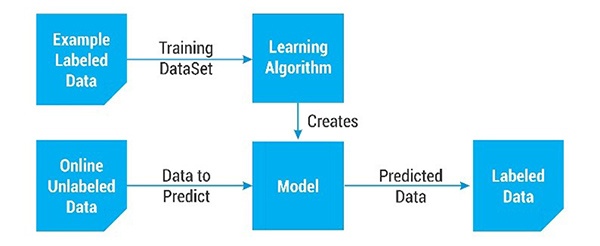
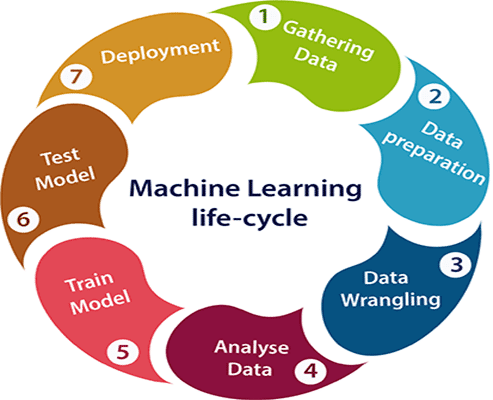
**MACHINE LEARNING**

Machine Learning is subset of Artificial Intelligence that enables machines to learn patterns from data and make predictions or decisions without being explicitly programmed for each specific task.

In simpler terms, ML systems improve their performance automatically through experience, just like humans learn from past experience.



**Machine Learning Life Cycle**



1. **Problem Definition**

Clearly defining the objective of the ML project.

Key Sub Steps: -

* Identifying business goals or research question
* Define the Type of ML problem like Classification, Regression, Clustering, etc.
* Understand the target variables

1. **Data Gathering**

Gathering relevant and sufficient data to solve the problem

Key Sub Steps: -

* Identifying Data Sources (Database, APIs, web scraping)
* Collecting Relevant Type of Data Structured (CSV, SQL) or Unstructured Data (Images, text)

1. **Data Preparation**

Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.

This step can be further divided into two processes

1. Data Exploration:

* It is used to understand the nature of data that we have to work with.
* We need to understand the characteristics, format, and quality of data.

Key Sub-Steps:

* **Data Summary**:
  + Shape of data (rows, columns)
  + Data types (numerical, categorical, text)
  + Count of missing values per column
* **Descriptive Statistics**:
  + Mean, Median, Standard Deviation
  + Range, Percentiles
* **Univariate Analysis**:
  + Histogram, Boxplot, KDE Plot for individual features
* **Bivariate/Multivariate Analysis**:
  + Scatter plots, Pair plots
  + Correlation heatmap (for feature relationships)
  + Grouped bar charts (categorical vs target)
* **Target Distribution**:
  + Check for class imbalance (in classification)
  + Understand output variable variance (in regression)
* **Outlier Detection**:
  + Boxplots, Z-score method, IQR method

1. Data Preprocessing

* Cleaning and transforming raw data into a suitable form for model training.
* Make the dataset machine-readable and consistent

Key Sub-Steps:

* **Handling Missing Values**:
  + Numerical: Mean, Median, Interpolation
  + Categorical: Mode, ‘Unknown’, Forward Fill
  + Dropping rows/columns (if justified)
* **Feature Encoding**:

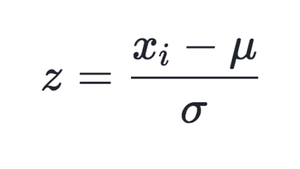
Machine learning models can only work with numerical values. For this reason, it is necessary to transform the categorical values of the relevant features into numerical ones. This process is called *feature encoding*.

* + **Label Encoding**: Ordinal categorical features
  + **One-Hot Encoding**: Nominal categorical features
  + **Feature Scaling**:

Feature Scaling is a technique to standardize the independent features present in the data. It is performed during the data pre-processing to handle highly varying values.

* + - **Standardization (Z-score)**:

Standardization (also called **Z-score normalization**) is the process of rescaling features so that they have the properties of a standard normal distribution — i.e., a mean = 0 and standard deviation = 1.



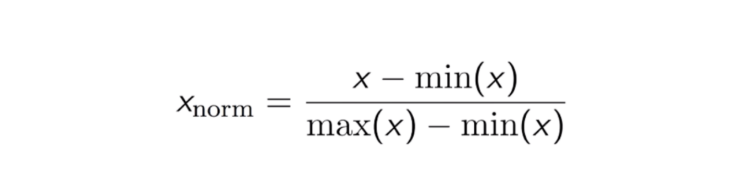
Where:

* x = individual data point
* μ = mean of the feature
* σ = standard deviation of the feature
* z = standardized value

**When to Use**

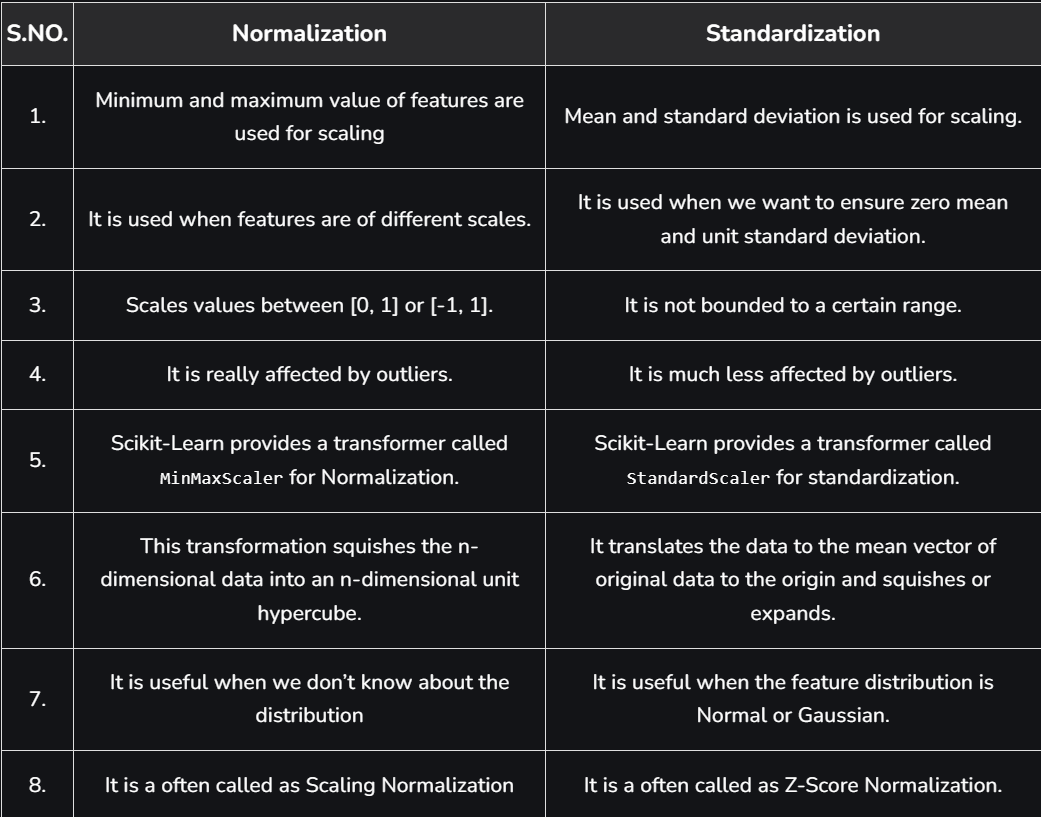
* When your data follows a **Gaussian (normal) distribution**
* Required by algorithms that assume standard distribution of data:
  + Linear Regression
  + Logistic Regression
  + SVM
  + K-Means
  + PCA
* **Normalization (Min-Max Scaling):**

Normalization (also known as **Min-Max Scaling**) is the process of rescaling features to a fixed range, usually [0, 1].



**When to Use**

* When features do not follow a normal distribution
* When you want all features on the same scale
* Required by algorithms sensitive to magnitude or distance:
  + K-Nearest Neighbours (KNN)
  + Neural Networks
  + Gradient Descent-based models



1. **Data Wrangling**

* Data Wrangling is a process of cleaning and converting raw data into usable format.
* It is the process of transforming raw, messy and unstructured data into a clean, structured, and usable format for analysis or modelling.

**Key Data Wrangling Tasks:**

**1. Handling Missing Data**

* Imputation (mean/median/mode)
* Forward/Backward Fill
* Dropping rows/columns

**2. Removing Duplicates**

* Drop exact or near-duplicate rows
* Use. drop\_duplicates() in pandas (Python)

**3. Correcting Data Types**

* Convert object/string to datetime
* Change numerical columns stored as strings to float/int

**4. Fixing Structural Errors**

* Typos in category names (e.g., “Male”, “male”, “MALE”)
* Inconsistent naming conventions
* Mixed data formats in the same column

**5. Filtering and Sorting**

* Removing irrelevant data points (e.g., out-of-range entries)
* Sorting based on timestamps, IDs, values

**6. Dealing with Outliers**

* IQR, Z-score methods
* Capping, trimming, or transforming data

**7. Flattening Nested Data**

* Convert JSON structures or nested tables into a tabular format
* Exploding lists into rows

**8. Text Normalization (if applicable)**

* Lowercasing, removing punctuation, special characters
* Removing HTML tags, Unicode cleanup

1. **Data Analysis**

Data Analysis is the process of inspecting, cleansing, transforming, and modelling data to extract useful insights, patterns, and knowledge that support decision-making or model development.

