Machine Learning
UNIT 4

PRASHANT TOMAR

INTRODUCTION TO LEARNING

Machine Learning is the study of how to build computer systems that adapt and improve with experience.

Classical AI deals mainly with **deductive** reasoning, learning represents **inductive reasoning**. Deductive reasoning arrives at answers to queries relating to a particular situation starting from a set of general axioms, whereas inductive reasoning arrives at general axioms from a set of particular instances.

Definition: A computer program is said to *learn* from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Machine learning is particularly attractive in several real life problem because of the following reasons:

- Some tasks cannot be defined well except by example
- Working environment of machines may not be known at design time

Deductive: Deduce rules/facts from already known rules/facts. (We have already dealt with this)

$$(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow C)$$

Inductive: Learn <u>new</u> rules/facts from a data set $\mathcal{D}$.

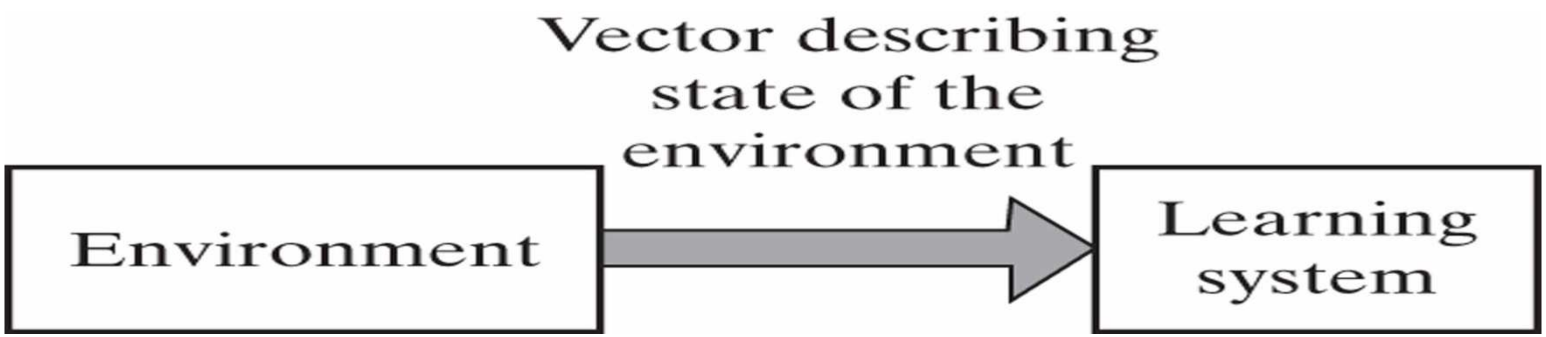
$$\mathcal{D} = \{\mathbf{x}(n), y(n)\}_{n=1\ldots N} \Rightarrow (A \Rightarrow C)$$

