Unit-6
Learning System

Introduction

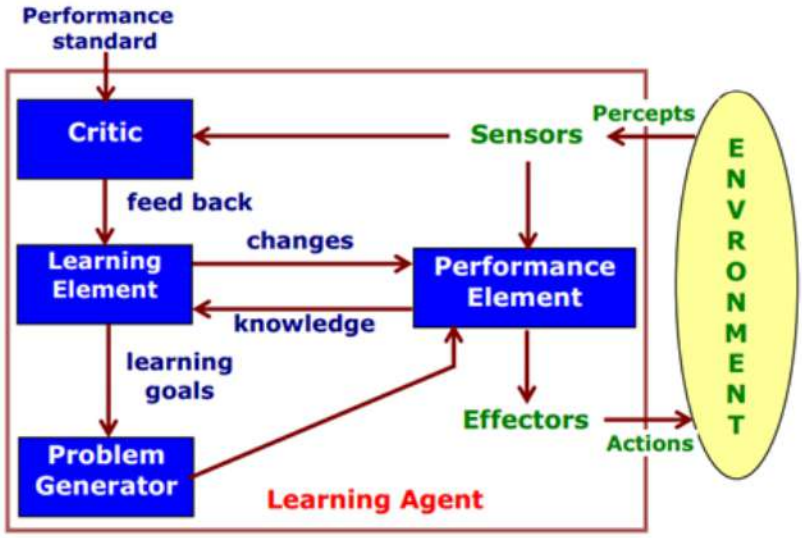
Learning refers to the process by which a system improves its performance on a specific task over time based on experience or data, without being explicitly programmed for every scenario.

A computer program is said to be learn from experience E with respect to some class of tasks T and performance measure P, if it's performance at tasks in T, as measured by P, improves with experience E.

Learning involves 3 factors:

- ***Changes:*** Learning changes the learner: for machine learning the problem is to determining the change and to represent them in efficient way.
- ***Generalization:*** Learning leads to generalization: performance must improve not only on the same task but on similar tasks.
- ***Improvement:*** Learning leads to improvements: machine learning must address the possibility that changes may degrade performance and find ways to prevent it.

Components of a Learning System

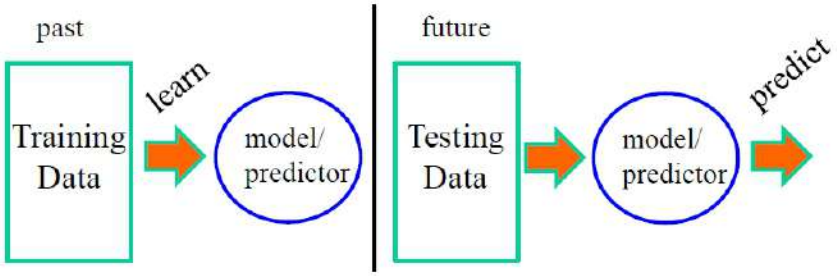


- ***Learning element:*** It is responsible for making improvements by learning from environment. It uses knowledge about the agent and feedback on its actions to improve performance.
- ***Critic (feedback element):*** Learning element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard.
- ***Performance element:*** It is responsible for selecting external actions according to the percepts it takes.
- ***Problem generator:*** This component is responsible for suggesting actions that will lead to new and informative experiences.

Machine Learning

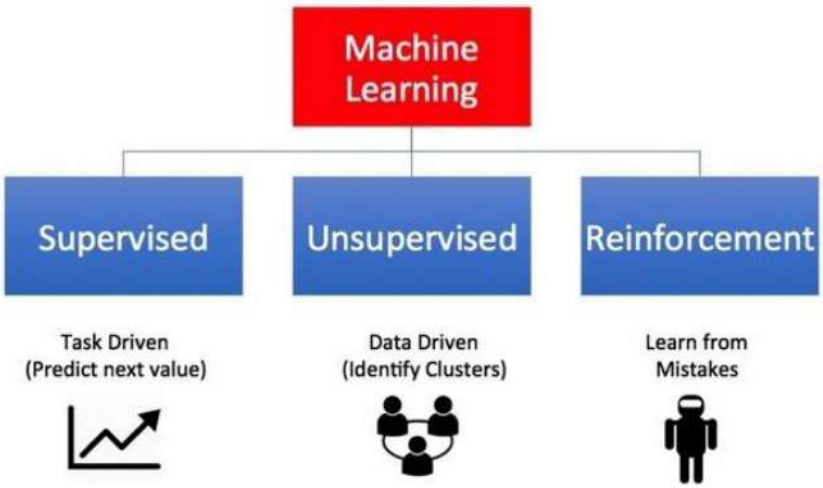
Machine learning is a sub field of Artificial Intelligence that provides computer the ability to learn and improve its learning from experience without being written rules explicitly.