In Machine Learning, **Data Analysis is the bridge** between raw data and modelling — it helps understand the data’s behaviour, detect trends, and decide which preprocessing or modelling techniques to apply.

Types Of Data Analysis:

* 1. Descriptive Analysis: Focuses on summarizing historical data to understand what has happened in the past.  For example, analysing sales data to understand the average monthly sales for the past year.

**Techniques:**

* Summary statistics (mean, median, std)
* Data visualization (bar charts, line graphs)
* Grouping and aggregation
  1. Diagnostic Analysis: Examines why certain outcome occurred by exploring relationships among data variables. For instance, finding out why a company's sales dropped in a particular month.

**Techniques:**

* Correlation analysis
* Hypothesis testing
* Drill-down and filtering
* Anomaly detection
  1. Predictive Analysis:  Uses statistical models and forecasting techniques to predict future outcomes based on past data. For example, forecasting the next quarter's sales

**Techniques:**

* Machine Learning models (Regression, Time Series, Classification)
* Decision trees, Random Forests
* Logistic Regression, ARIMA models
  1. Prescriptive Analysis: Suggests actions to benefit from predictions, using tools like machine learning and artificial intelligence. For example, recommending the best marketing strategies to increase future sales

**Techniques:**

* Optimization algorithms
* Reinforcement Learning
* Scenario modelling
* AI-driven decision support systems

1. **Feature Engineering**

Feature engineering is the process of transforming raw data into meaningful features that enhance the predictive performance of machine learning models. It involves selecting, creating, and modifying data attributes to better represent the underlying patterns in the dataset, making it easier for algorithms to learn and generalize effectively

*Key Components of Feature Engineering*

1. **Feature Selection**: Identifying and retaining relevant features while removing redundant or irrelevant ones using statistical methods or domain knowledge.
2. **Feature Transformation**: Modifying raw features into formats suitable for modelling, such as normalizing continuous variables or encoding categorical variables.
3. **Feature Creation**: Generating new features by combining existing ones to capture additional insights (e.g., creating "price per square foot" from "price" and "area").
4. **Handling Missing Data**: Addressing missing values through imputation, deletion, or native handling by algorithms.
5. **Time-Based Features**: Extracting temporal attributes like day of the week or rolling averages for time series data.
6. **Model Selection**

Model selection refers to the process of choosing the most appropriate machine learning model from a set of candidate models to solve a specific problem effectively. This involves evaluating models based on their performance metrics, computational efficiency, interpretability, and suitability for the given dataset and task.

*Common methods used in model selection include:*

* **Train-Test Split Evaluation**: This technique splits the dataset into two parts: a **training set** to train the model and a **test set** to evaluate its performance
* **K-Fold Cross-Validation**: In K-Fold Cross-Validation, the dataset is divided into K equal parts (folds). The model is trained on K-1 folds and tested on the remaining fold, repeating the process K times so every fold serves as a test set once.

Final performance is the average of all K evaluations.

* **Stratified K-Fold Cross-Validation:** A variation of K-Fold where the folds are created in such a way that the distribution of classes remains consistent across all folds. Best suited for classification problems with imbalanced datasets.
* **K-Fold Cross-Validation**: In K-Fold Cross-Validation, the dataset is divided into K equal parts (folds). The model is trained on K-1 folds and tested on the remaining fold, repeating the process K times so every fold serves as a test set once.

Final performance is the average of all K evaluations.

*When selecting a model, several factors should be considered:*

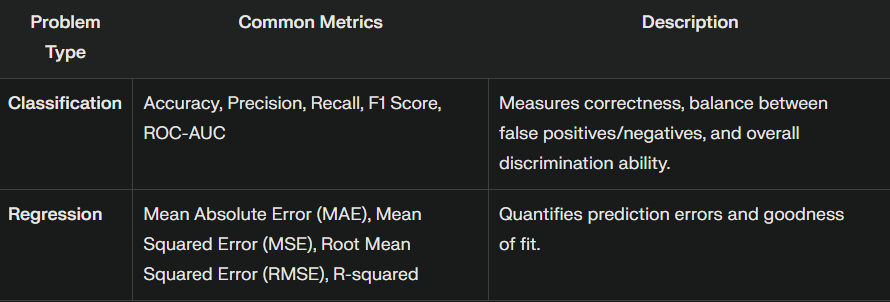
1. **Problem Type**: Determine whether the task is classification, regression, clustering, or another type. This helps narrow down suitable algorithms.
2. **Performance Metrics**: Evaluate models using metrics such as accuracy, precision, recall, or mean squared error to identify the best-performing model.
3. **Data Characteristics**: Consider the size, structure, and quality of the data, including handling missing values and outliers.
4. **Computational Efficiency**: Choose models that align with available resources and time constraints.
5. **Interpretability**: In fields like healthcare or finance, simpler models may be preferred for their ability to explain predictions clearly.
6. **Scalability**: Ensure the model can handle large datasets or adapt to changes in data distributions over time

**8. Model Training**

Model training is the process of teaching a machine learning algorithm to recognize patterns and make predictions based on data. It involves optimizing the model's parameters (such as weights and biases) using a training dataset to minimize the difference between its predictions and actual outcomes. This step is critical for enabling the model to generalize well to unseen data

**9. Model Evaluation**

Model evaluation is the process of assessing how well a trained machine learning model performs on new, unseen data. It measures the model’s ability to generalize beyond the training dataset and helps identify issues like overfitting or underfitting. Effective evaluation ensures the model’s predictions are reliable and suitable for deployment.

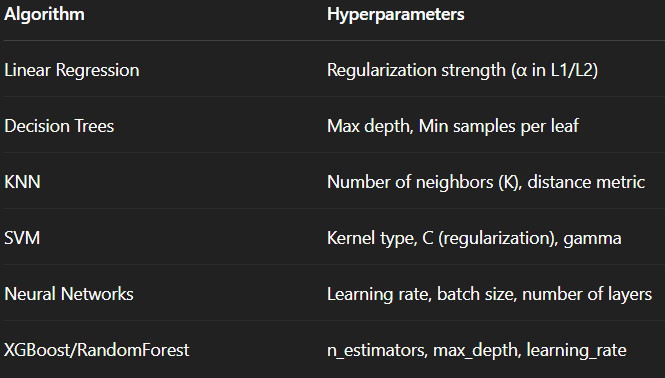


**10. Hyperparameter Tuning**

Hyperparameter tuning is the process of optimizing the hyperparameters of a machine learning model to achieve the best possible performance.

Unlike model parameters (which are learned during training), hyperparameters are set before training and control the learning process.

Examples:



**11. Model Deployment:**

Model deployment is the process of integrating a trained machine learning (ML) model into a production environment where it can process real-world data and deliver actionable outputs. This is a critical phase in the ML lifecycle, as it transforms a theoretical model into a practical tool that supports decision-making, automation, or predictive analytics.

**Probably Approximately Correct Learning**

* A framework for mathematical analysis of machine learning algorithm.
* A good learner will learn with high probability and close approximation to the target concept.
* With high probability, the selected hypothesis will have low error.
* PAC learning requires two parameters ϵ and δ
* **With probability at least (1- δ), a system learns the concept with error at most ϵ.**
* **ϵ is upper bound on the error in accuracy**, i.e. the hypothesis with error less than ϵ.

**Accuracy**: 1- ϵ

* **δ give the probability of failure** in achieving this accuracy δ, (0< δ<=1), the hypothesis generated is approximately correct at least 1- δ of the time

**Confidence**: 1- δ

* **Error Region**: P (c XOR h) <= ϵ
* 0 <= ϵ <=1/2
* **Probably Approximately Correct**

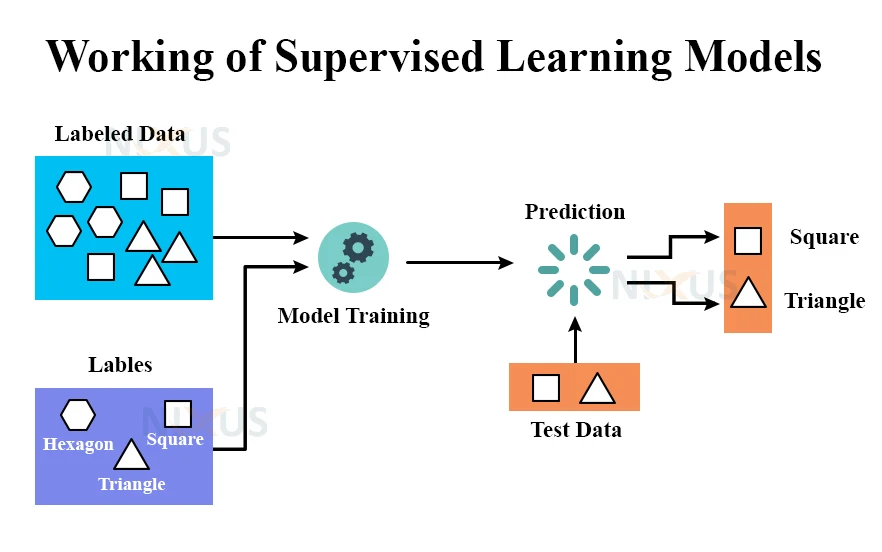
**[P (Error(h) <= ϵ)] <= 1 - δ**

**P (P (c XOR h) <= ϵ) <= 1 – δ**

* **Specialization:** hs is the tightest possible rectangle around a set of positive training examples.