Type of learning:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

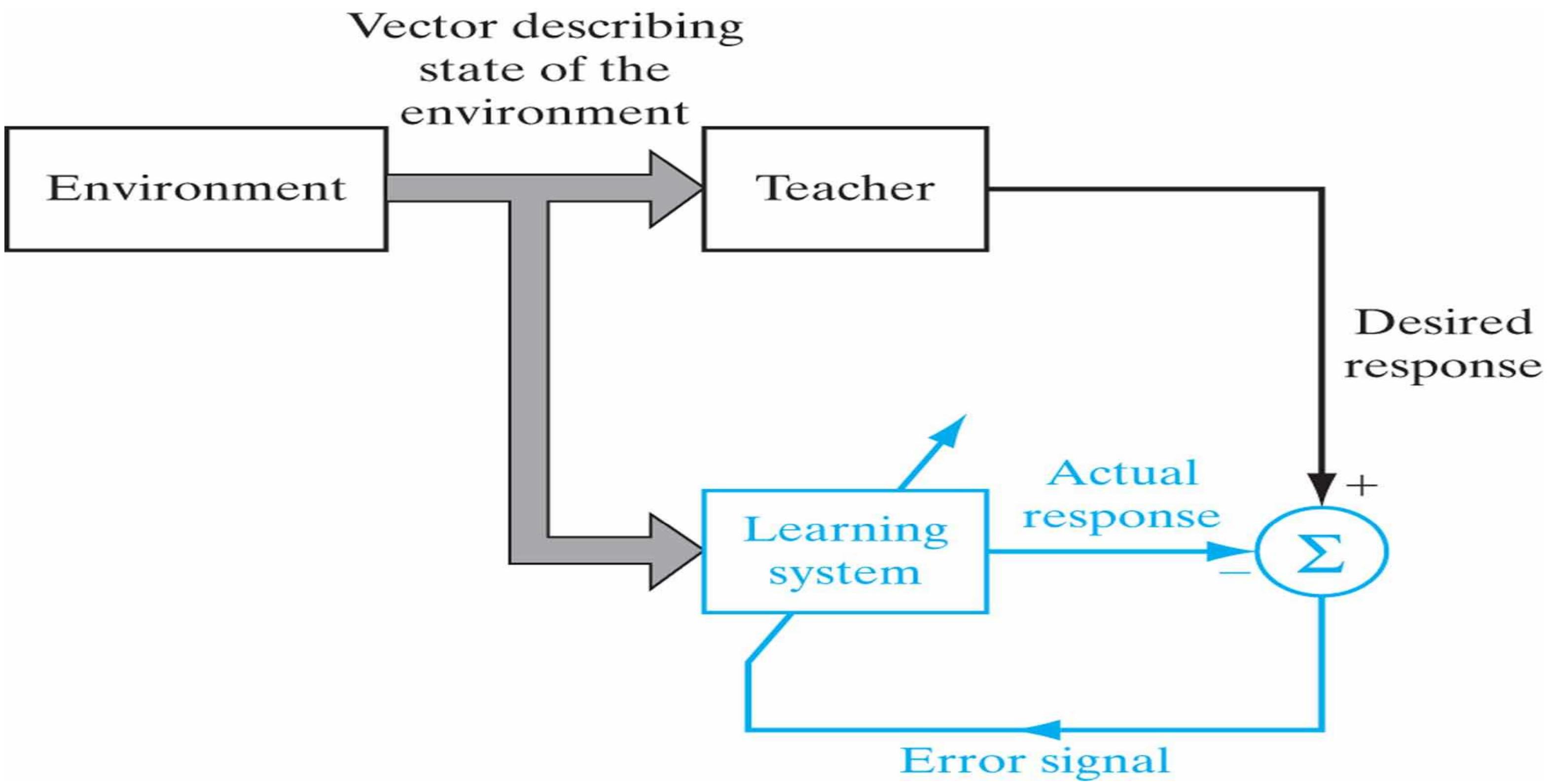
Unsupervised Learning

- Learning without a teacher.
- No teacher to oversee the learning process.
- No knowledge.
- The agent learns patterns in the input even though no explicit feedback is supplied.
- The most common unsupervised learning task is clustering: detecting potentially useful clusters of input examples.
- For example, a taxi agent might gradually develop a concept of "good traffic days" and "bad traffic days" without ever being given labeled examples of each by a teacher.



