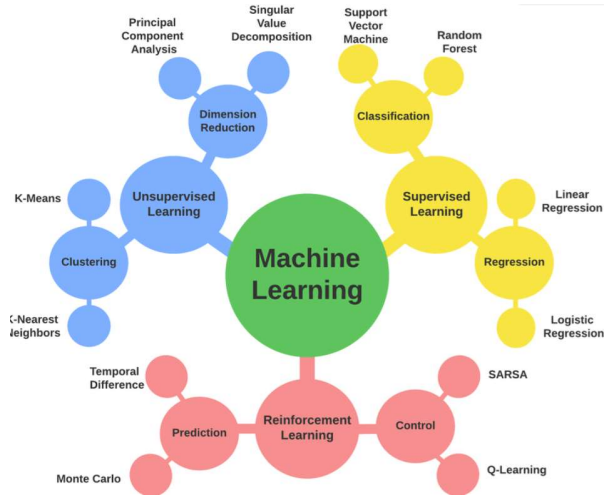
Các phương pháp học máy
Machine learning methods

4 TC: 2 LT – 2 TH

Giảng viên: **Tạ Hoàng Thắng**

tahoangthang@gmail.com

0975399307



Decision Tree

Definitions

Wikipedia: A decision tree is a **decision support recursive partitioning structure** that uses a tree-like model of decisions and their possible consequences

- chance event outcomes
- resource costs
- utility

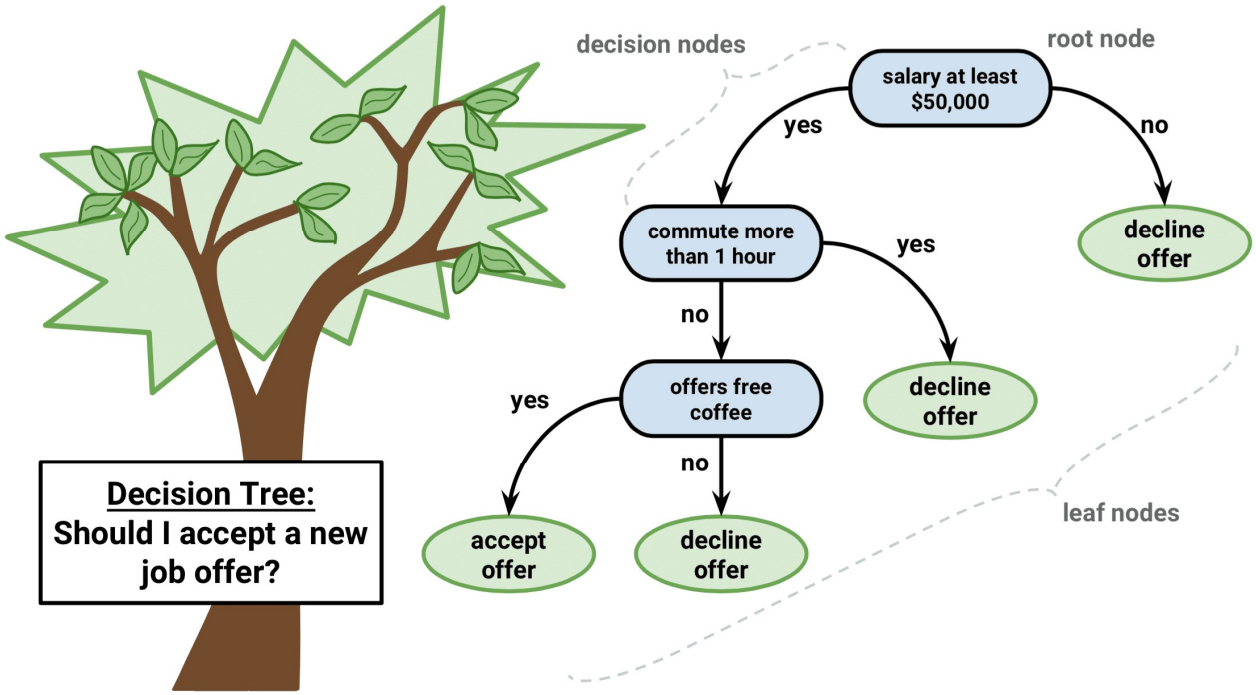
It is one way to display an algorithm that only contains conditional control statements.

Decision Tree

Definitions

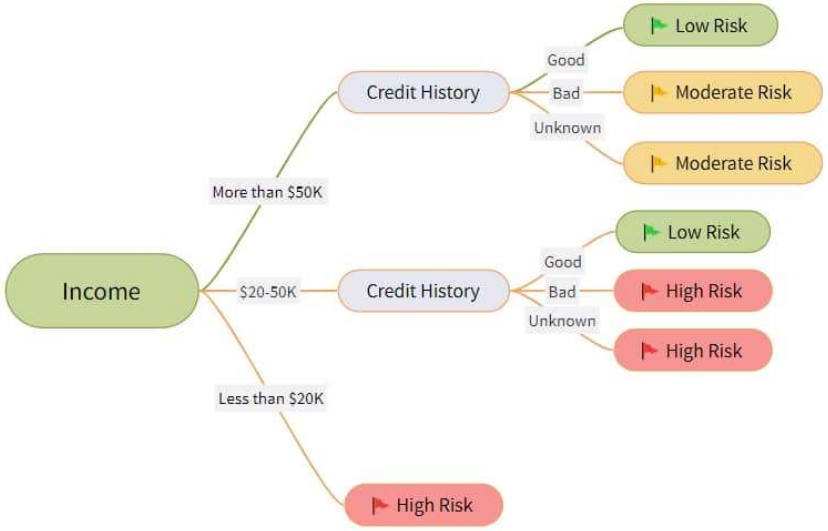
Wikipedia: Decision tree learning is a **supervised learning approach** used in statistics, data mining and machine learning.