**Supervised Learning: -**

* Supervised learning is a machine learning technique where models are trained using labelled data. Each input in the dataset is paired with its corresponding output, allowing the model to learn the relationships between inputs and outputs to make predictions on new, unseen data.
* The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y).



**How Supervised Machine Learning Works?**

Where **supervised learning algorithm** consists of input features and corresponding output labels. The process works through:

* **Training Data:**The model is provided with a training dataset that includes input data (features) and corresponding output data (labels or target variables).
* **Learning Process:**The algorithm processes the training data, learning the relationships between the input features and the output labels. This is achieved by adjusting the model’s parameters to minimize the difference between its predictions and the actual labels.
* **Testing phase** involves feeding the algorithm new, unseen data and evaluating its ability to predict the correct output based on the learned patterns.

Types:

* Classification
* Regression

Application:

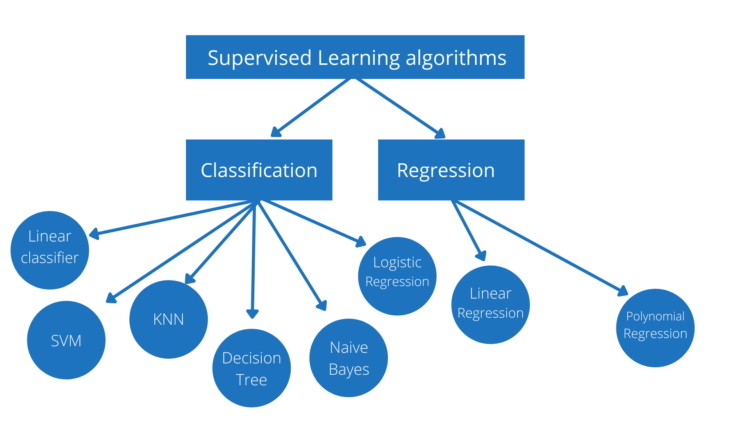
* Fraud Detection
* Spam Filtering
* Image classification
* Speech Recognition

Advantages:

* High accuracy in predictions with labelled data.
* Wide range of applications across domains.
* Models are often interpretable, aiding transparency.
* Clear goals due to labelled data.

Disadvantages:

* Requires large amount of labelled data, which can be expensive and time consuming
* Prone to overfitting if the model is too specific.
* May struggle with generalization to unseen scenarios.



* The prediction task is a ***classification*** when the target variable is discrete. An application is the identification of the underlying sentiment of a piece of text.
* The prediction task is a ***regression*** when the target variable is continuous. An example can be the prediction of the salary of a person given their education degree, previous work experience, geographical location, and level of seniority.

**Classification** in Machine Learning

* Classification is a supervised machine learning method where the model tries to predict the correct label of a given input data. In classification, the model is fully trained using the training data, and then it is evaluated on test data before being used to perform prediction on new unseen data.

**Lazy Learners Vs. Eager Learners**

There are two types of learners in machine learning classification: lazy and eager learners.

**Eager learners**are machine learning algorithms that first build a model from the training dataset before making any prediction on future datasets. They spend more time during the training process because of their eagerness to have a better generalization during the training from learning the weights, but they require less time to make predictions.

Most machine learning algorithms are eager learners, and below are some examples:

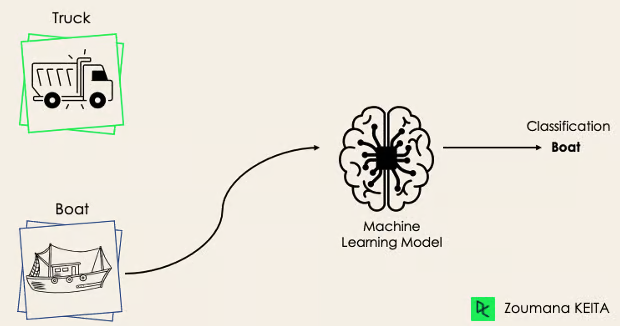
* Logistic Regression.
* Support Vector Machine.
* Decision Trees.
* Artificial Neural Networks.

**Lazy learners or instance-based learners**, on the other hand, do not create any model immediately from the training data, and this is where the lazy aspect comes from. They just memorize the training data, and each time there is a need to make a prediction, they search for the nearest neighbour from the whole training data, which makes them very slow during prediction. Some examples of this kind are:

* K-Nearest Neighbours.
* Case-based reasoning.
* **Types of Classification**

1. **Binary Classification:**

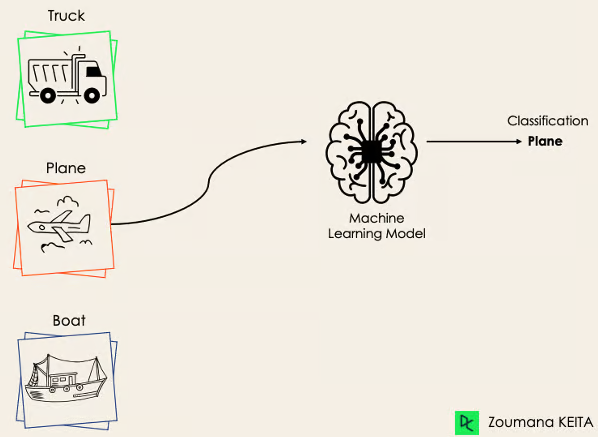
* In binary classification, the target variable has only two possible classes or categories.
* The goal is to classify the input data into two mutually exclusive categories.
* The training data in such a situation is labelled in a binary format: true and false; positive and negative; 0 and 1; spam and not spam, etc. depending on the problem being tackled.
* For instance, we might want to detect whether a given image is a truck or a boat.



* Logistic Regression and Support Vector Machines algorithms are natively designed for binary classifications. However, other algorithms such as K-Nearest Neighbours and Decision Trees can also be used for binary classification.

1. **Multi-Class Classification**

* In multi-class classification, the target variable has more than two possible classes, and each instance belongs to exactly one class.

****

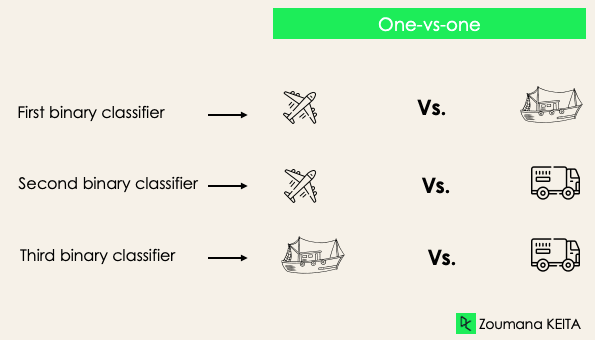
* Most of the binary classification algorithms can be also used for multi-class classification. These algorithms include but are not limited to:
* Random Forest
* Naive Bayes
* K-Nearest Neighbours
* Gradient Boosting
* SVM
* Logistic Regression.

*But wait! Didn’t you say that SVM and Logistic Regression do not support multi-class classification by default?*

*→ That’s correct. However, we can apply binary transformation approaches such as one-versus-one and one-versus-all to adapt native binary classification algorithms for multi-class classification tasks.*

**One-versus-one**: In One-vs-One, we train a **classifier for every pair of classes**. Each classifier **only distinguishes between two classes**.

For **N classes**, you need **N × (N – 1) / 2 classifier**.



Each classifier is trained on a single binary dataset, and the final class is predicted by a majority vote between all the classifiers. One-vs-one approach works best for SVM and other kernel-based algorithms.