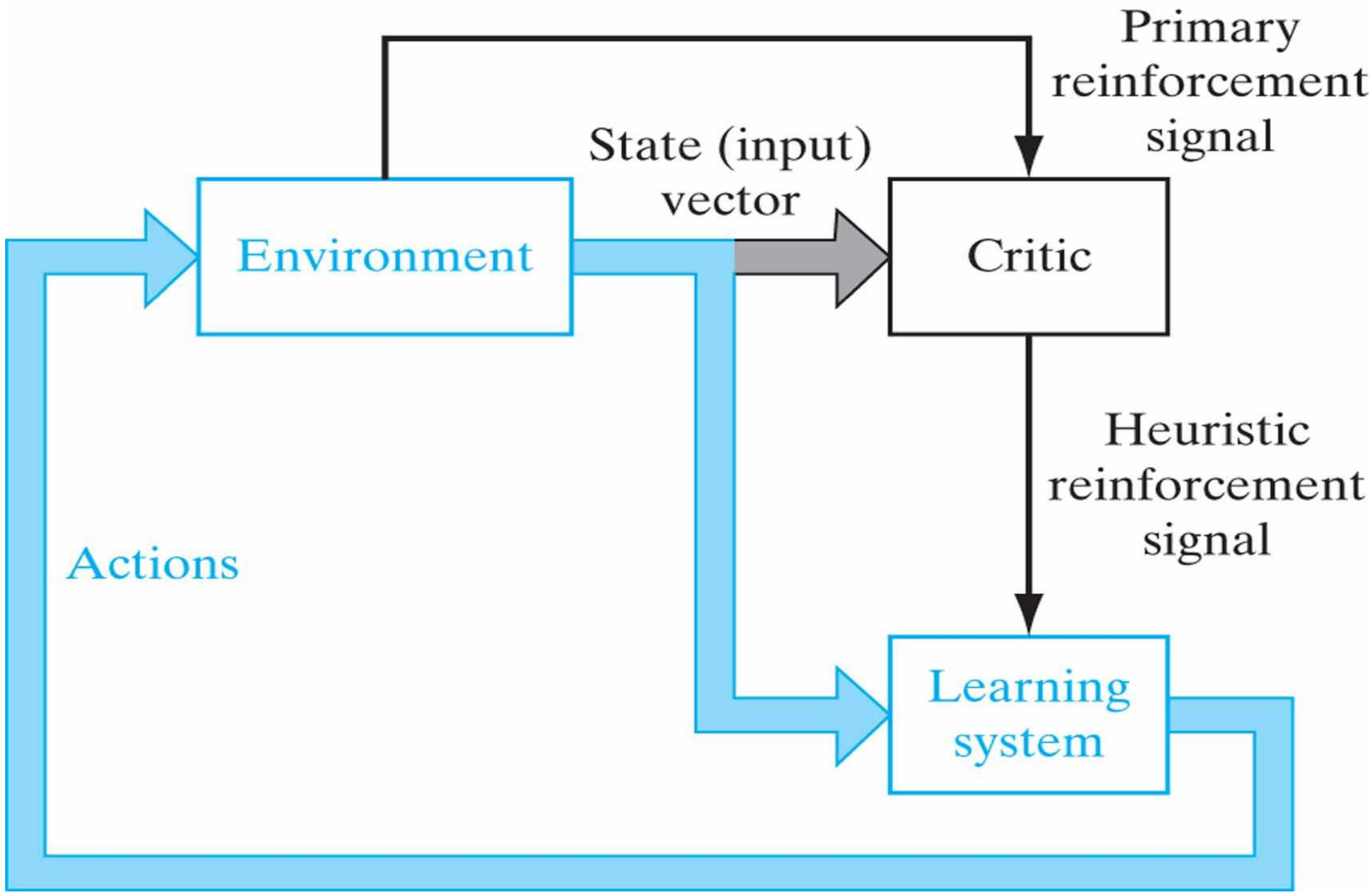
Supervised Learning

- With a teacher.
- Teacher has knowledge of environment
- Knowledge is represented by a set of input-output examples.
- The agent observes some example input–output pairs and learns a function that maps from input to output.



Reinforcement Learning

- The agent learns from a series of reinforcements—rewards or punishments.
- For example, the lack of a tip at the end of the journey gives the taxi agent an indication that it did something wrong.
- It is up to the agent to decide which of the actions prior to the reinforcement were most responsible for it.