Machine learning is about predicting the future based on the past. A machine learning system learns from historical data, builds the prediction model and whenever it receives new data, predicts the output for it.



- ***Training Phase:***
  - The system learns from training data to build a model/predictor.
  - This phase involves identifying patterns and relationships in the historical data.
- ***Testing and Prediction Phase:***
  - The trained model/predictor is tested on new data (testing data) to evaluate its performance.
  - The model is then used to make predictions on unseen future data.

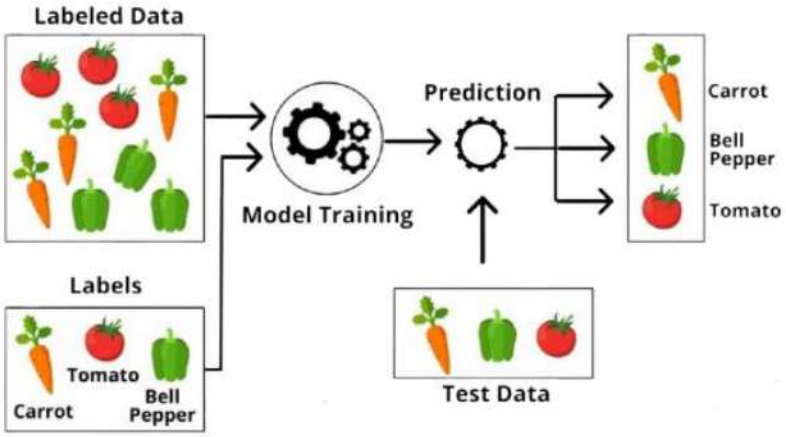
Types of Machine Learning



1. *Supervised Learning*

Supervised learning involves training a model on a labelled dataset, meaning that each training examples is paired with an output label. The model learns to make predictions or decisions based on the input-output pairs.

How it works: The algorithm tries to find patterns in the training data that map the input data (features) to the desired output (labels). It then uses this learned mapping to predict the output for new, unseen data.



Supervised learning is classified into:

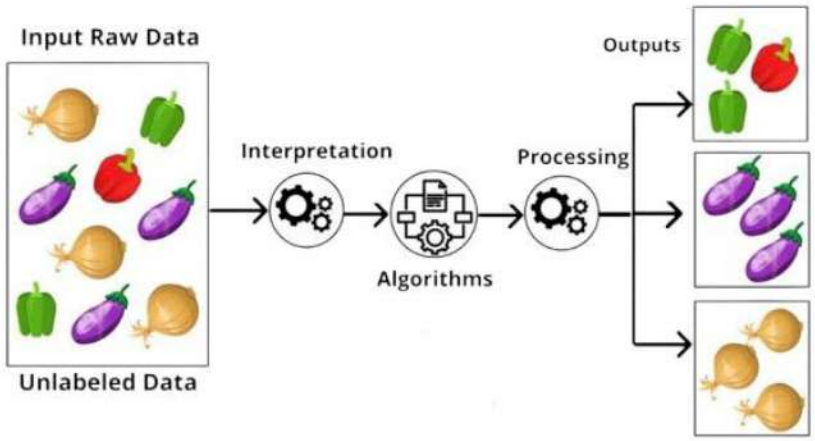
a) ***Classification:*** Classification is used to predict categorical values, such as whether an email is spam or not, or whether a medical image shows a tumor or not.

b) ***Regression:*** Regression is used to predict continuous values, such as house prices, or stock prices churn.