**Pros:**

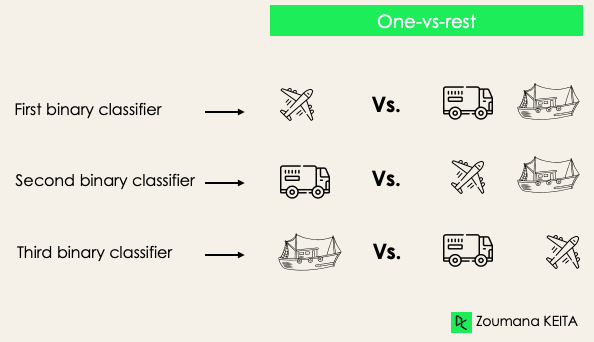
* Usually more accurate for small class counts
* Better handling of class overlap

**Cons:**

* Training time increases with number of classes
* More complex to manage predictions and conflicts

**One-vs-Rest:** In One-vs-Rest, we **build one binary classifier per class**. Each classifier tries to distinguish **one class** from **all other classes**.

For **N classes**, you train **N classifiers**



The classifier that gives the highest confidence score (e.g., probability) "wins."

**Pros:**

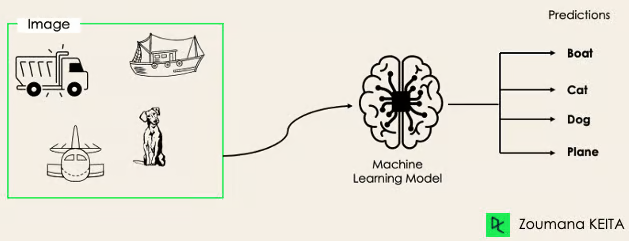
* Simpler and faster to train for large datasets
* Works well with unbalanced data

**Cons:**

* Can suffer when classes are very similar
* The “rest” group is often imbalanced

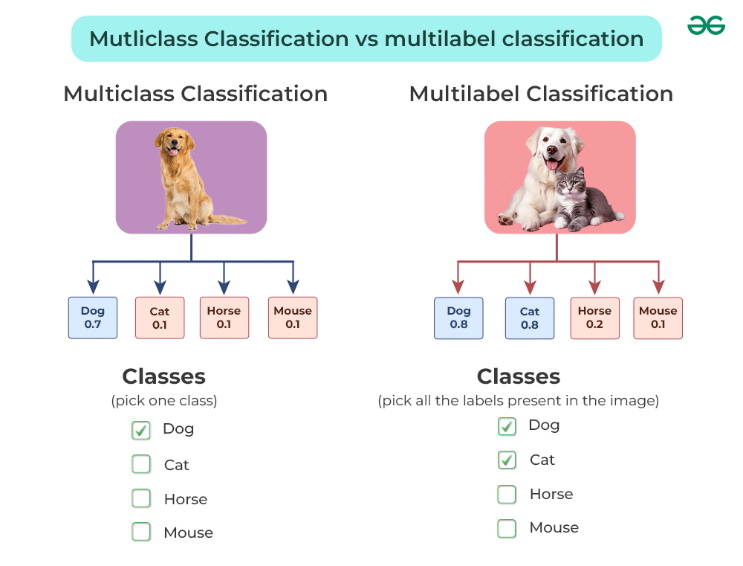
1. **Multi-Label Classification**

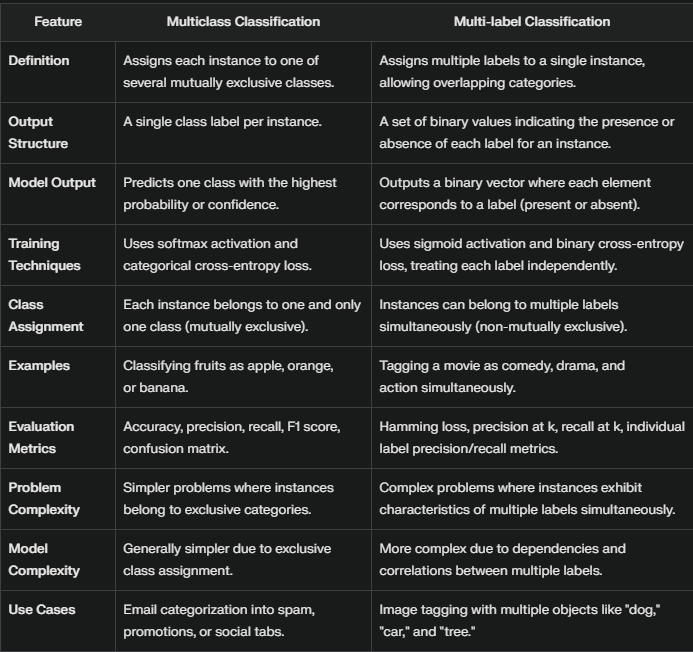
In multi-label classification, each instance can belong to multiple classes simultaneously.

****

It is not possible to use multi-class or binary classification models to perform multi-label classification. However, most algorithms used for those standard classification tasks have their specialized versions for multi-label classification.

* Multi-label Decision Trees
* Multi-label Gradient Boosting
* Multi-label Random Forests





1. **Imbalanced Classification**

Classification where the number of instances in different classes is highly imbalanced, often leading to challenges in model training and evaluation.  
Example:

* + Fraud detection (fraudulent transactions vs. legitimate transactions)
  + Rare disease detection (disease vs. healthy)

Using conventional predictive models such as Decision Trees, Logistic Regression, etc. could not be effective when dealing with an imbalanced dataset, because they might be biased toward predicting the class with the highest number of observations, and considering those with fewer numbers as noise.

**Sampling Techniques**

These techniques aim to balance the distribution of the original by:

* Cluster-based Oversampling:
* Random undersampling: random elimination of examples from the majority class.
* SMOTE Oversampling: random replication of examples from the minority class.

1. **Ordinal Classification**

Classification where the classes have a natural order, but the intervals between the classes are not necessarily equal or known.

**Examples**:

Customer satisfaction (very unsatisfied, unsatisfied, neutral, satisfied, very satisfied)

Credit ratings (A, B, C, D)

**Common Algorithms**:

Ordinal Logistic Regression

Decision Trees with ordinal splitting

Neural Networks adapted for ordinal output

1. **Hierarchical Classification**

Classification where classes are organized in a hierarchical structure, and predictions are made at multiple levels of the hierarchy.

**Examples**:

Document classification within a hierarchical taxonomy (e.g., science -> biology -> genetics)

Animal classification with taxonomic hierarchy (e.g., animal -> mammal -> carnivore -> dog)

**Common Algorithms**:

Hierarchical clustering combined with classification

Tree-based methods

Custom hierarchical models (neural networks designed for hierarchical classification)

1. **Single Class Classification or Anomaly Detection**

It involves identifying whether a new instance belongs to the normal class or is an anomaly.

**Examples**:

Network intrusion detection

Industrial equipment failure detection

**Common Algorithms**:

One-Class SVM

Isolation Forest

Autoencoders (neural networks for anomaly detection)

1. **Streaming Classification**

Classification where the data is continuously arriving over time, and the model needs to adapt incrementally.

**Examples**:

Real-time spam filtering

Stock market prediction

Online recommendation systems

**Common Algorithms**:

Incremental versions of traditional algorithms (e.g., incremental decision trees)

Online learning algorithms (e.g., Hoeffding Tree)

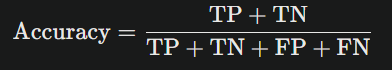
Adaptive algorithms designed for concept drift

Evaluation Metrics for Classification:

