Core principles of Supervised Learning Algorithms

Supervised learning algorithms are methods used to make predictions or classifications by learning the relationship between input data and corresponding labels.

In supervised learning, the model receives training data with labels and learns patterns from this data to predict or classify new, unlabeled data.

1，Linear Regression

Linear regression is a supervised learning algorithm used to establish and analyze linear relationships between variables. It is primarily employed to address regression problems, predicting the relationship between one or more continuous numerical outputs (dependent variables) and one or multiple input features (independent variables).

Basic Principle

Linear regression is based on a simple assumption that there exists a linear relationship between the dependent variable (output) and the independent variable(s) (input). This implies that we assume the output can be predicted by a linear combination of inputs, where each input feature is multiplied by a weight, these products are summed, and then added with an intercept (constant term).

Linear regression is a simple yet powerful tool commonly used for data analysis, prediction, and modeling tasks, assuming a linear relationship between input features and the output. If the relationship is more complex, considering other types of regression models might be necessary.

2，Logistic Regression

Logistic regression is a supervised learning algorithm used to solve binary classification problems. Its basic principle involves modeling the probability relationship between the dependent variable (output) and independent variables (inputs) using the logistic function (also known as the Sigmoid function). The goal of logistic regression is to estimate the probability of an event, often represented as 0 or 1, for instance, whether a tumor is malignant (1) or benign (0).

Basic Principle

Logistic regression is based on the concept that we want to map the output of a linear combination to a probability value between 0 and 1, indicating the likelihood of an event occurring. To achieve this, logistic regression uses the logistic function (Sigmoid function) to perform this mapping.

Logistic regression is widely utilized in various fields, including medical diagnosis, financial risk assessment, natural language processing, image classification, and is particularly well-suited for binary classification problems.

3，Decision Tree

A decision tree is a commonly used supervised learning algorithm for solving classification and regression problems. Its basic principle involves constructing a tree-like structure based on the features of the data, where each node of the tree represents a feature, each branch represents a feature's value, and leaf nodes represent output classes or values. The goal of a decision tree is to split the dataset into purer subsets by splitting features to minimize error or impurity.

Basic Principle

The fundamental principle of a decision tree involves making decisions based on a series of rules. These rules are determined by splitting the input features, dividing the dataset into more homogeneous subsets with each split. The construction process of a decision tree typically involves the following steps:

1. Select a feature for splitting, usually determined based on a certain measurement criterion (such as information gain, Gini impurity, etc.) to ensure higher purity of subsets after the split.
2. Divide the dataset into subsets based on the selected feature and splitting point.
3. Recursively apply steps 1 and 2 until meeting stopping conditions (like reaching a predetermined tree depth, subset size falling below a threshold, etc.).
4. Assign an output label (for classification problems) or value (for regression problems) to the leaf nodes, often representing the majority class or mean of samples within that leaf node.

Decision trees do not involve specific mathematical formulas like linear or logistic regression; instead, they make decisions based on the structure of the tree. However, there are evaluation criteria used for feature splitting, with the most common being Gini impurity and information gain. These standards aid in selecting the best splitting points to construct better decision trees.

Gini Impurity: Used for classification problems, it indicates the probability of a sample being misclassified if chosen randomly from the dataset. Lower Gini impurity indicates higher purity in the dataset.

Information Gain: Also used for classification problems, it measures the reduction in entropy (disorder) after a split on a particular feature, representing the decrease in uncertainty within the dataset. Higher information gain signifies a purer dataset after splitting based on the feature.

The training process of a decision tree aims to maximize information gain or minimize Gini impurity to select the best splitting points. Once the decision tree is constructed, it can be used for classification or regression predictions. Decision trees are easy to understand and visualize, but in some cases, they might overfit the data. Hence, techniques such as pruning are often required to enhance their generalization performance.

4, **Support Vector Machine (SVM)** is a powerful supervised learning algorithm primarily used for classification, but it can also be applied to regression and anomaly detection problems. The basic principle of SVM is to find an optimal hyperplane in the feature space to maximize the margin between different classes, thereby enhancing classification accuracy.

Basic Principles:

Margin Maximization: The objective of SVM is to locate a hyperplane that maximizes the distance (margin) between sample points of different classes. This maximization of distance is termed "margin maximization."

Support Vectors: In SVM, only a small subset of sample points significantly influences the position of the hyperplane; these are termed "support vectors." These support vectors are the sample points closest to the hyperplane, and they determine the position of the hyperplane.

Kernel Trick: SVM utilizes kernel functions that map data from the original feature space to a higher-dimensional feature space. This enables SVM to handle non-linearly separable data by creating non-linear decision boundaries.

The core of SVM lies in finding the solution to this optimization problem, thereby determining the best hyperplane for classification tasks. Different kernel functions can be used to handle various types of data, such as linear, polynomial, and Gaussian kernels.

Kernel functions enable SVM to address nonlinear problems by mapping data to higher-dimensional spaces, making them linearly separable in that space. By adjusting hyperparameters and selecting suitable kernel functions, SVM can perform exceptionally well in various classification and regression problems.

5，K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a commonly used supervised learning algorithm primarily used for classification and regression problems. The fundamental principle of KNN is based on predicting or classifying using the distances between sample points in the feature space.

For classification problems, KNN identifies the K nearest training samples in the feature space to the sample awaiting classification and makes a decision based on a majority vote of their class labels.

For regression problems, KNN identifies the K nearest training samples and computes their average or weighted average to predict the numerical output for the sample awaiting prediction.

Basic Principles:

Distance Measurement: KNN relies on the measurement of similarity between sample points based on their distances. Typically, metrics such as Euclidean distance, Manhattan distance, Minkowski distance, etc., are used to calculate these distances.

Choice of K: The 'K' in KNN represents the number of nearest neighbors selected. By choosing different values of K, the model's complexity can be adjusted. A smaller K value may result in the model being sensitive to noise, while a larger K value might lead to an overly smoothed model.

Voting or Averaging: For classification problems, KNN performs a vote among the class labels of the K nearest training samples and assigns the most voted label to the sample awaiting classification. For regression problems, KNN computes the average or weighted average of the numerical outputs from the K nearest training samples and uses this as the output for the sample awaiting prediction.

K Nearest Neighbors' Selection: For classification problems, KNN chooses the K nearest training samples to the sample awaiting classification based on their distances, then makes a decision through a voting process among their class labels. For regression problems, KNN selects the K nearest training samples to the sample awaiting prediction and calculates their numerical output's average or weighted average to predict.

KNN is a simple and intuitive algorithm that doesn't require a training process, but it might become computationally intensive when dealing with large-scale datasets. Selecting the appropriate distance measurement and K value are crucial aspects of KNN, often requiring adjustment and optimization based on the specific problem.

Furthermore, KNN might perform poorly when handling imbalanced or high-dimensional data, hence careful consideration is necessary when selecting its application scenarios.