MACHINE LEARNING ALGORITHMS

MACHINE LEARNING ALGORITHMS

Supervised learning is a type of machine learning where the algorithm is trained on a labeled dataset, which means each input has a corresponding correct output. The main goal is to learn from this data and make accurate predictions on new, unseen data. It works by mapping inputs to known outputs using different models like Linear Regression, Decision Trees, Random Forest, Support Vector Machines (SVM), and Neural Networks. Supervised learning is commonly used in real-world problems like spam email detection, predicting diseases, or whether a loan should be approved based on a person's profile.

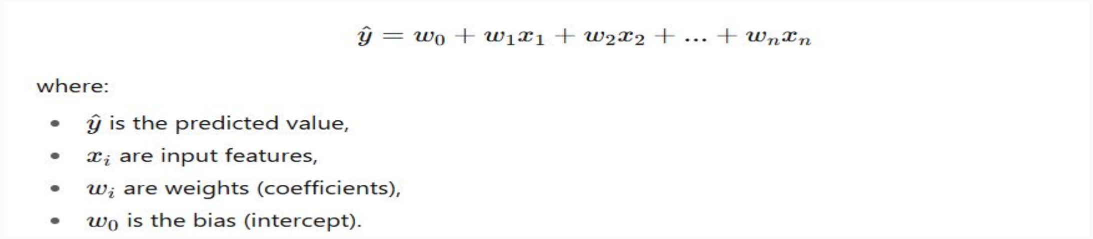
Unsupervised learning deals with data that has no labels, meaning the algorithm is not given the correct answers during training. Instead, it tries to find hidden patterns or structures in the data on its own. This type of learning is useful for grouping similar items (clustering) or reducing the number of features while keeping the important ones (dimensionality reduction). Popular algorithms include Clustering, PCA. Common use cases include customer segmentation, fraud detection, and organizing large sets of data without knowing what categories exist in advance.

MACHINE LEARNING ALGORITHMS

Reinforcement Learning (RL) is a branch of machine learning where an agent learns to make decisions by interacting with an environment to achieve a specific goal. Instead of being explicitly told what actions to take, the agent explores and receives feedback in the form of rewards or penalties, which guide it toward better strategies over time. The process involves key concepts like states (the current situation), actions (choices available to the agent), and rewards (feedback signals). RL is widely used in areas such as robotics, game playing, autonomous driving, as it enables systems to learn optimal behaviors through trial and error while adapting to dynamic environments.

LINEAR REGRESSION

Linear Regression is one of the fundamental supervised learning algorithms used for predicting continuous numerical values. At its core, it attempts to model the relationship between a dependent variable (target) and one or more independent variables (features) by fitting a linear equation to the observed data.



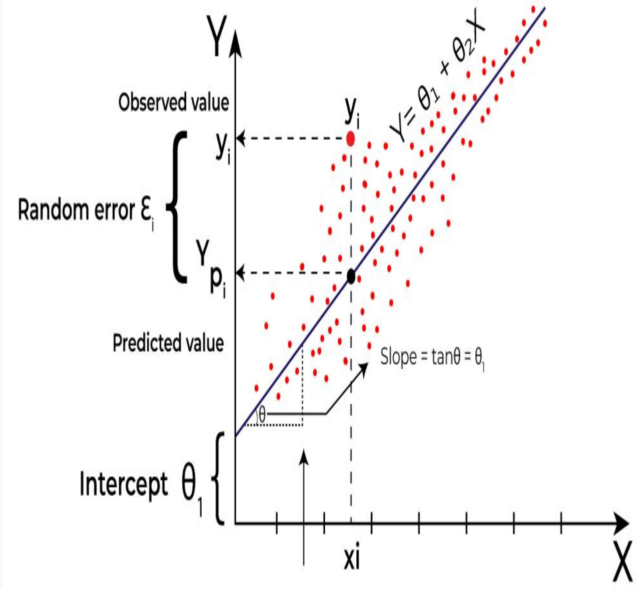
$$\hat{y} = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$

where:

- $\hat{y}$ is the predicted value,
- x_i are input features,
- w_i are weights (coefficients),
- w_0 is the bias (intercept).