1. Accuracy:

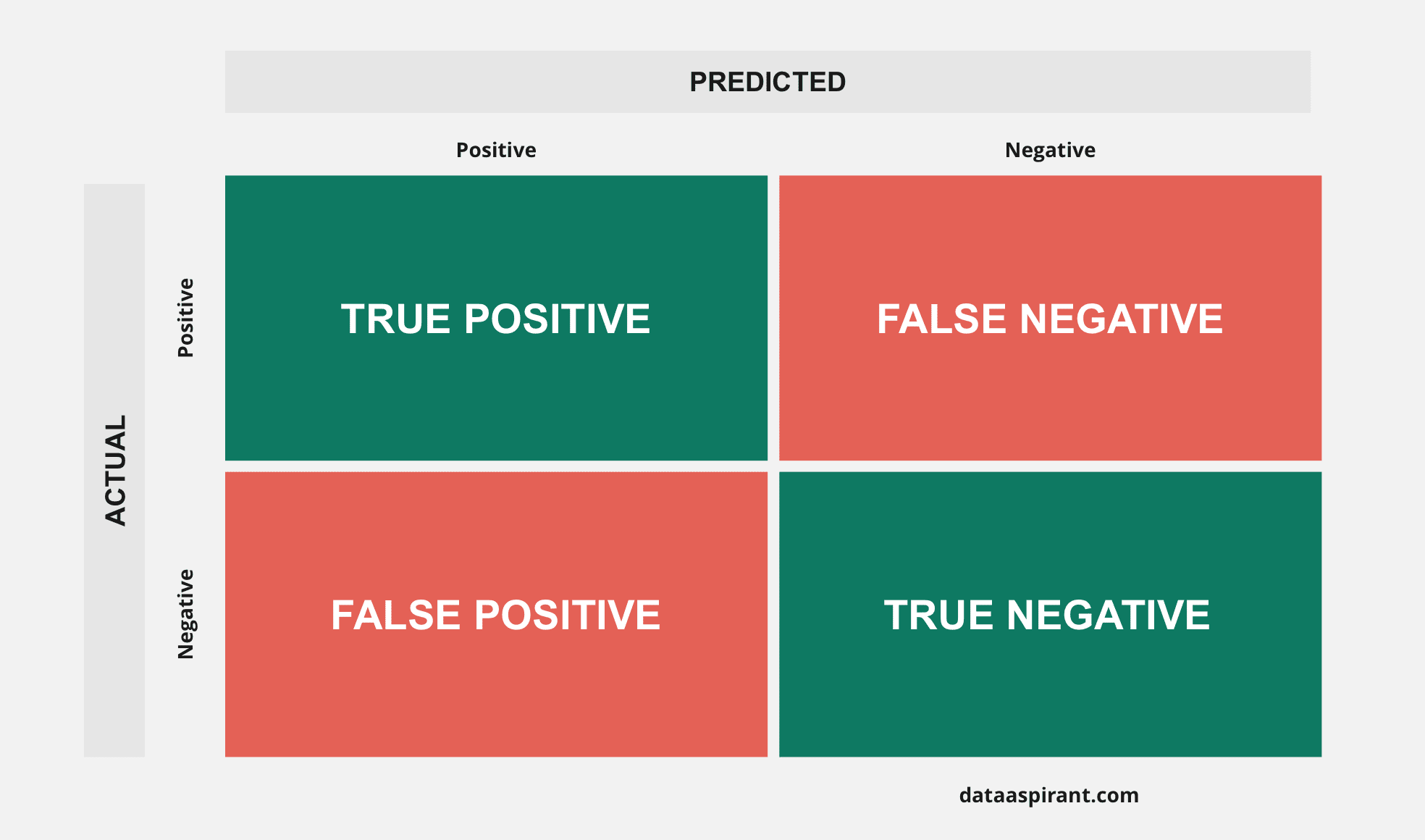
* It is calculated as the ratio of correct predictions to the total number of input samples.
* Suitable for balanced datasets where all classes have similar proportions.

: Avoid using accuracy for imbalanced datasets, as it may give misleading results by favouring the majority class



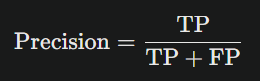
1. Confusion Metrics:

* A table summarizing the performance of a classification model with counts of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN)



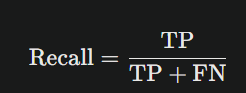
1. Precision:

* Precision is a measure of a model’s performance that tells you how many of the positive predictions made by the model are actually correct.



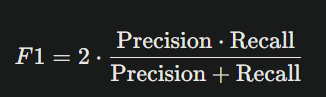
1. Recall:

* Recall is the ratio of correctly predicted positive instances to the total actual positive instances. It measures how well the model captures all relevant positive cases.



1. F1 Score

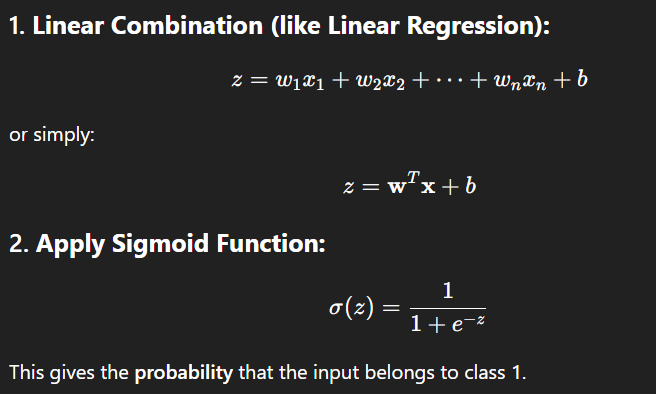
* Harmonic mean of precision and recall, balancing both metrics.

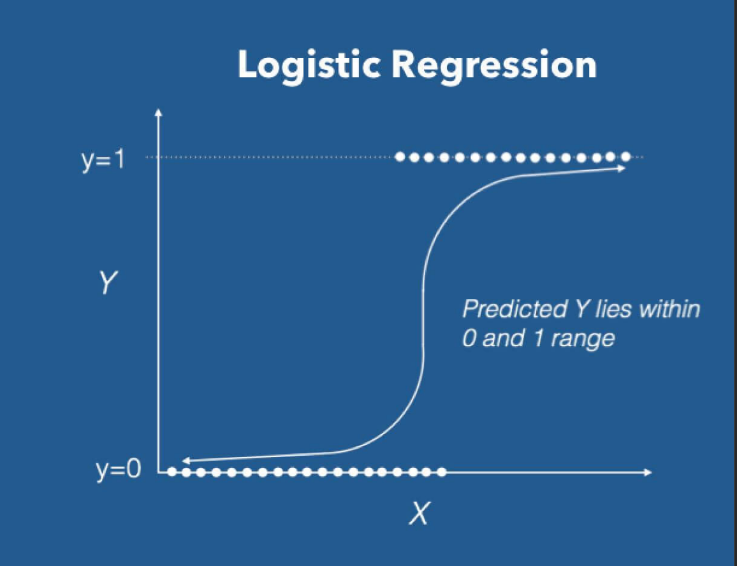


ALGORITHMS (CLASSIFICATION)

**Logistic Regression**

* Logistic Regression is a supervised learning algorithm used for classification tasks, making it ideal for binary classification problems
* Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value.
* Output is always between **0 and 1**
* Threshold is usually 0.5 used.
* In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
* It is especially useful when the dependent variable (the variable being predicted) is categorical or binary



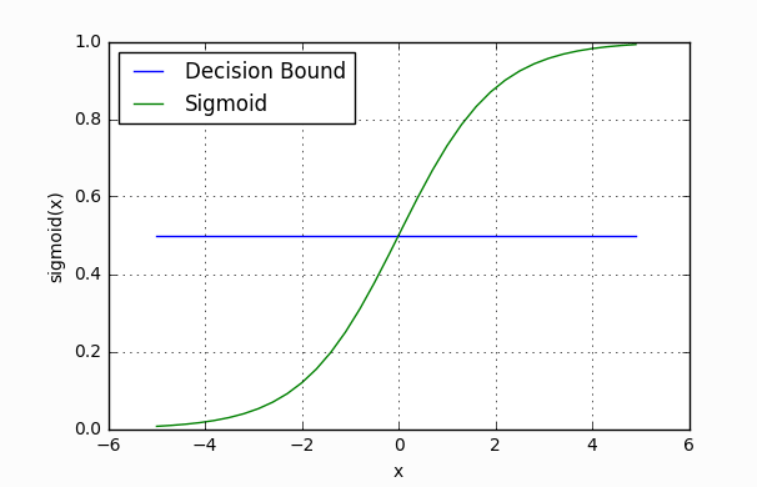


Types:

* 1. Binary Logistic Regression
* Used when the target variable has only two possible outcomes.
* Example: Email Spam or not Spam
* Sigmoid function is used
* Single output
  1. Multinominal Logistic Regression
     + Used when the targe variable has more than two classes, and the classes are mutually exclusive (each instance belongs to one class only)
     + Example: Classifying Animals: Cat, Dog, Buffalo, Weather Prediction
     + Softmax function is used
     + The class with the highest probability is selected
  2. Ordinal Logistic Regression
     + Used when the target variable has more than two classes, and the classes have a meaningful order or ranking
     + Example: High, Medium, Low
     + Models the cumulative probability

Decision Boundary:

* + - In a two-dimensional feature space, the decision boundary is represented by a line that separates the positive and negative instances.
    - The decision boundary is a threshold value that separates the instances.
    - The decision boundary is derived from the sigmoid function output, where the predicted probability is compared to the threshold value.
    - Instances with predicted probabilities above the threshold are classified as the positive class, and those below the threshold are classified as the negative class.
    - The decision boundary can be linear or non-linear, depending on the relationship between the input variables and the output probability.



**Pros**

* Simple and easy to implement
* Interpretable (you can examine coefficients)
* Fast to train
* Performs well on linearly separable data
* Provides **probabilities** for predictions

**Cons**

* Assumes a linear relationship (in log-odds)
* Doesn’t perform well on non-linear problems
* Sensitive to outliers
* Struggles with multicollinearity between features
* Less flexible than tree-based or neural network models

**Naive Bayes Classifier**

* + - Naïve Bayes algorithm is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems
    - It is mainly used in *text classification* that includes a high-dimensional training dataset.
    - The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, which can be described as:
    - Naïve: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features
    - **Bayes**: It is called Bayes because it depends on the principle of Bayes Theorem.

C:\Users\ADMIN\Desktop\naive-bayes-classifier-algorithm.png

**where,**

**P(A|B) is Posterior probability**: Probability of hypothesis A on the observed event B.

**P(B|A) is Likelihood probability**: Probability of the evidence given that the probability of a hypothesis is true.

**P(A) is Prior Probability**: Probability of hypothesis before observing the evidence.

**P(B) is Marginal Probability**: Probability of Evidence.

**Pros**

* Simple and easy to implement.
* Works well with high-dimensional data.
* Requires less training data.
* Fast in both training and prediction.