2. ***Unsupervised Learning***

Unsupervised learning involves training a model on data that does not have labelled responses. The model tries to find hidden patterns or intrinsic structures in the input data.

How it works: The algorithm analyzes the data to identify patterns, group similar data points together, or reduce the data's dimensionality. It does this without any specific guidance on what to look for.



Unsupervised learning is classified into:

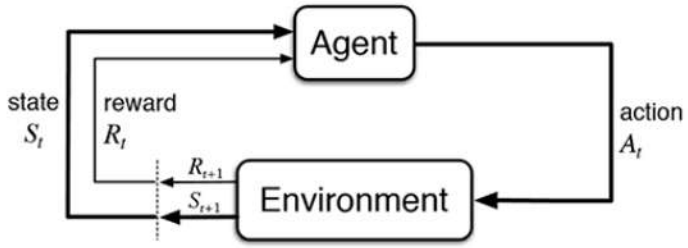
a) ***Clustering:*** Clustering is the process of grouping similar data points together based on their attributes or features.

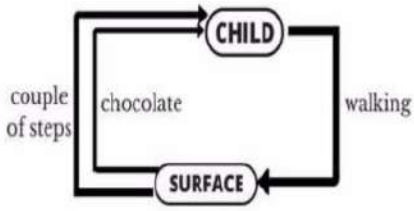
b) ***Association:*** The association is the process of identifying relationships or associations between items in a dataset.

c) ***Dimensionality reduction:*** Dimensionality reduction is the process of reducing the number of variables or features in a dataset while retaining the most important information.

3. ***Reinforcement Learning***

Reinforcement learning is a feedback based learning method, in which a learning agent gets a reward for each right action and gets penalty for each wrong action. The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment to learn how to perform a task. The goal of an agent is to get the most reward points, and hence, it improves its performance.





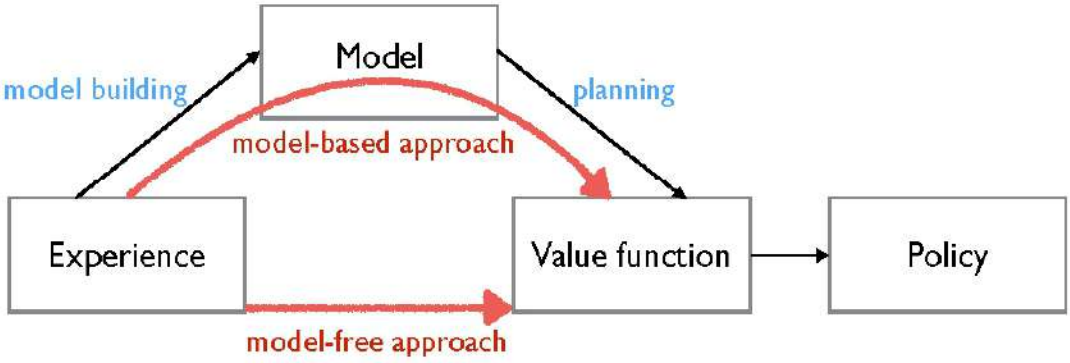
- The agent and environment interact in a sequence of discrete time steps, $t = 0, 1, 2, 3, \ldots$
- At each discrete time t, the agent (learning system) observes state $S_t \in S$ and choose action $A_t \in A$.
- Then agent receives an immediate numerical reward $R_{t+1} \in R$ and the state changes to S_{t+1}.
- At each time step, the agent implements a mapping from states to probabilities of selecting each possible action.
- The agent's goal, is to maximize the total amount of reward it receives over the long run.

Reinforcement learning is commonly used in applications such as *game playing*, *robotics*, and *autonomous driving*.

Types of Reinforcement Learning:

There are two types of reinforcement learning: *model-based* and *model-free*.

- ***Model-based***, as it sounds, has an agent trying to understand its environment and creating a model for it based on its interactions with this environment. In such a system, preferences take priority over the consequences of the actions i.e. the greedy agent will always try to perform an action that will get the maximum reward irrespective of what that action may cause.
- ***Model-free algorithms*** seek to learn the consequences of their actions through experience via algorithms such as Policy Gradient, Q-Learning, etc. In other words, such an algorithm will carry out an action multiple times and will adjust the policy (the strategy behind its actions) for optimal rewards, based on the outcomes.



