apache Cassandra, HBase
* **Model:** Data is stored in columns rather than rows. Well-suited for read and write-intensive workloads.

**d. Graph Databases:**

* **Example:** Neo4j, Amazon Neptune
* **Model:** Represents data as a graph, consisting of nodes and edges. Excellent for scenarios involving complex relationships.

**e. Multi-Model Databases:**

* **Example:** ArangoDB, OrientDB
* **Model:** Supports multiple data models within a single database system.

In the context of NoSQL databases and distributed systems, BASE properties stand for Basically Available, Soft state, and Eventually consistent. This model is an alternative to the traditional ACID (Atomicity, Consistency, Isolation, Durability) properties, and it reflects the design principles suitable for distributed and scalable architectures. Here's a brief explanation of BASE:

1. **Basically Available:**
   * The system remains operational and responsive even in the face of failures.
   * Availability is prioritized over immediate consistency.
2. **Soft State:**
   * The system may exhibit a degree of inconsistency that is acceptable during certain periods.
   * Inconsistency is temporary and will be resolved over time.
3. **Eventually Consistent:**
   * The system will become consistent over time as updates propagate through the system.
   * Consistency is not guaranteed at every moment but is achieved eventually.

### ETL (Extract, Transform, Load):

#### **Overview:**

* **Definition:** ETL is a data integration process that involves extracting data from source systems, transforming it to meet the target format, and loading it into a destination system, typically a data warehouse.

#### **Process:**

1. **Extract:**
   * Retrieve data from multiple sources, which can be databases, applications, flat files, or external APIs.
2. **Transform:**
   * Apply various transformations like cleaning, filtering, aggregating, and converting data to a format suitable for analysis.
3. **Load:**
   * Load the transformed data into a target data warehouse or database for storage and analysis.

#### **Characteristics:**

* **Batch Processing:**
  + ETL processes typically run as batch jobs at scheduled intervals.
* **Centralized Processing:**
  + Transformation is often performed in a dedicated ETL server before loading into the data warehouse.

### ELT (Extract, Load, Transform):

#### **Overview:**

* **Definition:** ELT is a data integration process that involves extracting data from source systems, loading it into a data warehouse as is, and then transforming it within the data warehouse.

#### **Process:**

1. **Extract:**
   * Retrieve data from multiple sources similar to ETL.
2. **Load:**
   * Directly load the raw data into a data warehouse without significant transformations.
3. **Transform:**
   * Apply transformations within the data warehouse using SQL or other processing capabilities.

#### **Characteristics:**

* **Near Real-Time Processing:**
  + ELT processes can support near real-time data integration.
* **Distributed Processing:**
  + Transformation is distributed across the data warehouse nodes, leveraging their processing power.

| **Aspect** | **ETL** | **ELT** |
| --- | --- | --- |
| **Order of Processing** | Extract, Transform, Load (Sequential) | Extract, Load, Transform (Parallel) |
| **Data Movement** | Data is extracted, transformed, and then loaded | Data is extracted and loaded before transformation |
| **Processing Location** | Centralized ETL server or cluster | Distributed processing within the data warehouse |
| **Use Cases** | Well-suited for batch processing and historical data | Suited for near real-time processing and big data scenarios |
| **Flexibility** | Allows for significant data transformations during the ETL process | Primarily focuses on loading raw data into the data warehouse before transformations |

* **Database:**
  + General-purpose system for storing and managing data for transactional applications.
  + Typically deals with smaller data volumes compared to data warehouses.
* **Data Warehouse:**
  + Specialized for analytical processing, reporting, and decision support.
  + Handles large volumes of data, often in the order of terabytes or more.