The “architecture” is essentially **a single layer with weighted connections**—there’s no deep structure like neural networks.
Types:

- **Simple Linear Regression**: One independent variable
- **Multiple Linear Regression**: Multiple independent variables
- **Polynomial Regression**: Non-linear relationships using polynomial features



Y
$Y = \theta_1 + \theta_2 X$
Observed value
y_i
y_i
Random error ε_i
Y_{p_i}
Predicted value
Slope = $\tan\theta = \theta_1$
θ
Intercept θ_1
xi
X

LINEAR REGRESSION

COST FUNCTION:

- The cost function measures how far predictions are from actual values.
- For Linear Regression, **Mean Squared Error (MSE)** is commonly used as cost.
- Goal: minimize J(w) by finding optimal weights w.

$$J(w) = \frac{1}{2n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Why squared error?

- Squaring avoids canceling out positive and negative errors.
- Larger errors are penalized more heavily, encouraging better overall fit.

Why the 1/2 factor?

- It simplifies the derivative during optimization (gradient descent)

EXAMPLE:In house price prediction, the cost function measures how far predicted prices are from actual sale prices.

MSE penalizes big pricing mistakes heavily, and optimization finds the model weights that make these errors as small as possible.

USE CASES

Predicting house prices based on size, location, and features.

Estimating product sales from advertising spend.

Predicting a student’s final score from study hours and attendance.

WHY LINEAR REGRESSION?

- Many housing features (size in sqft, number of rooms, location score) have an approximately **linear relationship** with price, making linear regression a simple yet effective baseline.
- Coefficients let you **interpret the impact** of each feature , which is useful for decision-making

- Sales often show a **proportional increase** with ad spend up to a point, so a linear fit provides a quick estimate of returns.
- The model helps **quantify ROI** . e.g., “Every $1,000 spent increases sales by Y units” , enabling budget planning.

- Study hours and attendance tend to have a **positive, roughly linear effect** on exam scores, making the model easy to interpret for educators.
- Allows **early intervention** by predicting at-risk students’ scores and suggesting improvements in study habits or attendance.

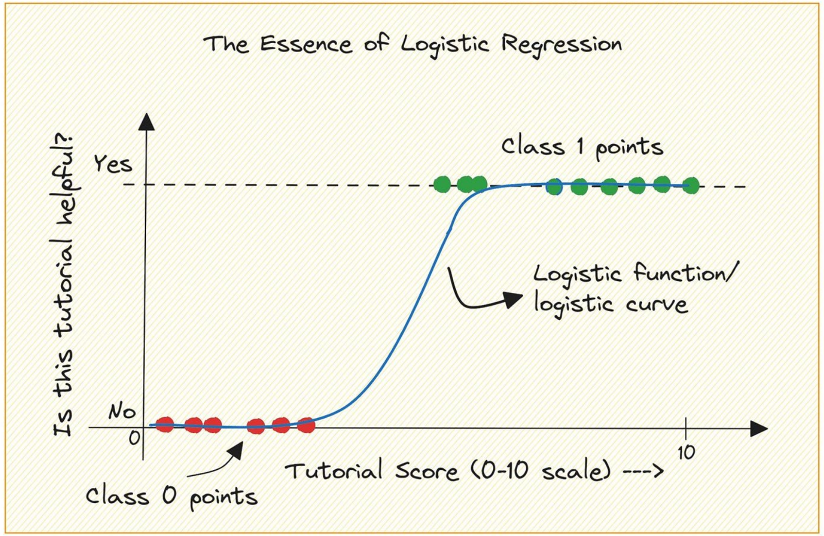
LOGISTIC REGRESSION

Logistic Regression is a supervised classification algorithm that predicts probabilities of class membership using a sigmoid (logistic) function to map linear combinations of features to values between 0 and 1.

$$\hat{y} = \sigma(w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n)$$

Where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid function.

Input:
Takes in one or more independent variables/features. These can be numerical or categorical (after encoding).
Output:
Probability of class (e.g., spam or not spam), which is then thresholded (usually at 0.5) to classify.



LOGISTIC REGRESSION

COST FUNCTION:

- The cost function measures how far predictions are from actual values.
- Binary Cross-Entropy / Log Loss
- Goal: minimize J(w) by finding optimal weights w.

$$J(w) = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right]$$

Purpose:

- Penalizes confident wrong predictions more than less confident ones.
- Encourages the model to produce probabilities close to the true class label.

Why Not MSE?

- MSE leads to non-convex cost in logistic regression, making optimization harder.
- Cross-Entropy is convex and works better for classification problems.

LOGISTIC REGRESSION

ACTIVATION FUNCTION:

- Sigmoid Function

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Purpose:

- Converts the linear combination of inputs into a probability between 0 and 1.
- Helps in classifying outputs based on a threshold (commonly 0.5).

Why Sigmoid?