Rote Learning

Rote learning is a memorization technique based on repetition. It is also called memorization because the knowledge, without any modification is, simply copied into the knowledge base.

Rote learning technique avoids understanding the inner complexities but focuses on memorizing the material so that it can be recalled by the learner exactly the way it read or heard.

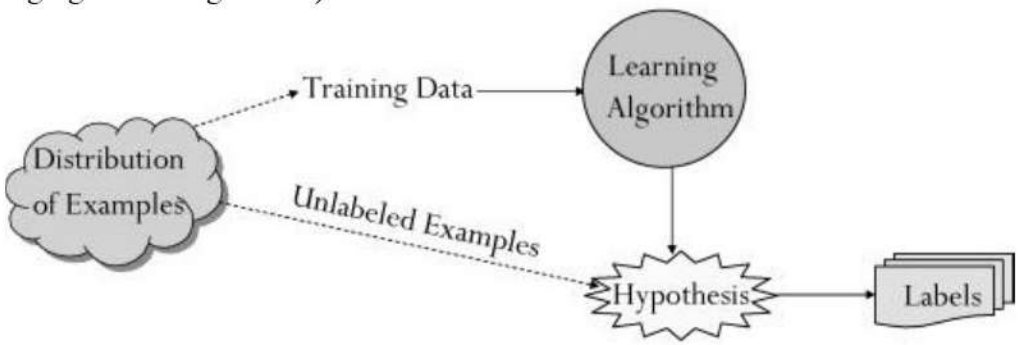
Example: Memorizing multiplication tables

Rote learning includes the capabilities:

- ***Organized storage of information:*** In order to improve the performance and speed up to use the stored value than it would be recomputed it. Then there must be a special technique that accesses the stored value quickly.
- ***Generalization:*** Here the number of distinct object that stores are very large. So that to keep the number of stored object manageable level some kind of generalization technique is necessary.

Learning From Examples: Inductive Learning

- Learning by example is a general learning strategy where a concept is learned by drawing inferences from a set of observed examples (facts).
- In inductive learning, a model requires a sufficient number of training examples to achieve a desired level of generalization accuracy.
- It generalizes from observed training examples by identifying features that separate positive examples (instances belonging to a target class) from negative examples (instances not belonging to the target class).



The figure represents the ***Inductive Learning Process***. A subset of labeled examples is taken from the distribution of examples and fed into a ***Learning Algorithm*** for training. The algorithm processes the training data to generate a ***Hypothesis*** (a model or function that generalizes patterns from the data). This hypothesis is then applied to unlabeled examples to predict their labels.

Three methods are used in inductive learning:

1. Winston's Learning Program
2. Version Spaces
3. Decision Trees

Decision Tree

In ***Decision tree learning***, a new example is classified by submitting it to a series of tests that determine the class label of the example. These tests are organized in a hierarchical structure called a ***decision tree***.

A ***decision tree*** is a flowchart-like tree structure, where each *internal node (non-leaf node)* denotes a test on an attribute, each *branch* represents an outcome of the test, and each *leaf node (or terminal node)* holds a class label. The top most node in a tree is the root node.

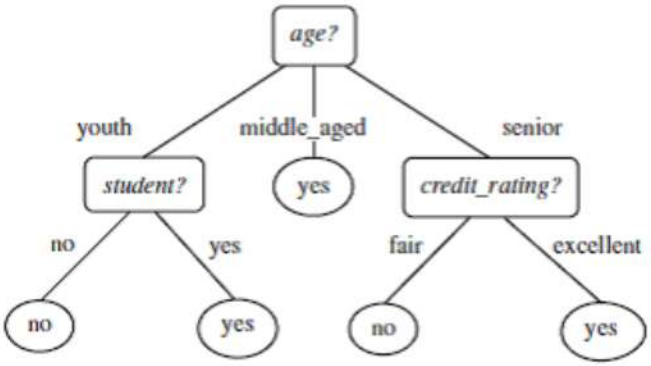


Fig: A decision tree for the concept *buys computer*

Class-label Yes: The customer is likely to buy a computer.

Class-label No: The customer is unlikely to buy a computer.