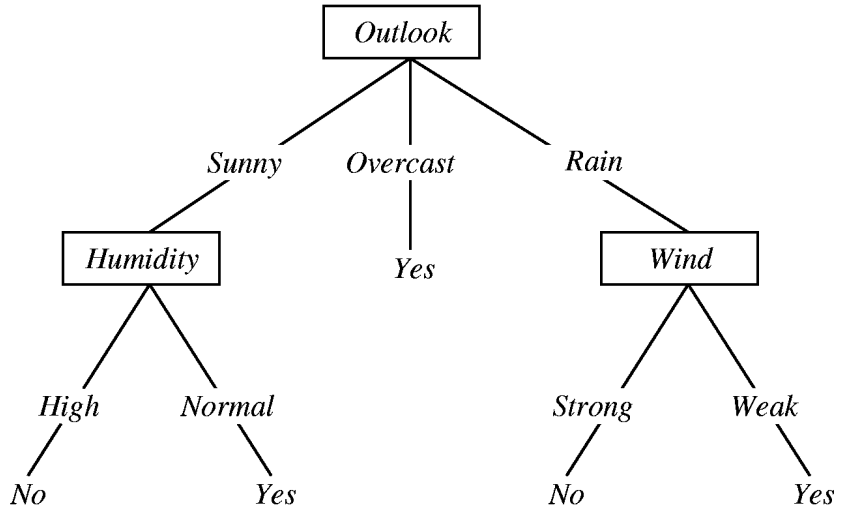
- In this formalism, a classification or regression decision tree is used as a predictive model to draw conclusions about a set of observations.



Decision Tree

Examples





Decision Tree

References

- https://scikit-learn.org/1.5/modules/tree.html
- https://machinelearningcoban.com/tabml_book/ch_model/decision_tree.html
- https://www.geeksforgeeks.org/decision-tree-introduction-example/

Decision Tree

Geeksforgeeks

https://www.geeksforgeeks.org/decision-tree-introduction-example/

Decision tree can be used in **regression and classification** problems.

Decision Tree

Decision Tree Terminologies

Root Node: A decision tree's root node, which represents the original choice or feature from which the tree branches, is the highest node.

Internal Nodes (Decision Nodes): Nodes in the tree whose choices are determined by the values of particular attributes. There are branches on these nodes that go to other nodes.

Leaf Nodes (Terminal Nodes): The branches' termini, when choices or forecasts are decided upon. There are **no more branches on leaf nodes**.

Branches (Edges): **Links between node**s that show how decisions are made in response to particular circumstances.

Decision Tree

Decision Tree Terminologies

Splitting: The process of dividing a node into two or more sub-nodes based on a decision criterion. It involves selecting a feature and a threshold to create subsets of data.

Parent Node: A node that is split into child nodes. The original node from which a split originates.

Child Node: Nodes created as a result of a split from a parent node.

Decision Criterion: The rule or condition used to determine how the data should be split at a decision node. It involves comparing feature values against a threshold.

Pruning: The process of **removing branches or nodes from a decision tree to improve its generalisation and prevent overfitting**.

Decision Tree

How Decision Tree is formed?

Forming a decision tree involves recursively partitioning data by **selecting the best attribute for splitting at each node**, based on criteria like information gain or **Gini impurity**.

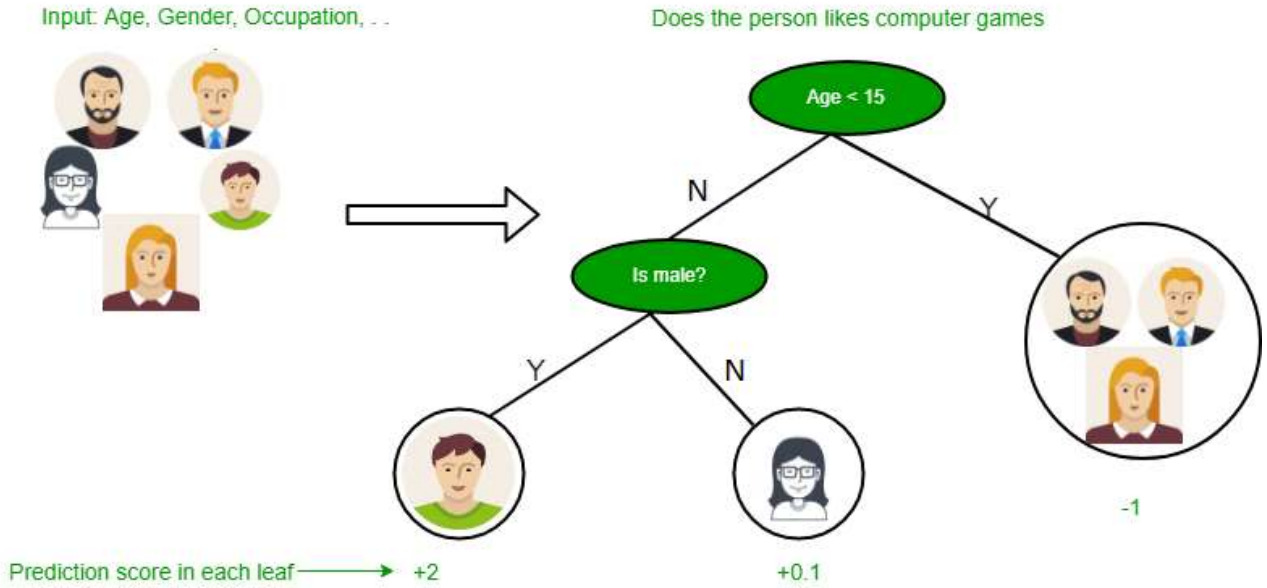
The process continues until a stopping criterion, such as maximum depth or minimum instances in a leaf, is reached.