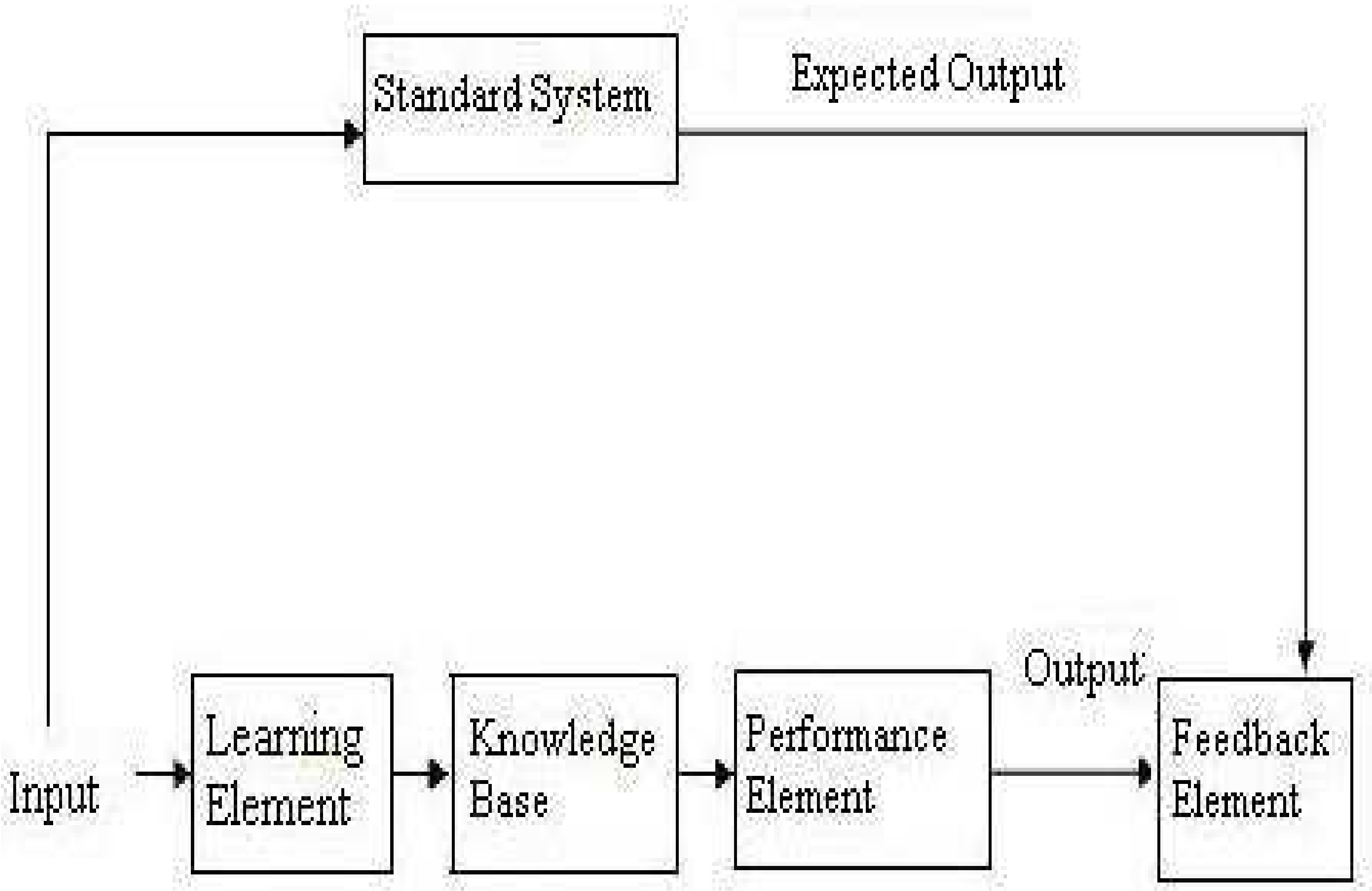


Block diagram of reinforcement learning;

- Explicit knowledge encoding may be difficult and not available
- Environments change over time
- Biological systems learn

Recently ,learning is widely used in a number of application areas including.

- Data mining and knowledge discovery
- Speech/image/video (pattern) recognition
- Adaptive control
- Autonomous vehicles/robots
- Decision support systems
- Bioinformatics
- WWW



Learning System Model

The learning system model consists of the following components.

1.Learning element
2.Knowledge base
3.Performance element
4.Feedback element
5. Standard system.

1. Learning element

It receives and processes the input obtained from a person (i.e. a teacher), from reference material like magazines, journals, etc, or from the environment at large.

2. Knowledge base

This is somewhat similar to the database. Initially it may contain some basic knowledge. Thereafter it receives more knowledge which may be new and so be added as it is or it may replace the existing knowledge.

3. Performance element

It uses the updated knowledge base to perform some tasks or solves some problems and produces the corresponding output.

4. Feedback element

It is receiving the two inputs, one from learning element and one from standard (or idealized) system. This is to identify the differences between the two inputs. The feedback is used to determine what should be done in order to produce the correct output.

5. Standard system

It is a trained person or a computer program that is able to produce the correct output. In order to check whether the machine learning system has learned well, the same input is given to the standard system. The outputs of standard system and that of performance element are given as inputs to the feedback element for the comparison. Standard system is also called idealized system.

Some example of machine learning:

- Learning to recognize spoken words .
- Learning to drive an autonomous vehicle .
- Learning to classify new astronomical structures.

Decision Trees : Decision trees are a class of learning models that are more robust to noise as well as more powerful as compared to concept learning.