**Data Warehouse Architecture Components:**

1. **Operational Data Sources:**
   * The starting point for data warehouse architecture, where data is sourced from operational databases, external systems, flat files, and other relevant sources.
2. **ETL (Extract, Transform, Load) Processes:**
   * Responsible for extracting data from source systems, transforming it to meet data warehouse requirements, and loading it into the data warehouse.
3. **Staging Area:**
   * An intermediate storage area where raw data from source systems is temporarily stored before undergoing transformation.
4. **Data Warehouse Database:**
   * The core storage component where data is organized, indexed, and optimized for analytical querying. It typically includes fact tables and dimension tables.
5. **Data Marts:**
   * Subsets of a data warehouse focused on specific business areas or departments. Data marts are designed for easier and more specialized access.
6. **Metadata Repository:**
   * Stores information about the structure, definitions, and lineage of data within the data warehouse. It facilitates data governance and management.
7. **OLAP (Online Analytical Processing) Engine:**
   * Enables multidimensional analysis and reporting. OLAP tools provide a user-friendly interface for querying and exploring data.
8. **Query and Reporting Tools:**
   * Interfaces that allow end-users to interact with the data warehouse, run queries, and generate reports for decision-making.
9. **Data Mining and Analysis Tools:**
   * Tools that leverage machine learning and statistical techniques to discover patterns, trends, and insights within the data.

**Data Warehouse Architecture Layers:**

1. **Source Layer:**
   * Ingests data from operational sources, including databases, applications, and external systems.
2. **Integration Layer:**
   * Involves the ETL processes that extract, transform, and load data into the data warehouse.
3. **Warehouse Database Layer:**
   * Houses the actual data warehouse, organized into tables, views, and other structures optimized for analytical processing.
4. **Access Layer:**
   * Provides interfaces for end-users to query, report, and analyze data. Includes OLAP tools, query tools, and reporting applications.

### OLAP (Online Analytical Processing):

#### **Theory:**

* **Definition:** OLAP is a category of software tools that allows users to interactively analyze multidimensional data from multiple perspectives. It's designed for complex queries and trend analysis rather than transactional processing.
* **Data Model:** Utilizes a multidimensional data model, often represented as data cubes. Data cubes have dimensions (e.g., time, geography) and measures (e.g., sales, revenue).
* **Query Complexity:** Supports complex queries involving aggregations, calculations, and comparisons. Focuses on analytical tasks to derive business insights.
* **Latency:** Tolerant to higher latency as it deals with historical and aggregated data. Real-time updates are not the primary concern.
* **Examples:** Tools like Microsoft Excel PivotTables, SAP BW, and Oracle OLAP.

### OLTP (Online Transaction Processing):

#### **Theory:**

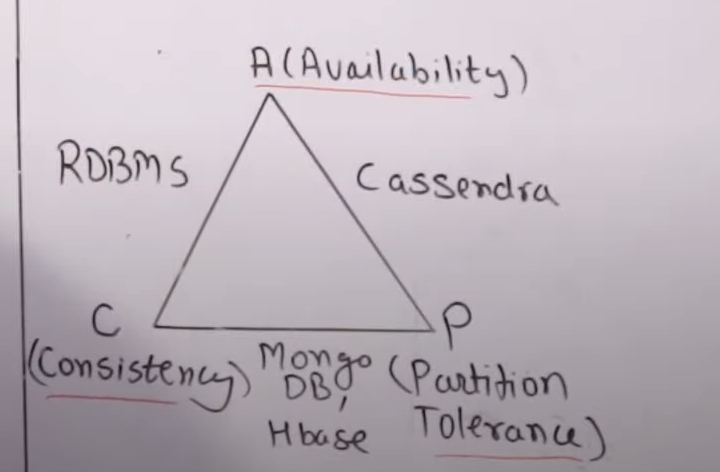
* **Definition:** OLTP is a category of software applications that facilitate and manage transaction-oriented processes. It involves the collection, modification, and retrieval of transactional data in real-time.
* **Data Model:** Utilizes a relational data model with normalized tables to minimize redundancy and ensure data integrity. Emphasizes quick and accurate transactional processing.
* **Query Complexity:** Involves simple, routine queries and updates. Primarily focuses on the efficient execution of individual transactions.
* **Latency:** Requires low latency for real-time processing. The system must respond quickly to individual transactions.
* **Examples:** Applications like e-commerce systems, banking systems, and airline reservation systems.