Decision Tree

Decision Tree Approach

Decision tree uses the **tree representation** to solve the problem in which each leaf node corresponds to a class label and attributes are represented on the internal node of the tree.

- We can represent any boolean function on discrete attributes using the decision tree.



Decision Tree

Decision Tree Approach

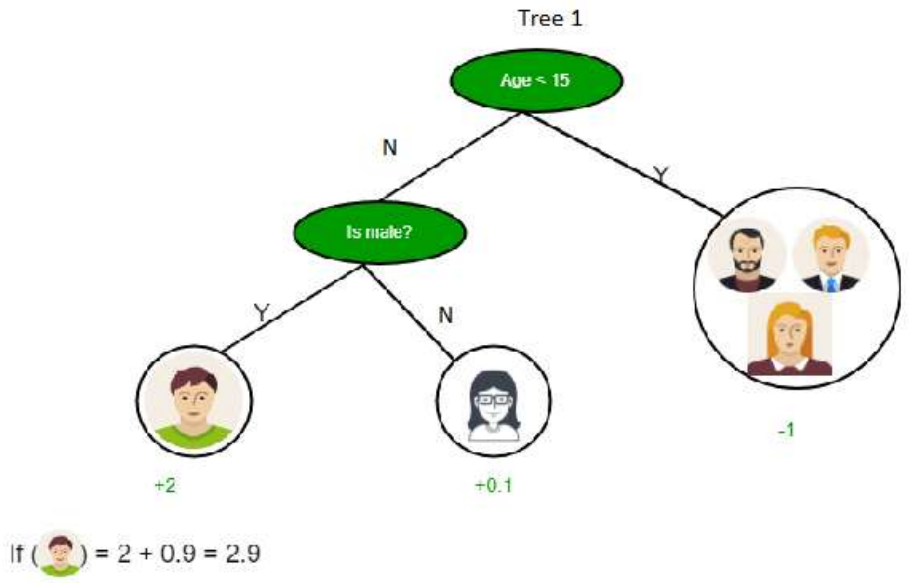
Below are some assumptions while using the decision tree:

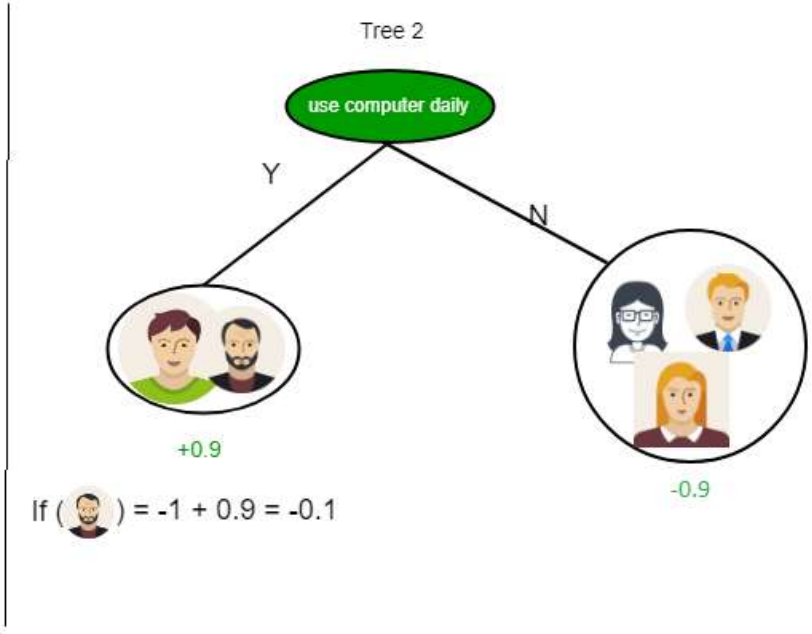
- At the beginning, **consider the whole training set as the root**.
- **Feature values are preferred** to be categorical.
  - If the values are continuous then **they are discretized prior** to building the model.
- On the basis of attribute values, records are distributed recursively.
- **Use statistical methods for ordering attributes as root** or the internal node.

Decision Tree

Decision Tree Approach

Decision Tree works on the Sum of Product form which is also known as **Disjunctive Normal Form**. Predict the use of computer in the daily life of people.





Decision Tree

Decision Tree Approach

In the Decision Tree, the major challenge is the identification of the attribute for the root node at each level. This process is known as attribute selection. There are two popular attribute selection measures:

- **Information Gain**
- **Gini Index**

Decision Tree

Information Gain

When we use a node in a decision tree to partition the training instances into smaller subsets **the entropy changes**.

Information gain is a measure of this change in entropy.

