







**Supervised learning** is a type of machine learning where the model is trained on **labelled data**, meaning that each input has a corresponding output or target value. The goal of supervised learning is to learn a function that can map inputs to outputs based on the given examples. There are many types of supervised learning algorithms that can be categorized into two main groups: **regression** and **classification**.

**Categorical features**: Converting categorical features into a numerical format (e.g., one-hot encoding, label encoding) is a pre-processing step to ensure that machine learning algorithms can work with the data.**Handling missing data**: Dealing with missing data by imputation, deletion, or other methods is an essential pre-processing step to ensure the quality and consistency of the dataset.**Data normalization**: Scaling and normalizing data to a specific range or distribution is a pre-processing step to ensure that features are on a similar scale, which can improve the performance of many machine learning algorithms.

Overfitting is a common problem in machine learning where a model learns the training data too well, **including noise and random patterns that are not representative of the underlying relationship between input features and the target variable.** As a result, the model becomes too complex and performs poorly on new, unseen data. Regularization is a technique used in machine learning to prevent overfitting and improve the generalization ability of a model by adding a penalty term to the loss function. By doing so, it discourages the model from relying too heavily on individual features or fitting the noise in the training data*.*

### *The sigmoid function*, also known as the logistic function, is an activation function that maps input values to a range between 0 and 1. It is used in logistic regression to convert the linear combination of input features into a probability value. The sigmoid function is defined as σ(x) = 1 / (1 + exp(-x)). *MSE* is a cost function that measures the average squared difference between the predicted values and the actual values. It is commonly used in linear regression and other regression problems. The goal is to minimize the MSE to find the best-fitting model parameters. However, it is not used in logistic regression, as it can lead to non-convex optimization surfaces, making gradient descent less effective*. Cross Entropy* is the cost function used in logistic regression. It measures the difference between the predicted probabilities and the actual class labels. Cross-entropy is defined as: Cross Entropy Loss = - (y \* log(p) + (1 - y) \* log(1 - p))where y is the actual class label (0 or 1), and p is the predicted probability. The goal is to minimize the cross-entropy loss to find the best-fitting model parameters. Cross-entropy loss is convex, making it suitable for optimization using gradient descent. Gradient Descent: Gradient Descent is an optimization algorithm used to minimize a function iteratively. It is commonly used in machine learning to find the best-fitting model parameters that minimize the cost function. In each iteration, the algorithm updates the model parameters by taking a step proportional to the negative of the gradient (derivative) of the cost function with respect to the parameters. The learning rate determines the size of the steps taken. Reinforcement Learning (RL) is a subfield of machine learning where an agent learns to make decisions by interacting with an environment. The learning process is driven by a system of rewards and penalties. The goal of the agent is to maximize the cumulative reward over time by discovering the best sequence of actions in different situations.**In reinforcement learning, the agent learns a policy that maps from its current state to an action. The agent receives feedback in the form of rewards or penalties from the environment after taking an action. The learning process involves updating the policy based on this feedback to improve the agent's decision-making abilities.** A Markov Decision Process (MDP) is a mathematical framework used to model decision-making problems in reinforcement learning. MDPs consist of four main components: **states**, **actions**, **transitions**, and **rewards**. The key idea behind MDP is that the future depends only on the current state and the chosen action, not on the history of past states or actions. **This property is called the Markov property.** States (S): States represent the different situations or configurations that the agent can find itself in. For example, in a robot navigation problem, the states could be the different locations of the robot within the maze. Actions (A): Actions are the choices the agent can make in each state. In the robot navigation example, the actions could be moving up, down, left, or right. Transitions (P): Transitions describe the probability of reaching a new state (s') given the current state (s) and the chosen action (a). In other words, it defines how the environment changes in response to the agent's actions. In the robot navigation problem, the transitions could be deterministic, meaning that taking an action always leads to the same next state, or they could be stochastic, reflecting uncertainty in the robot's movements. Rewards (R): Rewards are the immediate feedback that the agent receives from the environment after taking an action in a given state. The agent's goal is to maximize the cumulative reward over time. In the robot navigation example, the agent could receive a positive reward for reaching the goal, a negative reward for hitting a wall, and a small negative reward for each step taken to encourage reaching the goal quickly. The Bellman Equation is a fundamental concept in reinforcement learning that describes the relationship between the value of a state and the value of its successor states, taking into account the rewards and actions.In simple words, it expresses the idea that the value of a state is the sum of the immediate reward and the discounted value of the next state, considering the best possible action.The Bellman Equation helps to find the optimal value function, which estimates the expected cumulative reward for each state, and ultimately leads to the optimal policy (the best sequence of actions to take in each state).

Strength and Weakness of Naïve Bayes Strength: • It is easy and fast.• It performs well in multiclass predictions.• When assumption of independence holds, a Naive Bayes classifier performs better compared to other models like logistic regression and you need less training data.• It performs well in case of categorical input variables compared to numerical variable(s). For numerical variables, normal distribution is assumed. Weakness:• If the categorical variable has a category (in test data set) which was not observed in training data set, then the model will assign a zero probability and will be unable to make a prediction. This is often known as “Zero Frequency”. To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.• Another limitation of Naive Bayes is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors which are completely independent.

Strength and Weakness of KNN Strength• Robust to noisy training data.• Effective and accurate if the training data size is large.Weakness• It is always difficult to determine which is the most optimal metric for measuring the distance and which attributes to use to produce the best results. Therefore, KNN is suitable for applications with which sufficient domain knowledge is available. This knowledge supports the selection of an appropriate measure.• The computation cost is high because we need to compute the distance of the query instance from all the given sample instances.