The most popular algorithm for constructing decision trees is ***ID3 (Iterative Dichotomiser 3)***. ID3 uses *information gain* as its attribute selection measure. The attribute with the highest information gain is chosen as the splitting attribute for node *N*.

Information gain is calculated as:

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

S refers to the entire set of examples that we have. A is the attribute we want to partition or split. $|S|$ is the number of examples and $|S_v|$ is the number of examples for the current value of attribute A.

Where,

$$Entropy(S) = -\sum_{i=1}^{m} p_i log_2(p_i)$$

Where p_i is the probability that an arbitrary tuple in S belongs to class C_i.

ID3 Algorithm

1. If all examples are +ve , return leaf node 'positive' and stop.
1. If all examples are –ve , return leaf node 'negative' and stop
2. Otherwise
   a) Calculate entropy to select root node and branch node.
   b) Partition examples into subset
   c) Repeat until all examples are classified.

Example

Day	Outlook	Temperature	Humidity	Wind	Play ball
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

From the total of 14 rows in our dataset **S**, there are 9 rows with the target value "Yes" and 5 rows with the target value "No". The entropy of **S** is calculated as:

$$Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$$

We now calculate the Information Gain for each attribute:

Attribute: Outlook

$$Entropy(S_{Sunny}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.971$$
$$Entropy(S_{Overcast}) = -\frac{4}{4}log_2\frac{4}{4} - \frac{0}{4}log_2\frac{0}{4} = 0$$
$$Entropy(S_{Rain}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.971$$

$$Gain(S, Outllok) = 0.94 - \left(\frac{5}{14}\right)(0.971) - \left(\frac{4}{14}\right)(0) - \left(\frac{5}{14}\right)(0.971) = 0.2464$$

Attribute: Temperature

$$Entropy(S_{Hot}) = -\frac{2}{4}log_2\frac{2}{4} - \frac{2}{4}log_2\frac{2}{4} = 1.0$$
$$Entropy(S_{Mild}) = -\frac{4}{6}log_2\frac{4}{6} - \frac{2}{6}log_2\frac{2}{6} = 0.9183$$
$$Entropy(S_{Cool}) = -\frac{3}{4}log_2\frac{3}{4} - \frac{1}{4}log_2\frac{1}{4} = 0.8113$$

$$Gain(S,\ Temperature) = 0.94 - \left(\frac{4}{14}\right)(1.0) - \left(\frac{6}{14}\right)(0.9183) - \left(\frac{4}{14}\right)(0.8113) = 0.0289$$

Attribute: Humidity

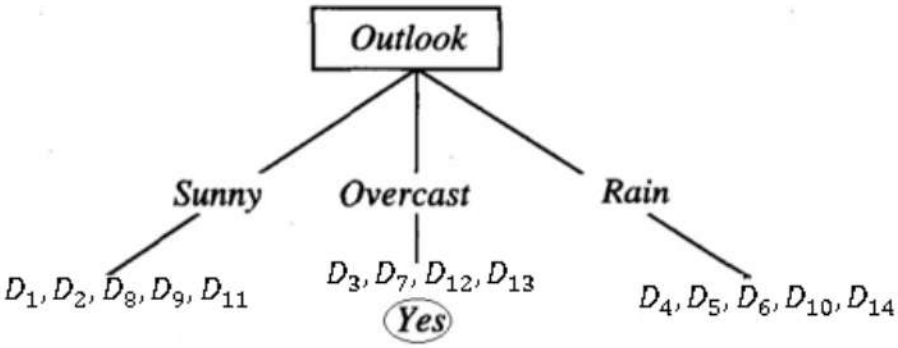
$$Entropy(S_{High}) = -\frac{3}{7}log_2\frac{3}{7} - \frac{4}{7}log_2\frac{4}{7} = 0.9852$$
$$Entropy(S_{Normal}) = -\frac{6}{7}log_2\frac{6}{7} - \frac{1}{7}log_2\frac{1}{7} = 0.5916$$

$$Gain(S, Humidity) = 0.94 - \left(\frac{7}{14}\right)(0.9852) - \left(\frac{7}{14}\right)(0.5916) = 0.1516$$