- Suppose S is a set of instances,
- A is an attribute
- S_v is the subset of S
- *v* represents an individual value that the attribute *A* can take and Values (A) is the set of all possible values of A, then

$$Gain(S, A) = Entropy(S) - \sum_v^A \frac{|S_v|}{|S|} . Entropy(S_v)$$

The goal is to find the attribute A that provides the highest information gain, as this attribute is the most effective at classifying the data, and thus is chosen as the splitting criterion in the decision tree.

Decision Tree

Information Gain

Information gain is a measure of this change in entropy.

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in A} \frac{|S_v|}{|S|} \cdot \text{Entropy}(S_v)$$

Explanation of Each Term

1. **Entropy(S)**: This is the entropy of the dataset S, which represents the disorder or impurity in the dataset before any split. The higher the entropy, the more mixed the classes are in S. Entropy for a dataset with classes can be calculated as:

$$\text{Entropy}(S) = -\sum_{i} p_i \log_2(p_i)$$

   where p_i is the probability of class i in S.

2. S_v: This is the subset of S where the attribute A has the value v. For each possible value v of attribute A, S_v represents all instances in S where $A = v$.
3. $\frac{|S_v|}{|S|}$: This term represents the fraction of the dataset S that falls into subset S_v.
4. **Entropy(S_v)**: This is the entropy of the subset S_v, representing the disorder within this subset after splitting on A.

Decision Tree

Information Gain

Entropy: is **the measure of uncertainty of a random variable**, it characterizes the impurity of an arbitrary collection of examples.

- The higher the entropy more the information content.

Suppose S is a set of instances, A is an attribute, S_v is the subset of S with A = v, and Values (A) is the set of all possible values of A, then

$$Gain(S, A) = Entropy(S) - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|} . Entropy(S_v)$$

Decision Tree

Information Gain

Entropy in the context of decision trees is a **measure of the uncertainty or "disorder" within a set of data**.

- Indicate how mixed the classes are.
- a higher entropy value means more disorder, while lower entropy implies the data is more "pure" (mostly one class).

Entropy Formula

For a dataset with two classes, entropy E is calculated as:

$$E = -p_1 \log_2(p_1) - p_2 \log_2(p_2)$$

where:

- p_1 is the proportion of the first class.
- p_2 is the proportion of the second class.

If the data has only one class (100% pure), entropy is 0, as there is no uncertainty. If it's an equal mix of classes (50/50), entropy is at its maximum, which is 1 in the binary case.

Decision Tree

Information Gain

Entropy in the context of decision trees is a **measure of the uncertainty or "disorder" within a set of data**.

Suppose we have a dataset of 10 fruits, where:

- 7 are apples
- 3 are oranges

The probability of picking an apple (p_{apple}) is $\frac{7}{10} = 0.7$. The probability of picking an orange (p_{orange}) is $\frac{3}{10} = 0.3$.

Now we calculate entropy:

$$E = -(0.7 \cdot \log_2(0.7) + 0.3 \cdot \log_2(0.3))$$

Using approximate values:

$$E \approx -(0.7 \cdot -0.515 + 0.3 \cdot -1.737)$$

$$E \approx 0.360 + 0.521 = 0.881$$

This entropy value (0.881) indicates some disorder in the dataset but not complete randomness. If the dataset were 100% apples or 100% oranges, entropy would be 0, representing no uncertainty in classifying the data.

Decision Tree

Information Gain

Calculate the entropy H(X):

For the set **X** = {a,a,a,b,b,b,b,b,b}

Total instances: 9

Instances of **b**: 6

Instances of **a**: 3

$$E = -p_1 \log_2(p_1) - p_2 \log_2(p_2)$$

Decision Tree

Building Decision Tree using Information Gain

- Start with all training instances associated with the root node.
- Use info gain to choose which attribute to label each node with.
- Note: No root-to-leaf path should contain the same discrete attribute twice.
- Recursively construct each subtree on the subset of training instances that would be classified down that path in the tree.
- If all positive or all negative training instances remain, the label that node “yes” or “no” accordingly.
- If no attributes remain, label with a majority vote of training instances left at that node.
- If no instances remain, label with a majority vote of the parent’s training instances.

Decision Tree

Building Decision Tree using Information Gain

Example:

- Training set: 3 features and 2 classes

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Decision Tree

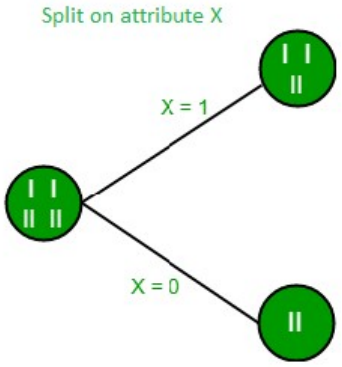
Building Decision Tree using Information Gain

Example:

- Take each of the features and calculate the information for each feature.