- It maps any real-valued number to a probability.
- Smooth and differentiable, which is useful for gradient descent optimization.

USE CASES

WHY LOGISTIC REGRESSION?

Email Spam Detection

- Emails are either spam or not spam ,logistic regression is built for **yes/no problems**.
- It gives a **probability** (e.g., 85% spam) so you can adjust the threshold depending on how strict you want the filter.

Customer Churn Prediction

- Outcome is **binary**: customer either leaves (churns) or stays.
- Probability output helps **prioritize** customers most at risk so you can take action early.

Disease Diagnosis

- Diagnosis is often **yes/no**: has the disease or not.
- Logistic regression gives a probability, which is useful for **risk-based medical decisions** rather than hard yes/no.

SUPPORT VECTOR MACHINE

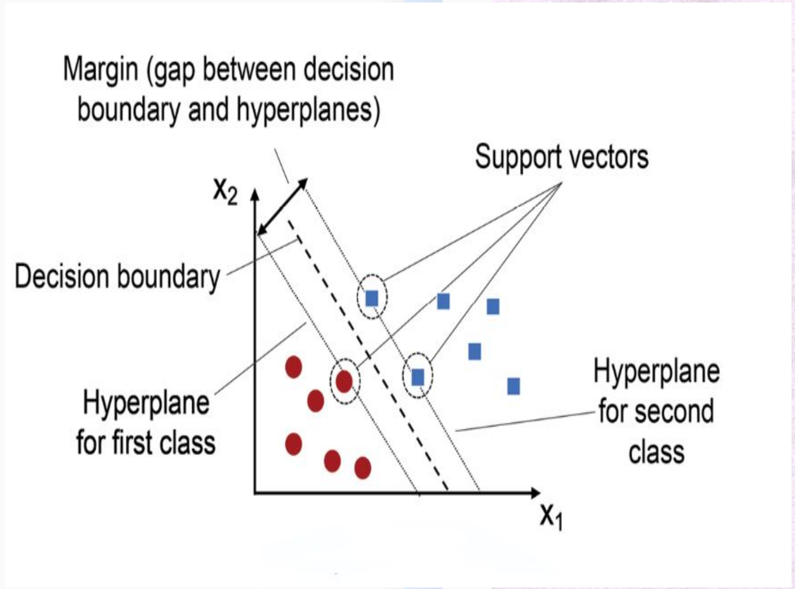
Support Vector Machine is a **supervised learning algorithm** used for **classification** and **regression**, mainly known for **binary classification**.

SVM finds the **optimal hyperplane** that best separates the data points of two classes with the **maximum margin**.
The **margin** is the distance between the hyperplane and the nearest data points from each class (called **support vectors**).

In **non-linearly separable cases**, SVM uses **kernel functions** (like RBF or polynomial) to project data into higher dimensions where it becomes linearly separable.

Key Components:

- **Hyperplane:** Decision boundary
- **Support Vectors:** Critical data points closest to the hyperplane
- **Margin:** Distance from support vectors to hyperplane (SVM maximizes this)


Margin (gap between decision boundary and hyperplanes)
Support vectors
X_2
Decision boundary
Hyperplane for first class
Hyperplane for second class
X_1

SUPPORT VECTOR MACHINE

COST FUNCTION:

- The cost function measures how far predictions are from actual values.
- **Hinge Loss Function** (for hard-margin SVM):

$$J(w) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \max(0, 1 - y_i(w^T x_i + b))$$

- **Goal:** Find the optimal weights w and bias b that define the hyperplane with the maximum margin

Purpose:

- The first term $\frac{1}{2}||w||^2$ tries to maximize the margin (keep weights small)
- The second term penalizes misclassified or margin-violating points
- C is a regularization parameter controlling the trade-off between maximizing margin and minimizing classification error

USE CASES

WHY SVM?

Email Spam Face Detection

- SVM handles **high-dimensional data** really well (emails become thousands of word features).
- It can find a **clear separating boundary** between spam and non-spam with few important examples

Text Classification

- Text features are **sparse but high-dimensional**,SVM is strong in that setting.
- With the right kernel, SVM captures **non-linear patterns** in word usage for better classification.