Attribute: Wind

$$Entropy(S_{Strong}) = -\frac{3}{6}log_2\frac{3}{6} - \frac{3}{6}log_2\frac{3}{6} = 1.0$$
$$Entropy(S_{Weak}) = -\frac{6}{8}log_2\frac{6}{8} - \frac{2}{8}log_2\frac{2}{8} = 0.8113$$
$$Gain(S, Wind) = 0.94 - \left(\frac{6}{14}\right)(1.0) - \left(\frac{8}{14}\right)(0.8113) = 0.0478$$

The attribute outlook has the ***highest information gain*** of 0.246, thus it is chosen as root.



Here, when Outlook == Overcast, it is of pure class (Yes). Now, we have to repeat same procedure for the data with rows consist of Outlook value as Sunny and then for Outlook value as Rain.

Now, finding the best attribute for splitting the data with Outlook=Sunny values { Dataset rows = [1, 2, 8, 9, 11]}.

Day	Temperature	Humidity	Wind	Play ball
D1	Hot	High	Weak	No
D2	Hot	High	Strong	No
D8	Mild	High	Weak	No
D9	Cool	Normal	Weak	Yes
D11	Mild	Normal	Strong	Yes

Complete Entropy of "Sunny" is:

$$Entropy(S_{Sunny}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.971$$

Calculating ***Information gain for attributes with respect to Sunny is:***

Attribute: Temperature

$$Entropy(S_{Hot}) = 0.0$$
$$Entropy(S_{Mild}) = 1.0$$
$$Entropy(S_{Cool}) = 0.0$$

$$Gain(S_{Sunny},\ Temperature) = 0.971 - \left(\frac{2}{5}\right)(0) - \left(\frac{2}{5}\right)(1.0) - \left(\frac{1}{5}\right)(0) = 0.571$$

Attribute: Humidity

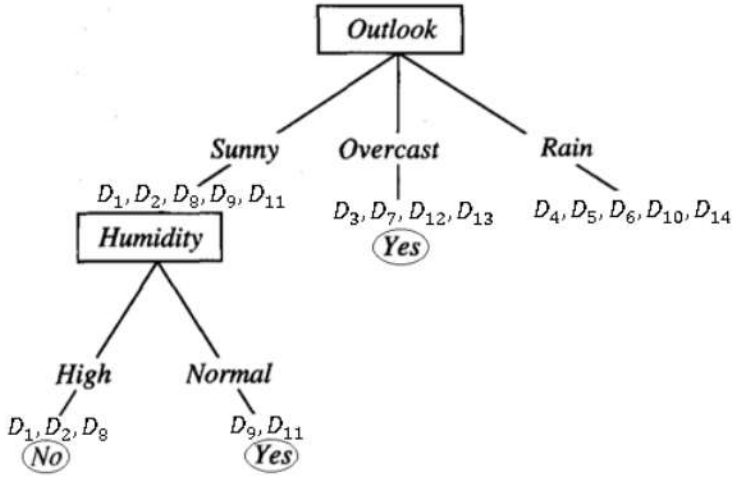
$$Entropy(S_{High}) = 0.0$$
$$Entropy(S_{Normal}) = 0.0$$

$$Gain(S_{Sunny}, Humidity) = 0.971 - \left(\frac{3}{5}\right)(0) - \left(\frac{2}{5}\right)(0) = 0.971$$

Attribute: Wind

$$Entropy(S_{Strong}) = 1.0$$
$$Entropy(S_{Weak}) = -\frac{1}{3}log_2\frac{1}{3} - \frac{2}{3}log_2\frac{2}{3} = 0.9183$$
$$Gain(S_{Sunny}, Wind) = 0.971 - \left(\frac{2}{5}\right)(1.0) - \left(\frac{3}{5}\right)(0.9183) = 0.02$$

The information gain for humidity is highest, therefore it is chosen as the next node.



Here, when Outlook = Sunny and Humidity = High, it is a pure class of category "No". And When Outlook = Sunny and Humidity = Normal, it is again a pure class of category "Yes". Therefore, we don't need to do further calculations.

Now, finding the best attribute for splitting the data with Outlook = Rain values{ Dataset rows = [4, 5, 6, 10, 14]}.