$$Gain(S, A) = Entropy(S) - \sum_v^A \frac{|S_v|}{|S|} . Entropy(S_v)$$

$$E = -p_1 \log_2(p_1) - p_2 \log_2(p_2)$$



$E_{parent} = 1$

GAIN = 1 – (3/4)(0.9184) – (1/4)(0) = 0.3112

$E_{child1} = -(1/3)\log_2(1/3) - (2/3)\log_2(2/3)$

= 0.5284 + 0.39

= 0.9184

$E_{child2} = 0$

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Decision Tree

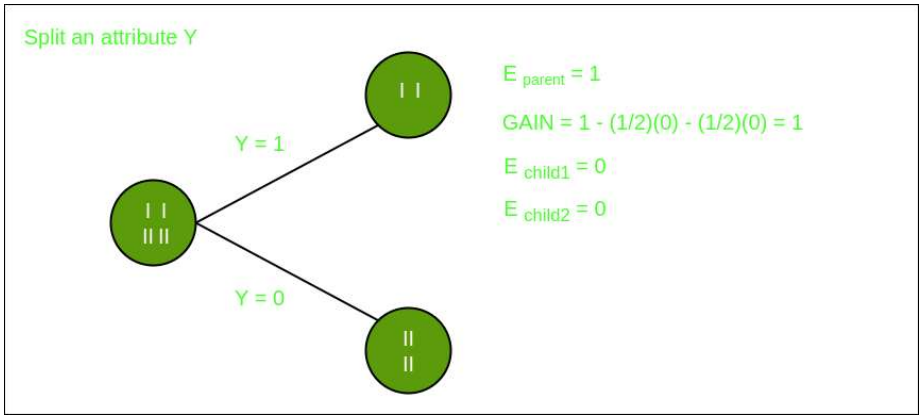
Building Decision Tree using Information Gain

Example:

- Take each of the features and calculate the information for each feature.

$$Gain(S, A) = Entropy(S) - \sum_v^A \frac{|S_v|}{|S|} . Entropy(S_v)$$

$$E = -p_1 \log_2(p_1) - p_2 \log_2(p_2)$$



X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Decision Tree

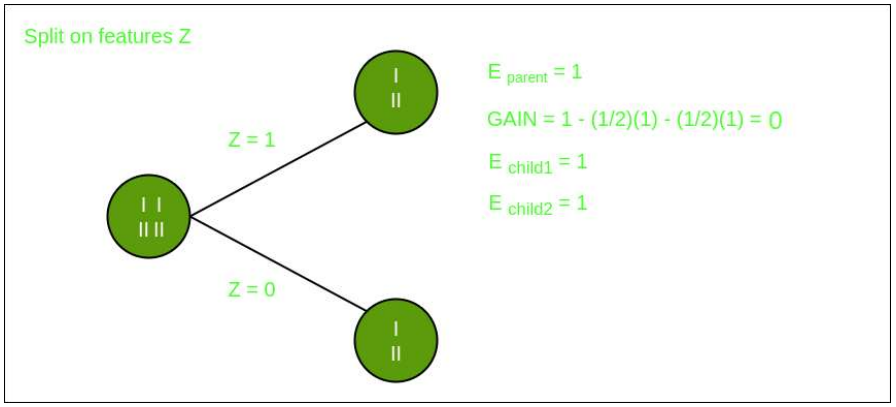
Building Decision Tree using Information Gain

Example:

- Take each of the features and calculate the information for each feature.

$$Gain(S, A) = Entropy(S) - \sum_v^A \frac{|S_v|}{|S|} . Entropy(S_v)$$

$$E = -p_1 \log_2(p_1) - p_2 \log_2(p_2)$$



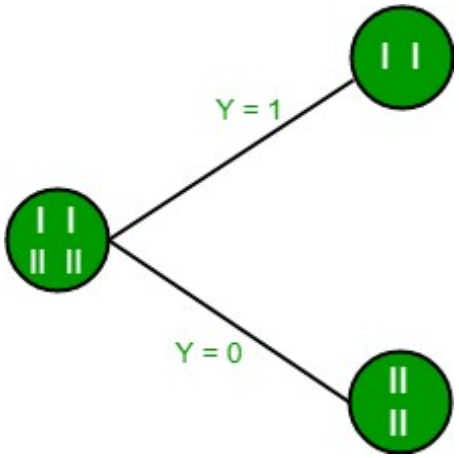
X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Decision Tree

Building Decision Tree using Information Gain

Example:

- The information gain is maximum when we make a split on feature **Y** (the root node best-suited feature).
  - While splitting the dataset by feature Y, the child contains a pure subset of the target variable. So we don't need to further split the dataset.



X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Decision Tree

Gini Index

Gini Index is a metric to measure **how often a randomly chosen element would be incorrectly identified**.

The Gini Index (or Gini Impurity) is a metric used to measure the "**impurity**" of a dataset.

The Gini Index measures how often a randomly chosen element from the dataset would be incorrectly labeled if it were randomly labeled according to the distribution of labels in the dataset.

- **It means an attribute with a lower Gini index should be preferred**.
- Sklearn supports “Gini” criteria for Gini Index and by default, it takes “gini” value.
- The Formula for the calculation of the Gini Index is given below.

Decision Tree

Gini Index

Gini Index is a metric to measure **how often a randomly chosen element would be incorrectly identified**.

For a dataset with two classes, the Gini Index G is calculated as:

$$G = 1 - (p_1^2 + p_2^2)$$

where:

- p_1 is the proportion of the first class,
- p_2 is the proportion of the second class.

If there are more than two classes, the formula generalizes to:

$$G = 1 - \sum_{i=1}^{n} p_i^2$$

where p_i is the proportion of class i and n is the total number of classes.

The Gini Index ranges from 0 to 0.5 for a binary classification:

- $G = 0$: Pure node (all instances belong to one class).
- $G = 0.5$: Maximum impurity (50/50 split in a binary case).