**Cons**

* Assumes feature independence, which is rarely true in real data.
* Can perform poorly if this assumption is violated.
* Not suitable for datasets with highly correlated features

**Applications**

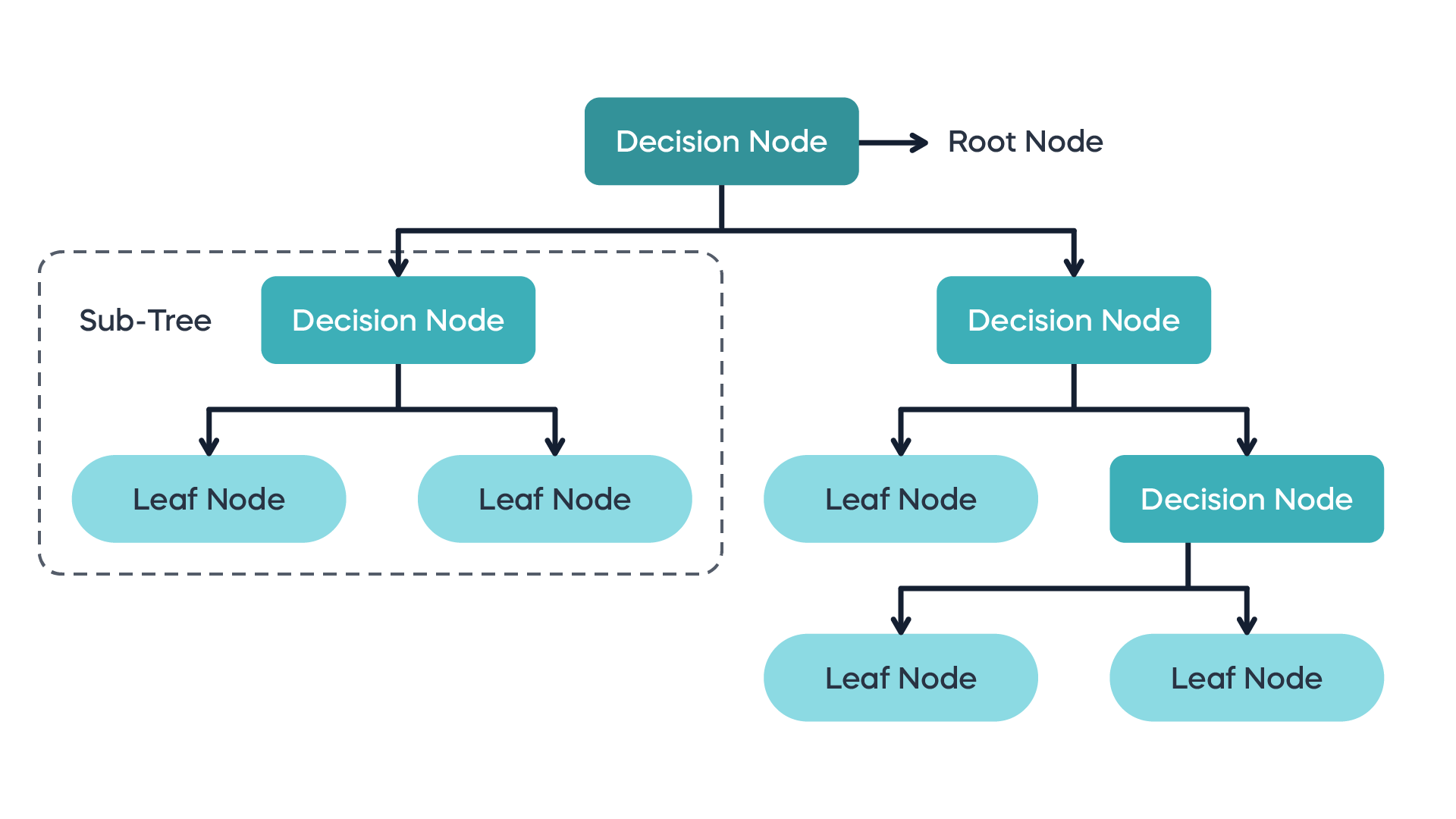
* Email spam filtering
* Document categorization
* Sentiment analysis

Types:

1. Gaussian
   * + When the **features are continuous** (real-valued, like age, height, income, etc.)
     + Assumes that **feature values follow a Normal (Gaussian) distribution** within each class
2. Multinominal
   * + When the features represent **discrete counts** (e.g., word frequency in a document)
     + Best suited for **text classification**, especially in **bag-of-words** or **TF-IDF** format
3. Bernoulli
   * + When features are **binary** (i.e., 0 or 1, indicating **presence or absence** of a feature)
     + Also useful for **text classification**, but instead of word count, it considers **whether a word exists or not**

**Decision Tree**

* + - Decision Tree is a **Supervised learning technique**that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.
    - ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***
    - It is a tree-structured classifier, where **internal nodes** represent the **features of a dataset**, **branches** represent the **decision rules** and each **leaf node** represent the **outcome**.
    - In Decision Tree there are two nodes,
    - Decision Nodes: Used to make any decision and have multiple branches
    - Leaf Nodes: Outputs of those decisions and do not contain any further branches.
* In order to build a tree, we use the **ID3** and **CART algorithm,** which stands for **Iterative Dichotomiser 3 / Classification and Regression Tree algorithm.**



**Pros**

* Easy to understand and interpret.
* Can handle both categorical and numerical data.
* No need for feature scaling or normalization.
* Can capture complex non-linear relationships.

**Cons**

* Prone to overfitting, especially with deep trees.
* Sensitive to small changes in the training data.
* Can create biased trees if some classes dominate.

Step-by-Step Process of how a decision tree works

**Step 1: Start with the Entire Dataset**

* You begin with all training data at the **root node**.
* Each instance has features (input variables) and a label (target/output).

**Step 2: Select the Best Feature to Split On**

* The tree decides **which feature** to split the data on by evaluating all features using a splitting criterion: **Attribute Selection Method**

**Common Criteria:**

* **For Classification Trees**:
  + **Gini Impurity**
  + **Information Gain** (using **Entropy**)
* **For Regression Trees**:
  + **Variance Reduction** or **Mean Squared Error (MSE)**

**Step 3: Create a Decision Node**

* Once the best feature is chosen, a **decision node** is created based on a **threshold or category**.
* The dataset is split into **two or more branches**, based on the feature's values.

**Step 4: Repeat for Each Branch**

* For each child node (branch), repeat:
  + Recalculate the best feature to split that subset
  + Split again
  + Create new nodes

**Step 5: Stopping Conditions**

The recursion stops when one or more of these conditions are met:

* All instances in a node belong to the **same class**
* Maximum tree **depth** is reached
* The node has too **few samples** to split
* No further **information gain** can be achieved

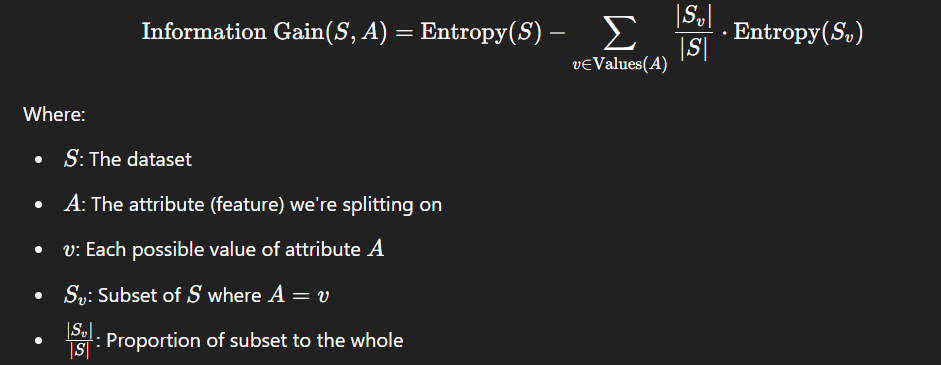
**Step 6: Assign Leaf Nodes**

* When a node can't be split further, it becomes a **leaf node**.
* The leaf node is assigned a **class label** (in classification) or a **predicted value** (in regression), usually the **majority class** or **average value** in that node.

**Attribute Selection Methods**:

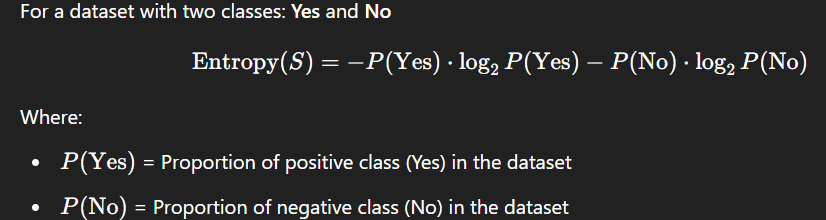
By using this method, we can select the best attributes for the nodes of the tree.