### Differences between OLAP and OLTP:

| **Aspect** | **OLAP** | **OLTP** |
| --- | --- | --- |
| **Purpose** | Analytical processing | Transactional processing |
| **Data Model** | Multidimensional (data cubes) | Relational (normalized tables) |
| **Query Complexity** | Complex queries for analysis | Simple queries for transactions |
| **Latency** | Tolerant to higher latency | Requires low latency for real-time processing |
| **Usage** | Decision support, business intelligence | Daily operations, routine transactions |
| **Examples** | Microsoft Excel PivotTables, SAP BW | E-commerce systems, banking systems |

The CAP theorem, also known as Brewer's theorem, is a fundamental principle in distributed systems that describes the trade-offs among three key properties: Consistency, Availability, and Partition Tolerance. The theorem was proposed by computer scientist Eric Brewer in 2000.

1. **Consistency:**
   * In a distributed system, consistency ensures that all nodes in the system see the same data at the same time. Every read receives the most recent write, and there is a global order to operations.
2. **Availability:**
   * Availability guarantees that every request to the system receives a response, without the guarantee that it contains the most recent version of the data. In other words, a system is considered available as long as it responds to queries, even if the response is not the latest data.
3. **Partition Tolerance:**
   * Partition tolerance refers to the system's ability to continue functioning and providing responses even when network partitions occur. A partition occurs when communication between nodes is lost or delayed.



SCALA -

Skylanders developed by Martin Odersky in 2001 and was released in 2000 and 2004.

It’s a mix paradigm language as it supports Oops and Functional Programming.

Runs on JVM so It can use Java code with example.

Better then Java because it is crisp and small.

It has type inference as it can automatically detect the variable.

(var a =1, var b=’c’).

Apache spark was made with SCALA.

### 3. ****Scala:****

#### **Characteristics:**

* **Functional and Object-Oriented:**
  + Scala combines functional and object-oriented programming paradigms, providing expressive and concise code.
* **Concurrency Support:**
  + Ability to execute 2 or more programs simulatenously.
* **Spark Native Language:**
  + Scala is the native language for Apache Spark, and Spark's core components are implemented in Scala.

## Data Cleansing:

### Definition:

Data cleansing, also known as data cleaning or data scrubbing, is the process of identifying and correcting errors, inconsistencies, and inaccuracies in datasets to improve data quality. The goal is to enhance the reliability and accuracy of the data, making it suitable for analysis and decision-making.

### Key Steps in Data Cleansing:

1. **Handling Missing Values:**
   * Identify and address missing values using techniques such as imputation (replacing missing values with estimated ones) or removal.
2. **Standardization:**
   * Ensure consistency by standardizing formats (e.g., dates, addresses, units) across the dataset.
3. **Removing Duplicates:**
   * Identify and eliminate duplicate records to avoid redundancy and maintain data integrity.
4. **Correcting Inaccuracies:**
   * Identify and rectify inaccuracies, such as typos, inconsistent spellings, and incorrect data entries.
5. **Outlier Detection and Treatment:**
   * Identify outliers that deviate significantly from the norm and decide whether to correct or remove them.
6. **Handling Inconsistent Data:**
   * Address inconsistencies in data representation, units, or scales to ensure uniformity.
7. **Addressing Data Integrity Issues:**
   * Identify and correct issues that affect the overall integrity of the dataset.
8. **Normalization:**
   * Normalize numerical data to a standard scale to prevent biases in analysis caused by varying units or scales.

### Importance of Data Cleansing:

* **Accurate Analysis:** Cleansed data ensures that analyses and decisions are based on accurate and reliable information.
* **Reduced Errors:** Minimizes the chances of errors in reporting, forecasting, and other data-driven activities.
* **Improved Decision-Making:** Enhances the quality of data used in decision-making processes, leading to better-informed decisions.
* **Increased Trust:** Users are more likely to trust and rely on clean data, fostering confidence in the organization's data assets.

## Data Modeling:

### Definition:

Data modeling involves designing the structure of a database or data warehouse to represent and organize data in a meaningful way. It includes defining tables, relationships, constraints, and data types to facilitate efficient data storage, retrieval, and analysis.