Decision Tree

Gini Index

Example:

Suppose we have a dataset of 10 fruits, where:

- 7 are apples
- 3 are oranges

1. The probability of picking an apple (p_{apple}) is $\frac{7}{10} = 0.7$.
2. The probability of picking an orange (p_{orange}) is $\frac{3}{10} = 0.3$.

Now we calculate the Gini Index:

$$G = 1 - (0.7^2 + 0.3^2)$$

$$G = 1 - (0.49 + 0.09)$$

$$G = 1 - 0.58 = 0.42$$

The Gini Index of 0.42 indicates that the dataset is somewhat impure (not all the data belongs to one class), but it's not entirely random either. A lower Gini Index means the node is "purer" (more instances of a single class).

If the dataset had only apples or only oranges, the Gini Index would be 0, showing no impurity.

Decision Tree

Gini Index

Some additional features and characteristics of the Gini Index are:

- A lower Gini Index indicates a more homogeneous or pure distribution, while a higher Gini Index indicates a more heterogeneous or impure distribution.
- Gini Index is used to evaluate the quality of a split by measuring the difference between the impurity of the parent node and the weighted impurity of the child nodes.
- Compared to other impurity measures like entropy, the **Gini Index is faster** to compute and more sensitive to changes in class probabilities.
- One disadvantage of the Gini Index is that it tends **to favour splits that create equally sized child nodes**, even if they are not optimal for classification accuracy.

Gradient Descent

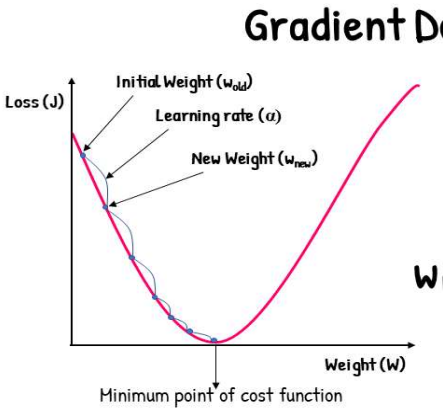
Definitions

ChatGPT: Gradient Descent is an optimization algorithm used to minimize a function by iteratively moving towards the function's minimum value.

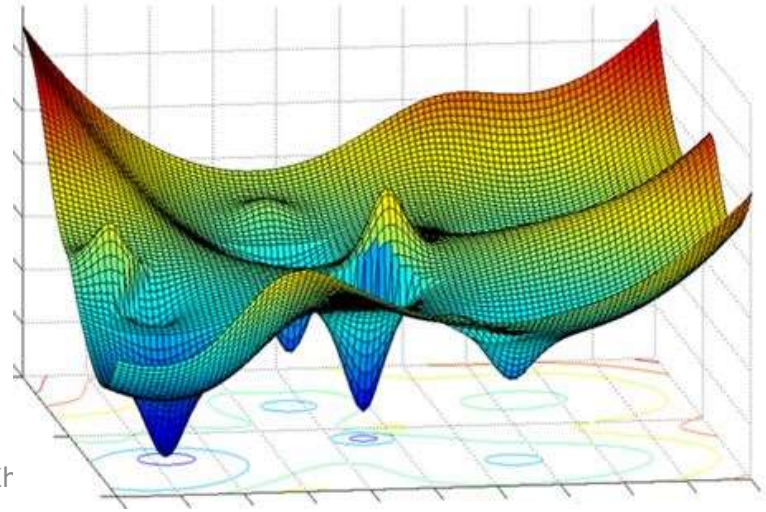
- widely used in machine learning and deep learning for optimizing models by reducing the loss (or error).

Wikipedia: Gradient descent is a method for unconstrained mathematical optimization.

- It is a first-order iterative algorithm for minimizing a differentiable multivariate function.



$$w_{new} = w_{old} - \alpha \frac{\delta J}{\delta w}$$



Gradient Descent

References

https://ndquy.github.io/posts/gradient-descent-2/

https://machinelearningcoban.com/2017/01/12/gradientdescent/

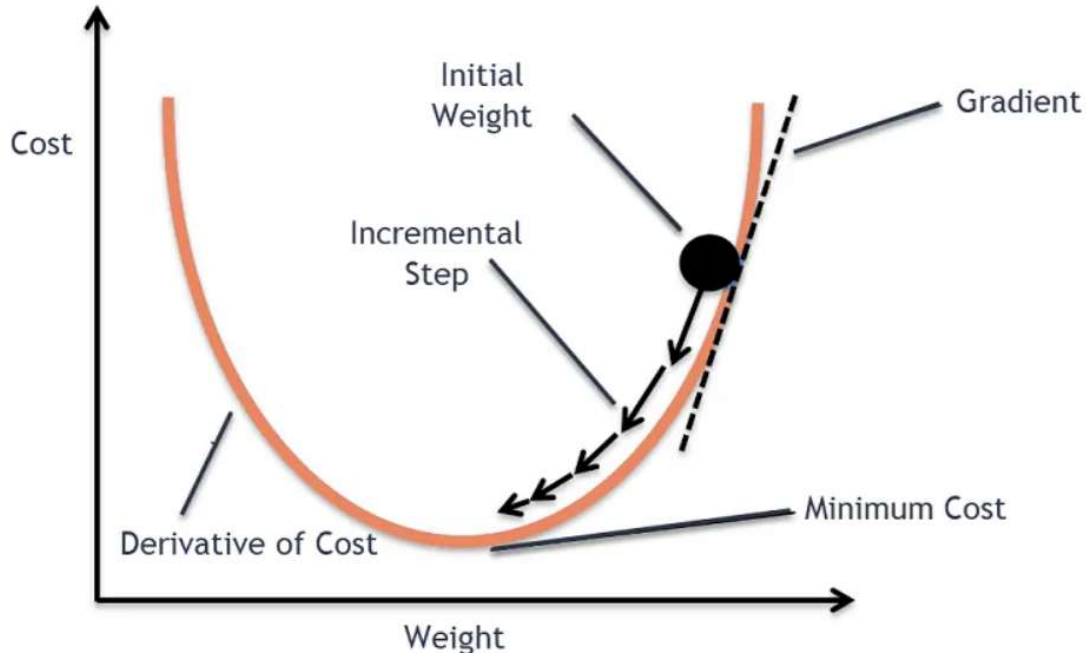
https://machinelearningcoban.com/2017/01/16/gradientdescent2/