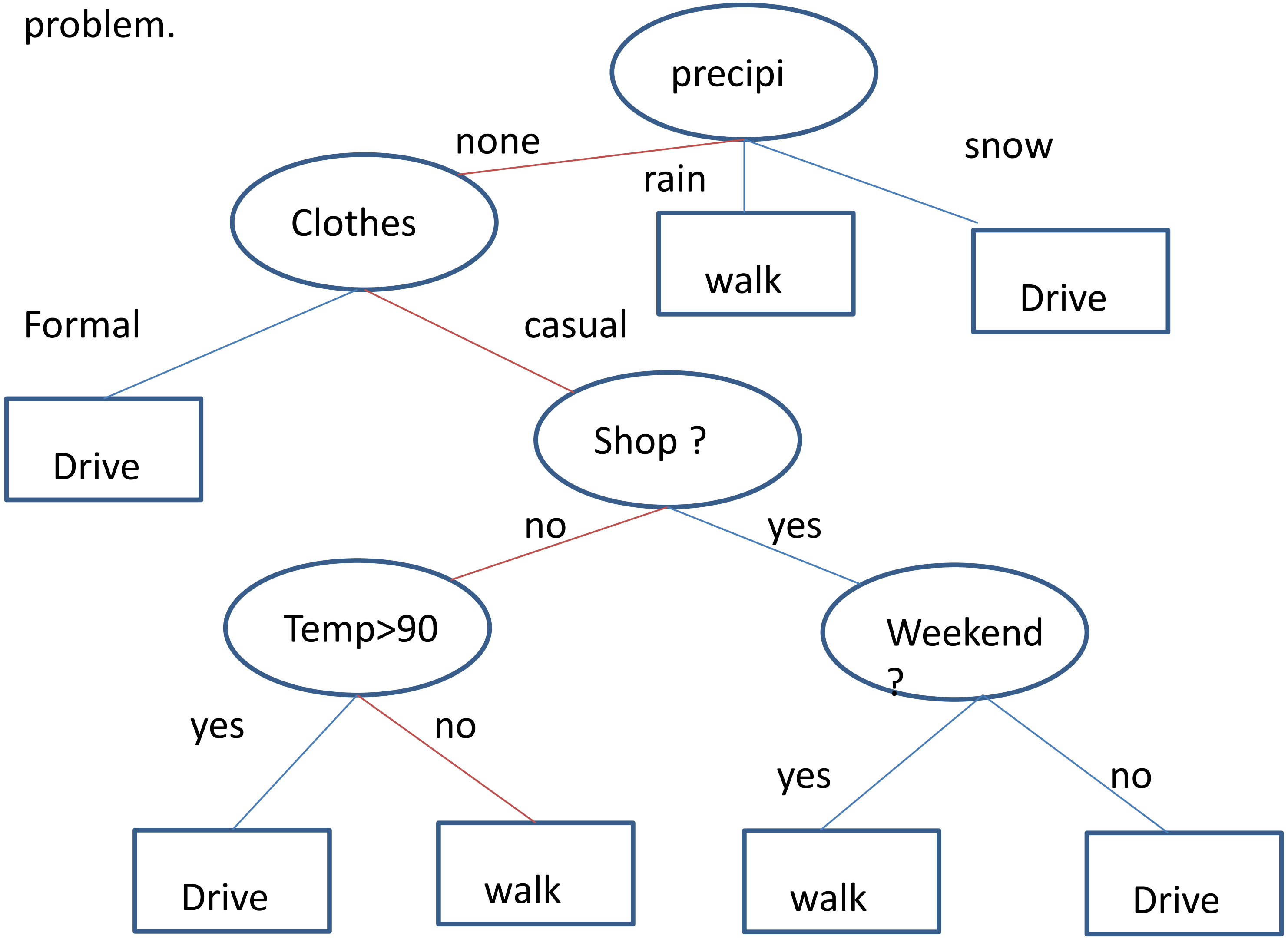
A decision-tree learning algorithm approximates a target concept using a tree representation, where each internal node corresponds to an attribute, and every terminal node corresponds to a class.

- There are two types of nodes:

- **Internal node**.- Splits into different branches according to the different values the corresponding attribute can take.
- **Terminal Node.-** Decides the class assigned to the example.

Decision trees adopt a DNF (Disjunctive Normal Form) representation. For a fixed class, every branch from the root of the tree to a terminal node with that class is a conjunction of attribute values; different branches ending in that class form a disjunction.

Another well-loved learning algorithm makes hypotheses in the form of decision trees. **In a decision tree, each node represents a question, and the arcs represent possible answers.** We can use this decision tree to find out what prediction we should make in the drive/walk problem.