Day	Temperature	Humidity	Wind	Play ball
D4	Mild	High	Weak	Yes
D5	Cool	Normal	Weak	Yes
D6	Cool	Normal	Strong	No
D10	Mild	Normal	Weak	Yes
D14	Mild	High	Strong	No

Complete Entropy of Rain is:

$$Entropy(S_{Rain}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.971$$

Calculating **Information gain for attributes with respect to Rain is:**

Attribute: Temperature

$$Entropy(S_{Hot}) = 0.0$$
$$Entropy(S_{Mild}) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3} = 0.9183$$
$$Entropy(S_{Cool}) = 1.0$$

$$Gain(S_{Rain},\ Temperature) = 0.971 - \left(\frac{0}{5}\right)(0) - \left(\frac{3}{5}\right)(0.9183) - \left(\frac{2}{5}\right)(1.0) = 0.02$$

Attribute: Humidity

$$Entropy(S_{High}) = 1.0$$
$$Entropy(S_{Normal}) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3} = 0.9183$$

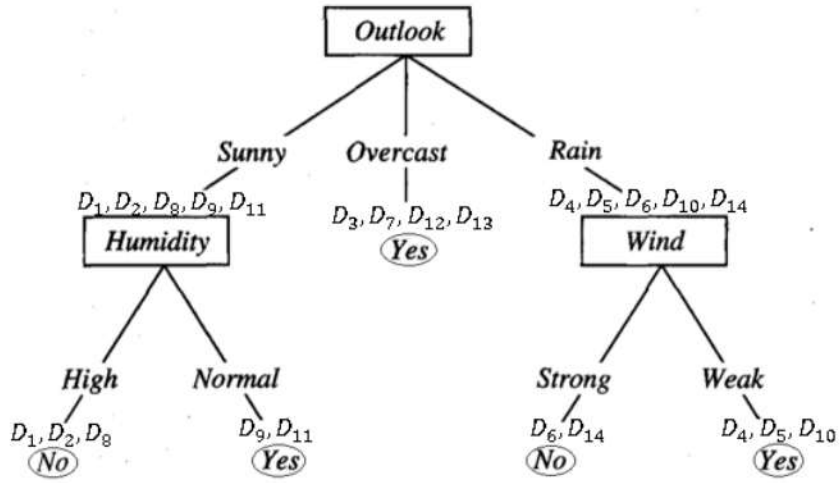
$$Gain(S_{Rain}, Humidity) = 0.971 - \left(\frac{2}{5}\right)(1.0) - \left(\frac{3}{5}\right)(0.9183) = 0.02$$

Attribute: Wind

$$Entropy(S_{Strong}) = 0.0$$
$$Entropy(S_{Weak}) = 0.0$$

$$Gain(S_{Rain}, Wind) = 0.971 - \left(\frac{2}{5}\right)(0) - \left(\frac{3}{5}\right)(0) = 0.971$$

Here, the attribute with maximum information gain is Wind. So, the decision tree built so far -



Here, when Outlook = Rain and Wind = Strong, it is a pure class of category "No". And When Outlook = Rain and Wind = Weak, it is again a pure class of category "Yes". And this is our final desired tree for the given dataset.

Explanation Based Learning

An explanation-based Learning (EBL) system accepts an example (i.e. a training example) and explains what it learns from the example. The EBL system takes only the relevant concepts of the training. This explanation is translated into particular form that a problem solving can understand. The explanation is generalized so that it can be used to solve other problem.

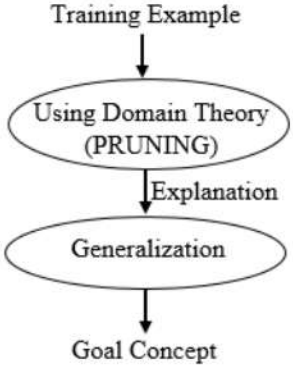
An EBL accepts 4 inputs:

a. ***A training Example:*** what the learning sees in the world.
b. ***A goal concept:*** a high level description of what the program is supposed to learn.
c. ***Operational criterion:*** a description of which concepts are usable.
d. ***A domain theory:*** a set of rules that describes relationships between object and action in a domain.