Gradient Descent

Key Idea

The basic idea is to start with a random point on the function, compute the gradient (the slope) at that point, and then take a step in the opposite direction of the gradient.

- The process is repeated iteratively, gradually moving closer to the minimum of the function.



Gradient Descent

Steps

Steps of Gradient Descent

1. **Initialize** the parameters randomly (e.g., weights in a neural network).
2. **Compute the Gradient** of the loss function with respect to each parameter. The gradient tells us the direction and steepness of the slope.
3. **Update the Parameters**: Move the parameters slightly in the opposite direction of the gradient, using a small fraction known as the learning rate (α).

$$\theta = \theta - \alpha \cdot \nabla J(\theta)$$

where:

- θ is the parameter,
- α is the learning rate (controls the step size),
- $\nabla J(\theta)$ is the gradient of the loss function with respect to θ.

Gradient Descent

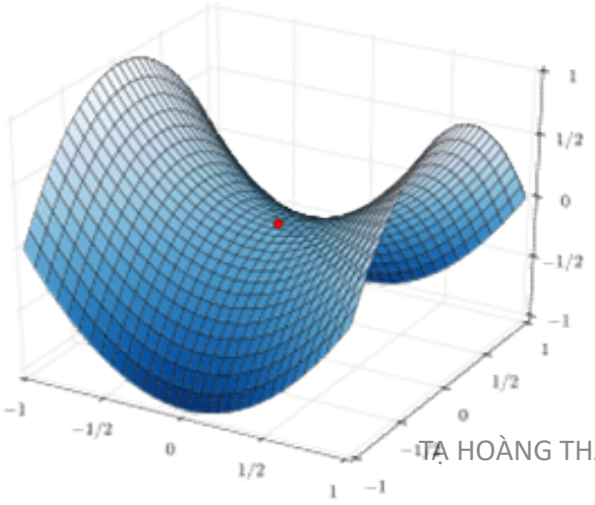
Types of Gradient Descent

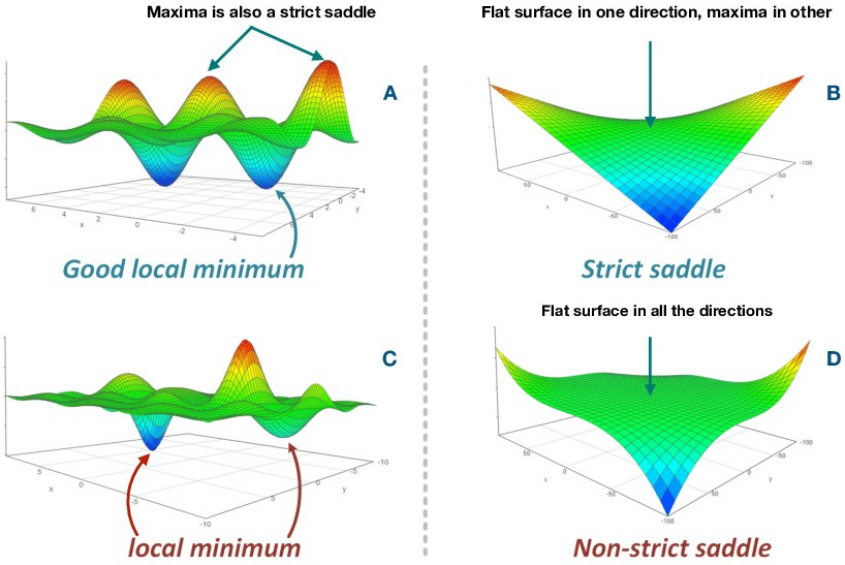
1. **Batch Gradient Descent**: Uses the entire dataset to compute the gradient. It can be slow for large datasets but is stable and guarantees convergence.
2. **Stochastic Gradient Descent (SGD)**: Uses one randomly chosen example at each step. It's faster and introduces noise, which can help escape local minima but may result in fluctuations around the minimum.
3. **Mini-batch Gradient Descent**: Combines the two by using a subset of the dataset (a mini-batch) to compute the gradient. It balances speed and stability and is commonly used in practice.

Gradient Descent

Challenges and Considerations

1. **Learning Rate**: If the learning rate is too high, gradient descent may overshoot the minimum; if too low, it may take a long time to converge.
2. **Local Minima**: Gradient descent may get stuck in local minima (suboptimal points), though this is less of a problem in high-dimensional spaces where saddle points are more common than true local minima.
3. **Vanishing/Exploding Gradients**: For deep networks, gradients can become very small or very large, making training challenging. Techniques like gradient clipping or using specific architectures can help mitigate this.