### Key Steps in Data Modeling:

1. **Identifying Entities:**
   * Identify the main entities (objects or concepts) in the data domain that need representation.
2. **Defining Attributes:**
   * Determine the characteristics or properties (attributes) of each entity.
3. **Establishing Relationships:**
   * Identify and define relationships between entities, specifying their nature (one-to-one, one-to-many, many-to-many).
4. **Normalizing Data:**
   * Normalize data by reducing redundancy and dependency through techniques like normal forms.
5. **Creating Tables:**
   * Based on entities and attributes, create tables with appropriate fields (columns) and data types.
6. **Defining Primary and Foreign Keys:**
   * Specify primary keys (unique identifiers) for each table and establish foreign keys to represent relationships between tables.
7. **Enforcing Constraints:**
   * Define and enforce constraints (e.g., unique constraints, check constraints) to maintain data integrity.
8. **Designing Indexes:**
   * Create indexes on columns to optimize query performance.

**The data engineering lifecycle** involves the end-to-end process of collecting, storing, processing, and analyzing data to derive insights and support decision-making. It encompasses various stages from data acquisition to deployment of data solutions. Here's an overview of the data engineering lifecycle:

**1. Requirements Gathering:**

* **Objective:** Understand business requirements and data needs.
* **Activities:**
  + Collaborate with stakeholders to identify data sources and desired outcomes.
  + Define data quality and security requirements.
  + Document business rules and transformation needs.

**2. Data Ingestion:**

* **Objective:** Collect and import data from various sources into the data storage system.
* **Activities:**
  + Select appropriate data sources (databases, APIs, logs).
  + Implement data extraction processes.
  + Validate and clean incoming data.

**3. Data Storage:**

* **Objective:** Store data in a structured and scalable format.
* **Activities:**
  + Choose an appropriate storage solution (relational databases, data lakes, NoSQL databases).
  + Define data schemas and structures.
  + Optimize storage for performance and cost.

**4. Data Processing:**

* **Objective:** Transform and prepare data for analysis.
* **Activities:**
  + Implement ETL (Extract, Transform, Load) processes.
  + Perform data cleansing, normalization, and enrichment.
  + Handle data partitioning and indexing.

**5. Data Quality Assurance:**

* **Objective:** Ensure the accuracy, completeness, and reliability of the data.
* **Activities:**
  + Implement data validation checks.
  + Monitor data quality metrics.
  + Set up data profiling and anomaly detection.

**6. Data Integration:**

* **Objective:** Combine and integrate data from different sources.
* **Activities:**
  + Resolve schema mismatches.
  + Establish data relationships.
  + Enable cross-functional analysis.

**7. Data Governance and Security:**

* **Objective:** Ensure compliance with data governance policies and secure data access.
* **Activities:**
  + Define data access controls and permissions.
  + Implement encryption and authentication measures.
  + Enforce data retention policies.

**8. Metadata Management:**

* **Objective:** Maintain a catalog of metadata to document and manage data assets.
* **Activities:**
  + Create and update metadata repositories.
  + Document data lineage, definitions, and usage.
  + Facilitate discovery and understanding of data assets.

**9. Data Exploration and Analysis:**

* **Objective:** Enable data analysts and scientists to explore and derive insights from the data.
* **Activities:**
  + Provide tools for querying and analyzing data.
  + Support ad-hoc queries and exploratory data analysis.
  + Implement data visualization tools.

**10. Modeling and Feature Engineering:**

* **Objective:** Create models and features for machine learning and advanced analytics.
* **Activities:**
  + Develop predictive models.
  + Engineer features for model training.
  + Evaluate model performance.

**11. Deployment and Automation:**

* **Objective:** Deploy data solutions and automate processes for ongoing efficiency.
* **Activities:**
  + Deploy data pipelines and models into production.
  + Set up scheduling and automation for regular updates.
  + Implement monitoring for pipeline health and performance.

**12. Maintenance and Optimization:**

* **Objective:** Ensure ongoing performance, scalability, and reliability.
* **Activities:**
  + Monitor and optimize storage and processing efficiency.
  + Address issues related to data drift.
  + Update data pipelines and models as needed.