Given this four inputs, the task is to determine a generalization of the *training example* that is sufficient concept definition for the *goal concept* and that satisfies the *operationality criteria*.

EBL has two steps:

1. ***Explanation:*** In this step, the domain theory is used to prune away all unimportant aspects of the training example with respect to the goal concept.
2. ***Generalization:*** In this step, the explanation is generalized as far as possible while still describing the goal concept.



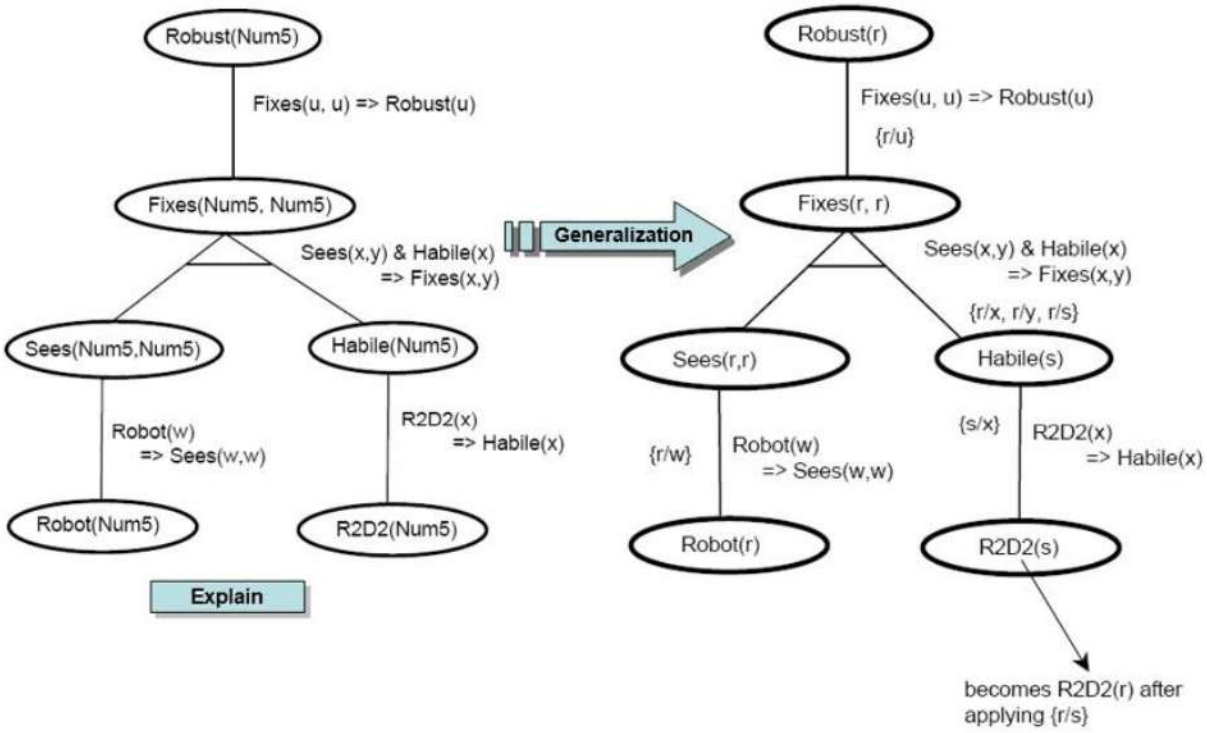
Example

Training Example:

$$Robot(Num5) \wedge R2D2(Num5) \wedge Age(Num5,5) \wedge \ldots\ldots \rightarrow Robust(Num5)$$

Domain Theory:

$Fixes(u,u) \rightarrow Robust(u)$ *// An individual that can fix itself is robust*
$Sees(x,y) \wedge Habile(x) \rightarrow Fixes(x,y)$ *// A habil individual that can see another entity can fix that entity*
$Robot(w) \rightarrow Sees(w,w)$ *// All robots can see themselves*
$R2D2(x) \rightarrow Habile(x)$ *// R2D2-class in individuals are habil*
… … …



Resulting Rule:

$$Robot(r) \wedge R2D2(r) \rightarrow Robust(r)$$