* 1. Information Gain
     + Information Gain is the measurement of changes in entropy.
     + It calculates how much information a feature provides us about a class.
     + According to the value of information gain, we split the node and build the decision tree.
     + A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having information gain is split first.

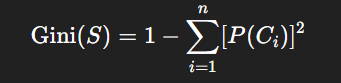


**Entropy:**

* + - Entrop is a metric to measure the impurity in a given attribute.
    - It specifies the randomness in data.



* 1. Gini Index
     + Gini Index is a measure of impurity or purity used while creating a Decision Tree.
     + It uses CART (Classification and Regression Tree) algorithm.
     + An attribute with the **low** Gini index should be **preferred** as compared to the high Gini index.
     + It only **creates binary splits**, and the CART algorithm uses the Gini index to create binary splits.



* 1. **Pruning**
     + Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.
     + A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset
     + Therefore, a technique that decreases the size of the learning tree without reducing accuracy is known as Pruning.
     + There are mainly two types of tree **pruning**technology used:
     + **Cost Complexity Pruning**
     + **Reduced Error Pruning.**

ERROR BOUNDS:

* + - Error bounds are estimations that provides uncertainty or potential error made by the predictions of the model.
    - They provide a range within which the predicted value might lie, giving a measure of reliability and accuracy of the model.
    - It is necessary for both theoretical and practical application of statistics and machine learning.

Types:

* + - Confidence Intervals:

A confidence interval provide a range of values, derived from the sample data, that is likely to contain the true value of an unknown population parameter.

* **Example**: If a 95% confidence interval for a population mean is (5, 10), we are 95% confident that the true mean lies between 5 and 10.
* 
  + - Prediction Intervals
      * A prediction interval provides a range within which a single new observation is expected to fall with a certain probability
      * A 95% prediction interval for the next value in a series might be (8, 12), meaning there's a 95% chance the next observed value will lie between 8 and 12.
    - Chebyshev’s Inequality
    - Hoeffding’s Inequality
    - PAC Bounds

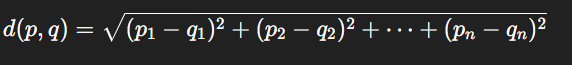
**K-Nearest Neighbour (KNN) Algorithm**

* + - K-Nearest Neighbour is one of the simplest machine learning algorithms based on Supervised Machine Learning
    - KNN is a **lazy learner** – it doesn’t build a model during training; it stores the data and learns at prediction time.
    - K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
    - K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
    - K-NN is a **non-parametric** algorithm, which means it does not make any assumption on underlying data.

How KNN Classification Works

* + - **Choose a value of K**  
      → Number of neighbours to consider
    - **Compute the distance** between the new input point and all points in the training set

Usually **Euclidean distance**, but can use others (Manhattan, Minkowski, etc.)



* + - **Sort the distances** and pick the **K nearest neighbours**
    - **Assign the class label** based on **majority vote** among K neighbours
    - **Output** the predicted class

**Choosing the Right K**

* **Too small (e.g., K=1)** → High variance, overfitting
* **Too large** → High bias, underfitting

✅ Tip: Use **odd K** for binary classification to avoid ties  
✅ Use **cross-validation** to find the optimal K

**Pros**

* Simple, intuitive
* No training time
* Works well for small datasets
* Adapts to nonlinear boundaries

**Cons**

* **Slow** prediction (since it computes distance from all training points)
* **Sensitive to irrelevant features and feature scaling**
* **Curse of dimensionality** (performance drops in high dimensions)
* **Memory-intensive**

**Applications**

* **Image recognition**
* **Recommender systems**
* **Medical diagnosis**
* **Handwriting detection**
* **Customer segmentation**

**Best Practices**

* **Normalize/Standardize** features (KNN is distance-based)
* **Feature selection** to reduce irrelevant dimensions
* Use **KD-Trees** or **Ball Trees** for fast searching (on large datasets)
* Apply **Cross-Validation** to tune K



**PERCEPTRON:**

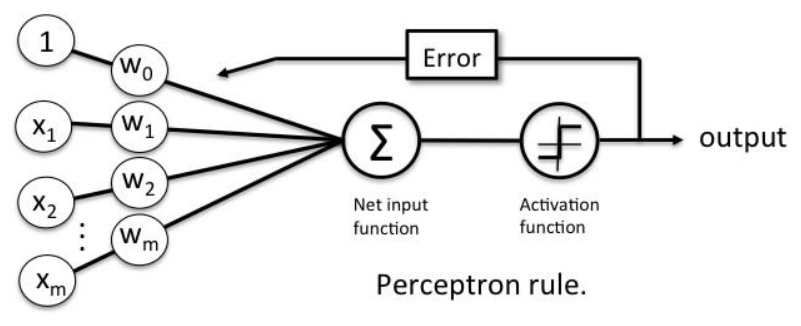
* + - A Perceptron is the simplest type of artificial neural network, used primarily for binary classification.
    - It models a biological neuron and learns a linear decision boundary to classify data points.
    - A Perceptron is an algorithm for supervised learning of binary classifiers
    - There are two types of Perceptron’s
    - Single layer and Multilayer.
    - Single layer Perceptron’s can learn only linearly separable patterns.
    - Multilayer Perceptron’s or feed forward neural networks with two or more layers have the greater processing power. Has **hidden layers**, can solve **non-linear problems** (uses activation like ReLU, Sigmoid)
    - The Perceptron algorithm learns the weights for the input signals in order to draw a linear decision boundary.
    - This enables you to distinguish between the two linearly separable classes +1 and -1.

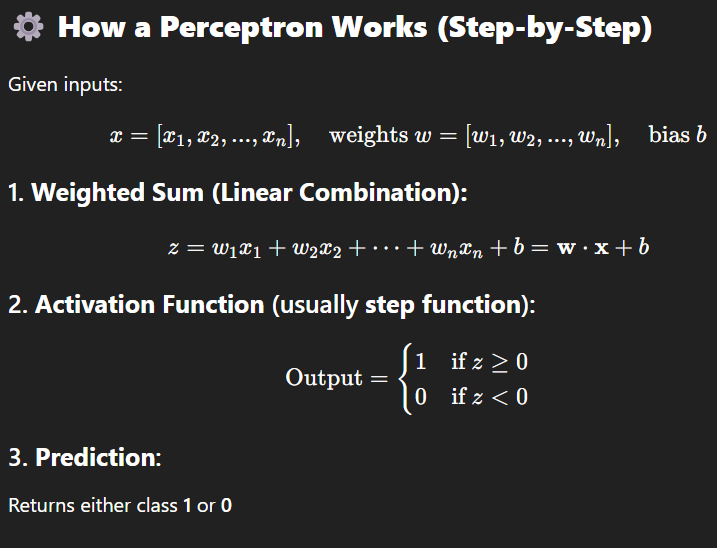
**Advantages**

* Simple and easy to implement
* Fast training for small datasets
* Useful for understanding deeper neural networks

**Limitations**

* **Only works for linearly separable data**
* Can’t solve problems like **XOR** without hidden layers
* Limited in expressive power





**SUPPORT VECTOR MACHINE: -**

* + - SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression Problems.
    - However, primarily, it is used for Classification Problems in ML.
    - The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data in the correct category in the future.
    - There can be multiple lines/ decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points.
    - This best boundary is known as hyperplane of SVM.
    - The dimensions of the hyperplane depend on the features present in the dataset.
    - If we have 2 features the hyperplane will be a straight line.
    - If we have 3 features the hyperplane will be a 2-dimension plane.
    - We always create a hyperplane that has a maximum margin.
    - The data points or vectors that are closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector.

Types:

1. Linear SVM

Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a straight line, then such data is termed as linearly separable data.



1. Non-linear SVM

Non-linear SVM is used for non-linearly separated data, which means if a dataset can’t be classified by using a straight line.