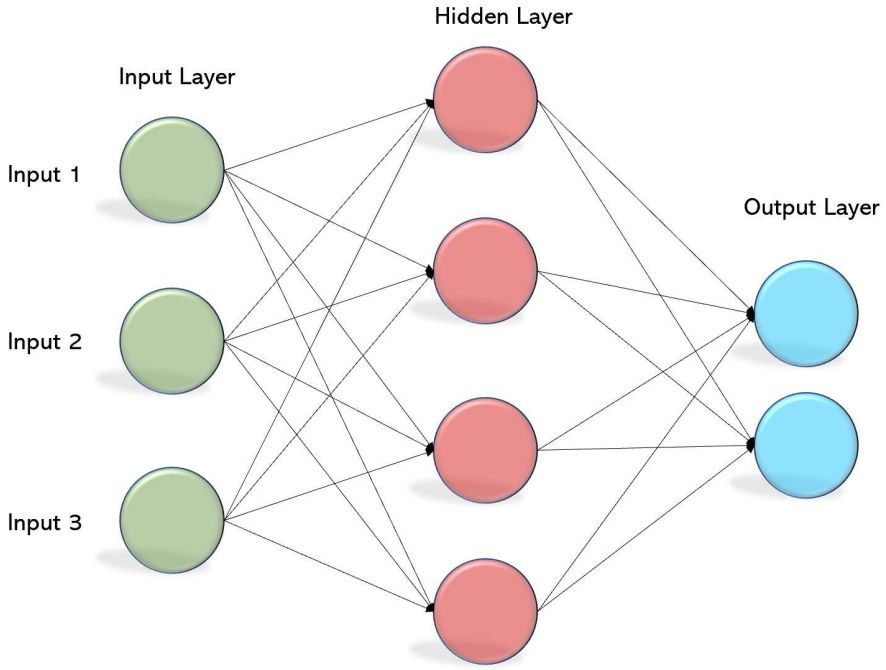
MLP

A multilayer perceptron (MLP) is a name for a modern feedforward artificial neural network,

- consisting of fully connected neurons with a nonlinear activation function,
- organized in at least three layers,
- notable for being able to distinguish data that is not linearly separable.

References

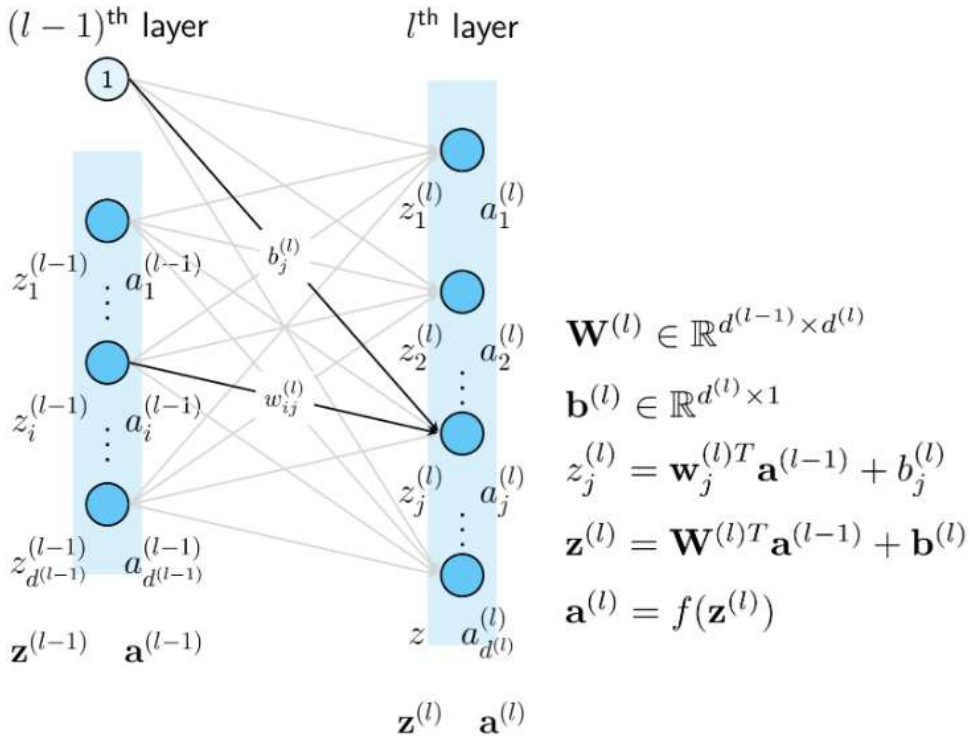
- https://machinelearningcoban.com/2017/02/24/mlp/
- https://www.datacamp.com/tutorial/multilayer-perceptrons-in-machine-learning



MLP

A multilayer perceptron (MLP) is a name for a modern feedforward artificial neural network,

- consisting of fully connected neurons with a nonlinear activation function,
- organized in at least three layers,
- notable for being able to distinguish data that is not linearly separable.



Hình 4: Các ký hiệu sử dụng trong MLP.

KAN = Kolmogorov-Arnold Networks

FC-KAN: Function Combinations in Kolmogorov-Arnold Networks

- https://arxiv.org/abs/2409.01763