**13. Documentation and Knowledge Transfer:**

* **Objective:** Document processes, architectures, and best practices for knowledge transfer.
* **Activities:**
  + Document data engineering processes.
  + Provide training and documentation for new team members.
  + Share insights and lessons learned.

**14. Continuous Improvement:**

* **Objective:** Continuously enhance data engineering processes and systems.
* **Activities:**
  + Collect feedback from users and stakeholders.
  + Implement improvements based on lessons learned.
  + Stay informed about emerging technologies and best practices.

**Apache Spark: Overview of a Big Data Framework**

Apache Spark is an open-source, distributed computing system that provides a fast and general-purpose cluster computing framework for big data processing. It was developed to address limitations and improve upon the MapReduce model. Spark is designed to be fast, easy to use, and versatile, supporting various workloads such as batch processing, iterative algorithms, interactive queries, and streaming.

**Key Features:**

1. **In-Memory Processing:**
   * Spark stores intermediate data in memory, reducing the need for disk I/O and speeding up iterative algorithms.
2. **Distributed Data Processing:**
   * Distributes data across a cluster of machines, allowing for parallel processing and improved scalability.
3. **Ease of Use:**
   * Offers high-level APIs in Java, Scala, Python, and R, making it accessible to a wide range of developers.
4. **Unified Platform:**
   * Integrates with various data processing components, including SQL, streaming, machine learning, and graph processing, providing a unified platform for diverse workloads.
5. **Fault Tolerance:**
   * Provides fault tolerance through lineage information and recomputation of lost data partitions.
6. **Lazy Evaluation:**
   * Optimizes the execution plan by evaluating only the necessary transformations and actions when needed.
7. **Advanced Analytics:**
   * Supports machine learning (MLlib), graph processing (GraphX), and stream processing (Structured Streaming) libraries.
8. **Compatibility:**
   * Integrates with Hadoop Distributed File System (HDFS) and other data sources like HBase, Cassandra, and Amazon S3.

**Components of Apache Spark:**

1. **Spark Core:**
   * The foundational library that provides basic functionality for distributed computing, including task scheduling, fault recovery, and communication.
2. **Spark SQL:**
   * A module for structured data processing, allowing users to query structured data using SQL-like syntax. It supports DataFrame and SQL API.
3. **Spark Streaming:**
   * A micro-batch processing engine for handling real-time data streams, enabling scalable and fault-tolerant stream processing.
4. **MLlib (Machine Learning Library):**
   * A scalable machine learning library with algorithms for classification, regression, clustering, and collaborative filtering.
5. **GraphX:**
   * A graph processing library for graph-parallel computation, enabling the execution of iterative graph algorithms.
6. **Structured Streaming:**
   * A scalable and fault-tolerant stream processing engine based on Spark SQL, allowing developers to express streaming computations using SQL queries.

Hadoop – apache based open source framework implemented in java, Distributed storage managed by HDFS, Big data processed in distributed processing by MapReduce. Parallel Processing happens, local computation and storage is there.

Common Utilities/hadoop common – Java lib, utilities or files required by other modules for Hadoop to work.

YARN(yet another resource negotiater) framework – Job scheduling, resource management.

HDFS – Hadoop distributed file system – Distributed storage based on google file system. As it is distributed if one machine fail we can bring the data from other source.

Name node stores meta data, manage data nodes and Data node stores actual data.

MapReduce – Distributed processing, parallel processing.

Input data =>Map breaks one form of set of data into other set of data tupples (Key value pair),=>Reduce – Combine Key value pairs on the basis of keys=>output.

Master Job Tracker – One – kaun se task ko kitne zaroorat hai utna hi dena hai aur kb tk dena hai Resources, Scheduling task, Monitoring task.

Slave Task Tracker – Many – One slave is done then baaki slaves ko de deta hai. Execute krta hai task.

Apache Hive – is an open source data warehouse system for querying and analysing large set of stored Hadoop files.

Hadoop stored it in hard drives and it will use batch processing,Spark uses Resilient distributed data set, RAM m store hota hai Data. It is faster and more efficient than Hadoop.