We'd start by asking what the current precipitation is. If it's snow, we just stop asking questions and drive.
If there's no precipitation, then we have to ask what kind of clothes the neighbor is wearing. If they're formal, she'll drive. If not, we have to ask another question.

We can always continue asking and answering questions until we get to a leaf node of the tree; the leaf will always contain a prediction.

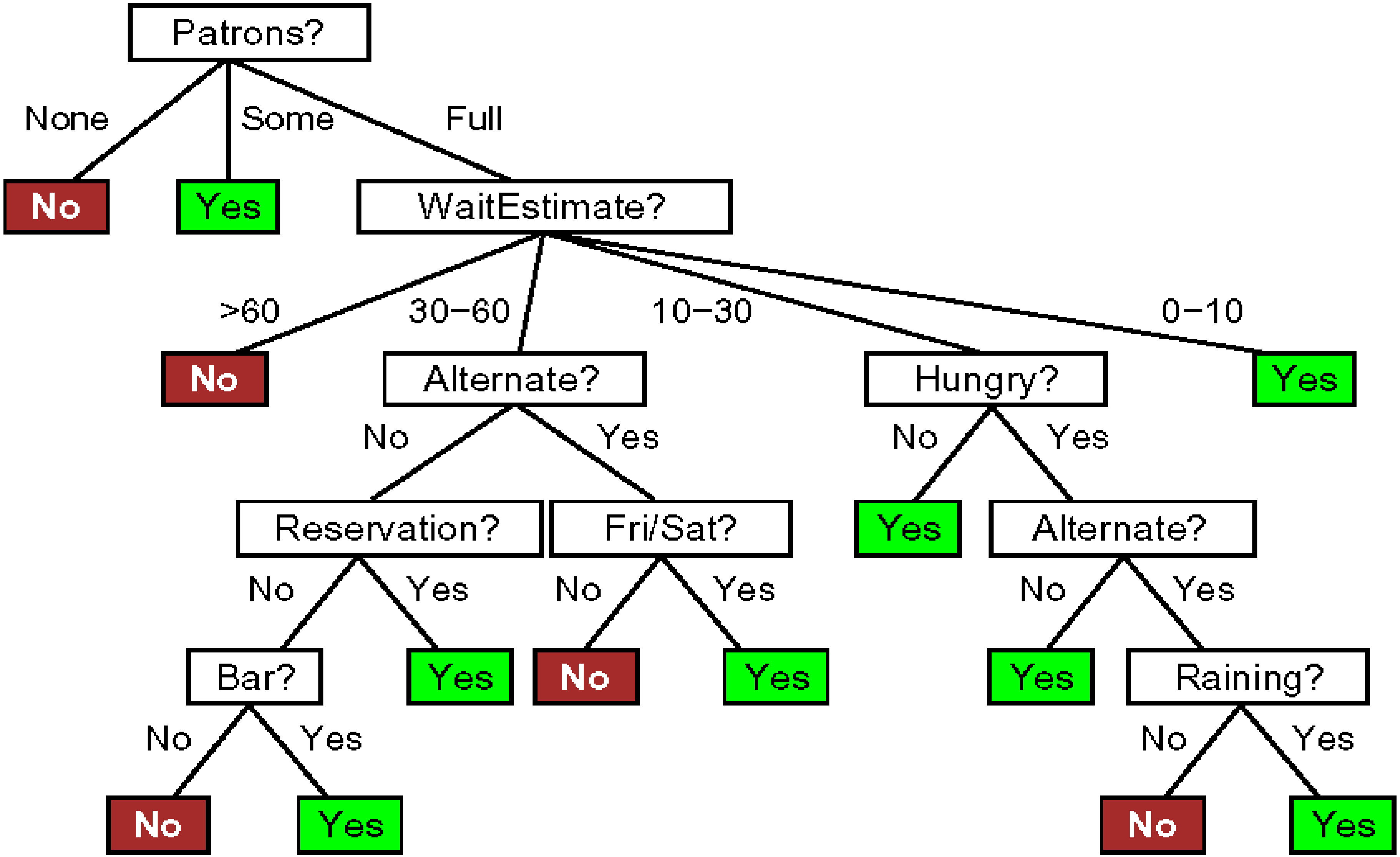
An example:Whether to wait for a table at a restaurant

Goal Predicate: ***WillWait***. To set this as learning problem, what attributes are available to describe examples in the domain.