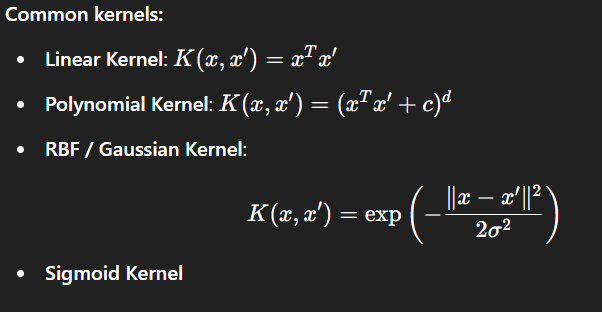
* 1. Soft Margin

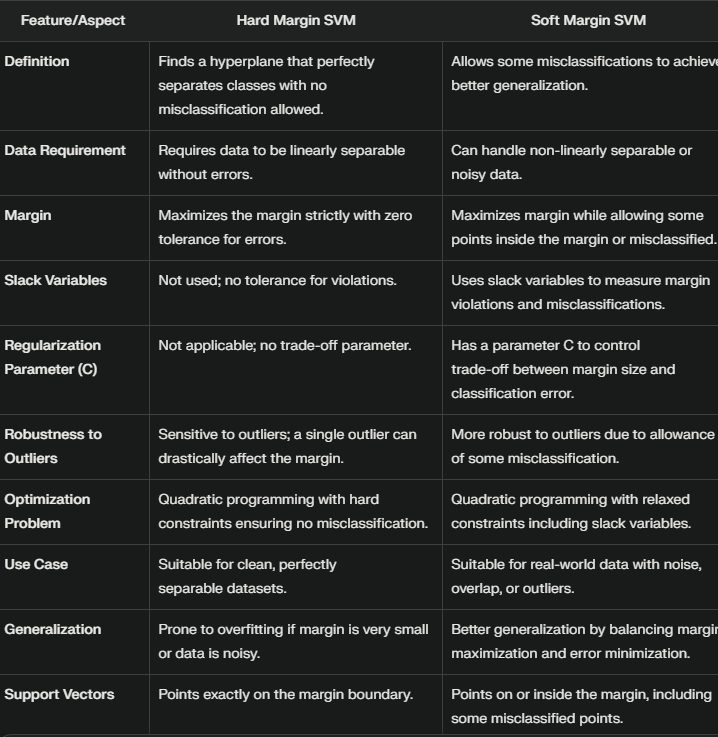
In a **hard margin SVM**, we strictly **do not allow any misclassification** — this only works if the data is **perfectly linearly separable**.

But in real-world data, we often:

* Have noise
* Have overlapping classes
* Can’t separate classes with a perfect margin

So, we introduce the **Soft Margin SVM**, which allows **some misclassified points** for better **generalization**.

* 1. Kernel Trick
     + The kernel trick allows a **linear algorithm to learn nonlinear decision boundaries** by **implicitly mapping input data into a higher-dimensional feature space** where the **data becomes linearly separable.**
     + Enables linear algorithms like SVMs to solve nonlinear classification and regression problems.
     + 



**Unsupervised Learning**

* + - Unsupervised learning is a learning method in which a machine learns without any supervision.
    - The training is provided to the machine with the set of data that has not been labelled, classified, or categorized, and the algorithm needs to act on that data without any supervision.
    - The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.
    - In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data.

Advantages of Unsupervised Learning

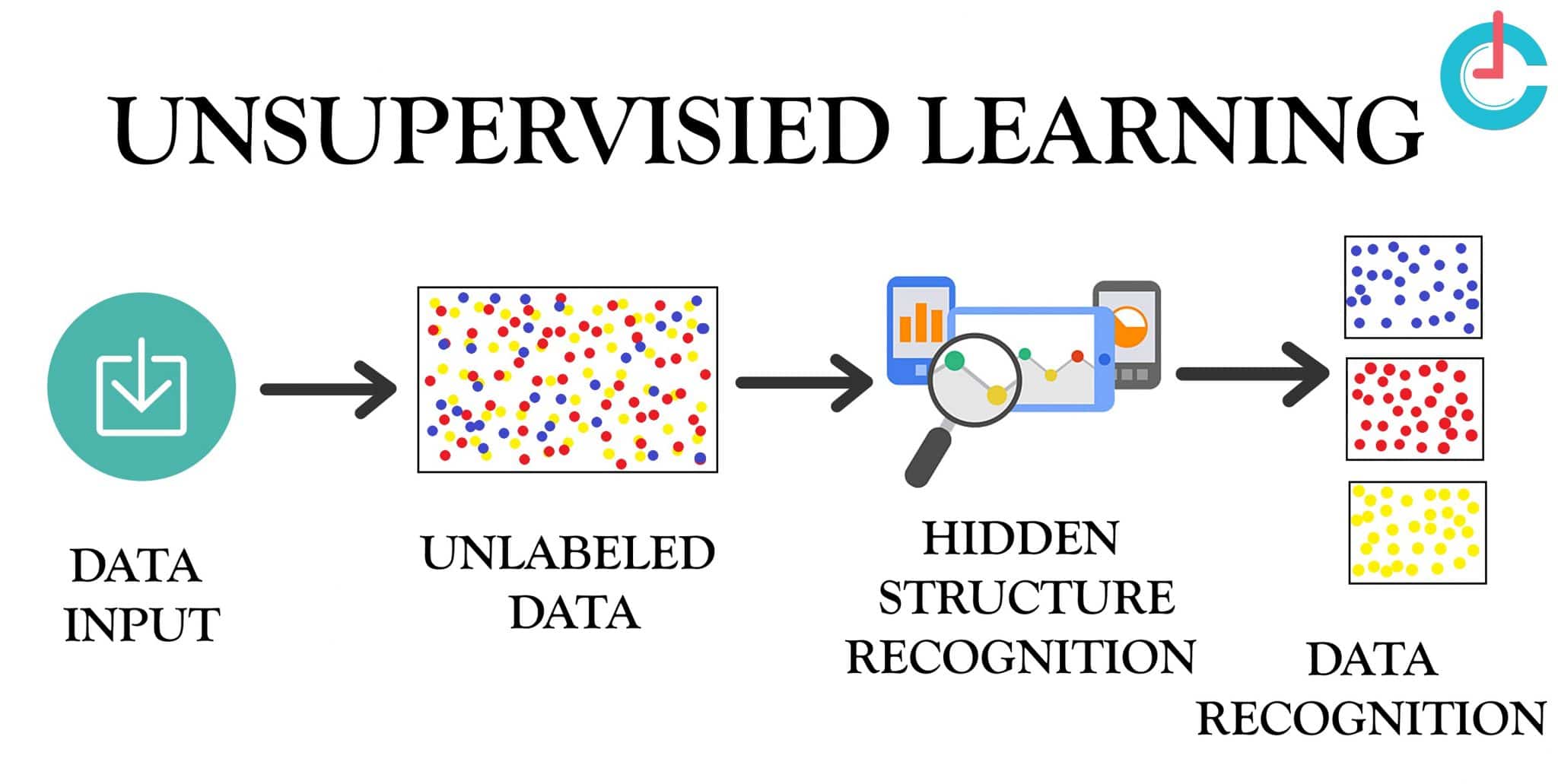
• Unsupervised learning is used for more complex tasks as compared to supervised learning because, in unsupervised learning, we don't have labelled input data.

• Unsupervised learning is preferable as it is easy to get unlabelled data in comparison to labelled data

Disadvantages of Unsupervised Learning

• Unsupervised learning is intrinsically more difficult than supervised learning as it does not have corresponding output.

• The result of the unsupervised learning algorithm might be less accurate as input data is not labelled, and algorithms do not know the exact output in advance

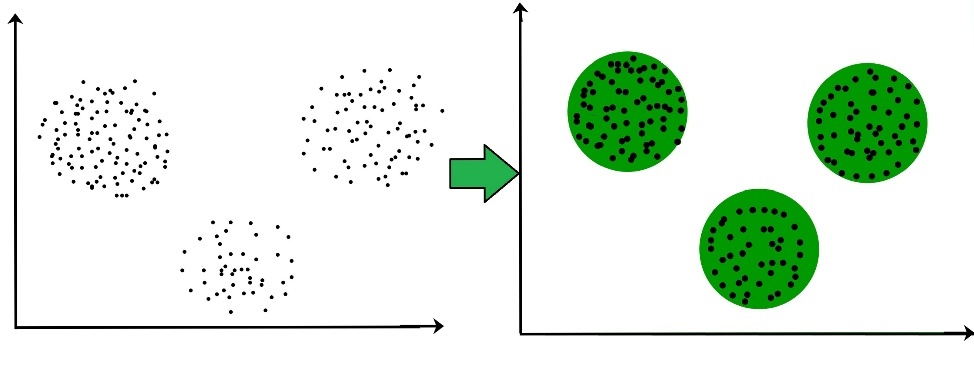


Applications:

* + - Customer Segmentation
    - Anomaly Detection
    - Market Basket Analysis
    - Dimensionality Reduction
    - Word Embeddings

**CLUSTERING**

* + - Clustering is an unsupervised learning technique that involves grouping a set of data points into clusters, such that:
      * Points in the same cluster are more similar to each other,
      * Points in different clusters are as dissimilar as possible.
    - It does not require labelled data and is often used for pattern discovery, segmentation, and data summarization.
    - It is an unsupervised learning method, hence no supervision is provided to the algorithm, and it deals with the unlabeled dataset



Types:

1. **Partitioning Clustering**
   * + It is a type of clustering that divides the data into non-hierarchical groups. It is also known as the **centroid-based method**. The most common example of partitioning clustering is the **K-Means Clustering algorithm.**
     + In this type, the dataset is divided into a set of k groups, where K is used to define the number of pre-defined groups
     + The cluster center is created in such a way that the distance between the data points of one cluster is minimum as compared to another cluster centroid.
2. **Density-Based Clustering**
3. **Distribution Model-Based Clustering**
4. **Hierarchical Clustering**
5. **Fuzzy Clustering**