Handwritten Digit Recognition

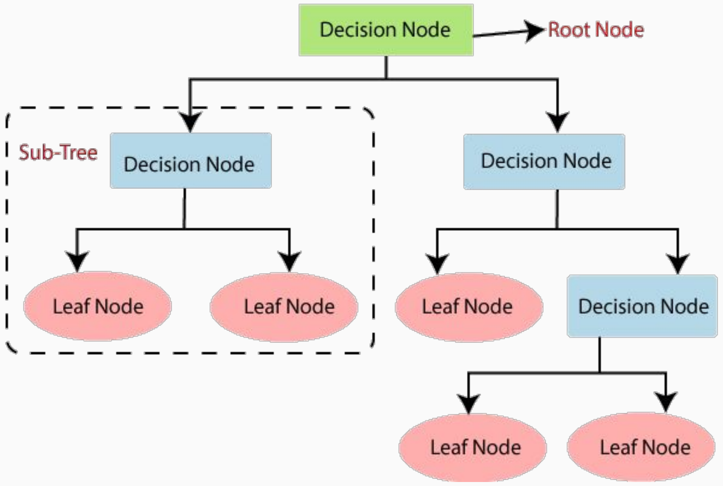
- SVM with kernels can **separate complex patterns** in pixel data that aren’t linearly separable.
- Often achieves **high accuracy** with relatively small training sets .

DECISION TREE

A **Decision Tree** is a **supervised learning algorithm** that can solve **classification** and **regression** problems, but it's most widely used for **classification tasks** due to its **simple, rule-based structure**.

A **decision tree** is a rule-based model that splits data step-by-step like a flowchart of nodes and branches.

- **Root Node:** Holds the full dataset; first split uses the most informative feature.
- **Internal Nodes:** Decision points testing specific features.
- **Branches:** Outcomes of decisions.
- **Leaf Nodes:** Final predictions (class labels or numeric values).
  The tree grows through **recursive splitting**, choosing the best feature at each step until a stopping condition is met.



DECISION TREE

COST FUNCTION:

In Decision Trees, the **cost function** is not like a single global loss (as in regression or logistic regression).Instead, at each **split (internal node)**, the algorithm uses a **"split criterion"** (also called **impurity measure**) to decide **which feature and value to split on**. The goal is to **maximize the "purity"** of the resulting nodes.

What does the cost function measure?

- It measures **how mixed or impure** the data in a node is.
- A **pure node** = All data points belong to the same class.
- The **lower the impurity**, the better the split.

Entropy

$$H(P) = -\sum_{x \in C} P(x) \log P(x)$$

Information Gain :tells us how much "uncertaintv" is reduced by the split.

$$IG = Entropy_{parent} - \sum_{j} \frac{n_j}{n} \cdot Entropy_{child_j}$$

DECISION TREE

COST FUNCTION:

The **Gini Index** is a measure used in decision trees to determine the impurity of a dataset. It calculates how often a randomly chosen element would be incorrectly classified if it was randomly labeled according to the distribution of labels in the dataset. A lower Gini Index indicates a purer node.

$$\text{Gini Index} = 1 - \Sigma\,(\,pi\,)^2$$

If the Gini Index = 0 . The node is *pure*, meaning all samples belong to the same class.

If the Gini Index is high . The node is *impure*, meaning samples are spread across multiple classes.

USE CASES

WHY DECISION TREE?

Classifying customers into "High Risk" or "Low Risk" for insurance.

- Can handle both numerical (age, income) and categorical (smoker, region) data without much preprocessing.
- Produces clear decision rules, making it easy for insurers to explain decisions.

Predicting whether a student will pass based on attendance and assignment scores.

- Naturally models threshold-based decisions (“If attendance < 70% , at risk”).
- Easy to visualize and interpret, so teachers can understand and act on the rules.

Determining eligibility for social welfare schemes.

- Handles complex eligibility criteria involving multiple yes/no conditions.
- Can process missing data and mixed data types without heavy data cleaning.

COMPARISON :

LINEAR REGRESSION	LOGISTIC REGRESSION	SVM	DECISION TREES
Type of Output:Predicts continuous values.	Type of Output: Predicts probabilities.	Type of Output:Predicts Class labels.	Type of Output:Predicts Class labels
Model Assumption: Assume linear relationship between features and output.	Model Assumption: Assume linear relationship	Model Assumption: No strict assumption	Model Assumption: No assumption
Interpretability:Easy to interpret coefficients.	Interpretability:Easy to interpret coefficients.	Interpretability:Harder to interpret, especially with non-linear kernels.	Interpretability: Very interpretable via decision rules.
USECASE: Forecasting numerical trends (e.g., house prices).	USECASE: Binary/multi-class classification e.g., churn prediction).	USECASE: High-dimensional classification (e.g., handwriting recognition).	USECASE: Rule-based decisions (e.g., loan approval).