10 attributes:

1. ***Alternate*: ** Is there a suitable alternative restaurant nearby? {yes,no}
2. ***Bar***: Is there a bar to wait in? {yes,no}
3. ***Fri/Sat***: Is it Friday or Saturday? {yes,no}
4. ***Hungry***: Are you hungry? {yes,no}
5. ***Patrons***: How many are seated in the restaurant? {none, some, full}
6. ***Price***: Price level {\$,\$\$,\$\$\$}
7. ***Raining***: Is it raining? {yes,no}
8. ***Reservation***: Did you make a reservation? {yes,no}
9. ***Type***: Type of food {French,Italian,Thai,Burger}
10. ***WaitEstimate***: {0-10 min, 10-30 min, 30-60 min, >60 min}

The wait@restaurant decision tree



Statistical learning Model :

It is a framework for Machine learning drawing from the fields of statistics and functional analysis . Statistical learning theory deals with the problem of finding a predictive function based on data. Statistical learning theory has led to successful applications in fields such as computer vision, speech recognition, bioinformatics

Statistical Learning

- In which, LEARNING is a form of uncertain reasoning from observation.

- Agents can handle uncertainty by using the methods of probability and decision theory , but first they must learn their probabilistic theories of the world from experience.

- Here data are evidence i.e. instantiations of some or all of the random variables describing the domain .
- The hypothesis are probabilistic theories of how the domain works.

Learning With Complete Data

In machine learning, **naive Bayes classifiers** are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 3" in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness and diameter features.

For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods.

Abstractly, naive Bayes is a conditional probability model: given a problem instance to be classified, represented by a vector representing some *n* features
(dependent variables), it assigns to this instance probabilities

$$p(C_k|x_1, \dots, x_n)$$

for each of *k* possible outcomes or *classes*.
The problem with the above formulation is that if the number of features *n* is large or if a feature can take on a large number of values, then basing such a model on probability tables is infeasible. We therefore reformulate the model to make it more tractable. Using Bayes' theorem, the conditional probability can be decomposed as

$$p(C_k|\mathbf{x}) = \frac{p(C_k)\ p(\mathbf{x}|C_k)}{p(\mathbf{x})}.$$

the above equation can be written as

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}.$$

Learning with hidden data : EM –Algorithm:

Maximum Likelihood estimate (MLE-EM); the maximum a posteriori (MAP-EM) is used when some data is missing i.e. some observable data is hidden.

Expectation-Maximization (EM) is a technique used in point estimation. Given a set of observable variables X and unknown (latent) variables or missing value Z we want to estimate unknown parameters θ in a statistical model, along with a **likelihood function L(θ; X,Z)=p(X,Z/θ)**, the **MLE** of the unknown parameters is determined by the **marginal likelihood** of the observed data

$$L(\boldsymbol{\theta}; \mathbf{X})=\mathbf{p}(\mathbf{X}/\boldsymbol{\theta})= \sum_Z p(X,Z/\boldsymbol{\theta})$$

if Z is a sequence of events, so that the number of values grows exponentially with the sequence length, making the exact calculation of the sum extremely difficult).

Notation:

X -Observed variables.

Z -Latent (unobserved) variables .

$\boldsymbol{\theta}^{(t)}$ -The estimate of the parameters at iteration t.

ℓ**(θ)**- The marginal log-likelihood log p(x|θ).

log p(x , z|θ) -The complete log-likelihood, i.e., when we know the value of Z.

q(z|x,θ) -Averaging distribution, a free distribution that EM gets to vary.

Q(θ| $\theta^{(t)}$)-The expected complete log-likelihood $\sum_Z q\left(\frac{Z}{\mathrm{x}}, \theta\right)$log p(x,z|θ).

H(q)- Entropy of the distribution q(z | x,θ).

The EM algorithm to find the MLE of the marginal likelihood by iteratively applying the following two steps

Expectation step (E step): Calculate the **expected value** of the **log likelihood** function, with respect to the **conditional distribution** of Z given X under the current estimate of the parameters $\theta^{(t)}$.

E-step :

Compute $Q(\theta|\theta^{(t)}) = E_{p(z|x, \theta^{(t)})}[\log p(x,z|\theta)]$

Maximization step (M step): Find the parameter that maximizes this quantity

M-step :

$\theta^{(t+1)} = \arg\max_{\theta} E_{p(z|x,\theta(t))}[\log p(x,z|\theta)]$