Introduction
WHAT IS MACHINE LEARNING?

In cognitive science, learning is typically referred to as the process of gaining information through observation.

Ever since computers were invented, we have wondered whether they might be made to learn. If we could understand how to program them to learn-to improve automatically with experience-the impact would be dramatic.

Imagine computers learning from medical records which treatments are most effective for new diseases, houses learning from experience to optimize energy costs based on the particular usage patterns of their occupants, or personal software assistants learning the evolving interests of their users in order to highlight especially relevant stories from the online morning newspaper.

A successful understanding of how to make computers learn would open up many new uses of computers and new levels of competence and customization and a Detailed understanding of information processing algorithms for machine learning might lead to a better understanding of human learning abilities (and disabilities) as well.

In order to make computers to learn nearly as well as people learn, there are algorithms have been invented that are effective for certain types of learning tasks.

Many practical computer programs have been developed to exhibit useful types of learning, and significant commercial applications have begun to appear.

For problems such as speech recognition, algorithms based on machine learning outperform all other approaches that have been attempted to up to date.

In the field known as data mining, machine learning algorithms are being used routinely to discover valuable knowledge from large commercial databases containing equipment maintenance records, loan applications, financial transactions, medical records,…etc.

As our understanding of computers continues to mature, it seems inevitable that machine learning will play an increasingly central role in computer science and computer technology.

Machine Learning(ML) Applications :

A few specific achievements provide a glimpse of the state of the art in ML:

Programs have been developed that successfully

- Learn to recognize spoken words
- Predict recovery rates of pneumonia patients
- Detect fraudulent use of credit cards
- Drive autonomous vehicles on public highways
- Play games such as backgammon at levels approaching the performance of human world

Theoretical results have been developed that characterize the fundamental relationship among the number of training examples observed, the number of hypotheses under consideration, and the expected error in learned hypotheses.

A hypothesis, in a scientific context, is a testable statement about the relationship between two or more variables or a proposed explanation for some observed phenomenon.

Machine learning applications are summarized below.

•**Learning to recognize spoken words.**

All of the most successful speech recognition systems employ machine learning in some form. For example, the SPHINX system learns speaker-specific strategies for recognizing the primitive sounds and words from the observed speech signal.

Neural network learning methods and methods for learning hidden Markov models are effective for automatically customizing to, individual speakers, vocabularies, microphone characteristics, background noise, etc. Similar techniques have potential applications in many signal-interpretation problems.

•**Learning to drive an autonomous vehicle.**

Machine learning methods have been used to train computer-controlled vehicles to steer correctly when driving on a variety of road types.

For example, the ALVINN system has used its learned strategies to drive unassisted at 70 miles per hour, for 90 miles on public highways among other cars. Similar techniques have possible applications in many sensor-based control problems.

• **Learning to classify new astronomical structures.**

Machine learning methods have been applied to a variety of large databases to learn general regularities implicit in the data. For example, decision tree learning algorithms have been used by NASA to learn how to classify celestial objects from the second Palomar Observatory Sky Survey. This system is now used to automatically classify all objects in the Sky Survey, which consists of three tera bytes of image data.

• **Learning to play world-class backgammon.**

The most successful computer programs for playing games such as backgammon are based on machine learning algorithms. For example, the world's top computer program for backgammon, TD-GAMMON, learned its strategy by playing over one million practice games against itself.

It now plays at a level competitive with the human world champion. Similar techniques have applications in many practical problems where very large search spaces must be examined efficiently.

Machine learning, can be described as a variety of learning paradigms, algorithms, theoretical results, and applications. Machine learning is inherently a multidisciplinary field. It draws on results from artificial intelligence, probability and statistics, computational complexity theory, control theory, information theory, philosophy, psychology, neurobiology, and other fields.

The following summarizes the key ideas from each of these fields that impact the field of machine learning.

- **Artificial intelligence**

Learning symbolic representations of concepts. Machine learning as a search problem. Learning as an approach to improving problem solving. Using prior knowledge together with training data to guide learning.

- ***Bayesian methods***

Bayes theorem as the basis for calculating probabilities of hypotheses. The naive Bayes classifier. Algorithms for estimating values of unobserved variables.

•***Computational complexity theory***

Theoretical bounds on the inherent complexity of different learning tasks, measured in terms of the computational effort, number of training examples, number of mistakes, etc. required in order to learn. Control theory Procedures that learn to control processes in order to optimize predefined objectives and that learn to predict the next state of the process they are controlling.

• ***Information theory***

Measures of entropy and information content. Minimum description length approaches to learning. Optimal codes and their relationship to optimal training sequences for encoding a hypothesis. Philosophy Occam's razor, suggesting that the simplest hypothesis is the best. Analysis of the justification for generalizing beyond observed data.

•***Psychology and neurobiology***

The power law of practice, which states that over a very broad range of learning problems, people's response time improves with practice according to a power law. Neurobiological studies motivating artificial neural network models of learning.

• ***Statistics***

Characterization of errors (e.g., bias and variance) that occur when estimating the accuracy of a hypothesis based on a limited sample of data. Confidence intervals, statistical tests.

1.1 **WELL-POSED LEARNING PROBLEMS**

- Before answering the question ‘What is machine learning?’ more fundamental questions that peep into one’s mind are
  - Do machines really learn?
  - If so, how do they learn?

Do machines really learn?

- At the onset, it is important to formalize the definition of machine learning.
- This will itself address the first question, i.e. if machines really learn.

- There are multiple ways to define machine learning. But the one which is perhaps most relevant, concise and accepted universally is the one stated by Tom M. Mitchell.

‘ 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. ’

- What this essentially means is that a machine can be considered to learn if it is able to gather experience by doing a certain task and improve its performance in doing the similar tasks in the future.
- When we talk about past experience, it means past data related to the task.
- This data is an input to the machine from some source.

Considering few learning tasks in Machine learning. For the purposes we will define learning broadly, to include any computer program that improves its performance at some task through experience.

For example, a computer program that learns to play checkers might improve its performance as measured by its ability to win at the class of tasks involving playing checkers games, through experience obtained by playing games against itself.

In general, to have a well-defined learning problem, we must identity these three features: the class of tasks, the measure of performance to be improved, and the source of experience.

• A checkers learning problem:

1. Task T: playing checkers
2. Performance measure P: percent of games won against opponents
3. Training experience E: playing practice games against itself

We can specify many learning problems in this fashion, such as learning to recognize handwritten words, or learning to drive a robotic automobile autonomously.

- **A handwriting recognition learning problem:**

1. Task T: recognizing and classifying handwritten words within images
2. Performance measure *P: percent of words correctly classified*
3. Training experience E: a database of handwritten words with given classifications

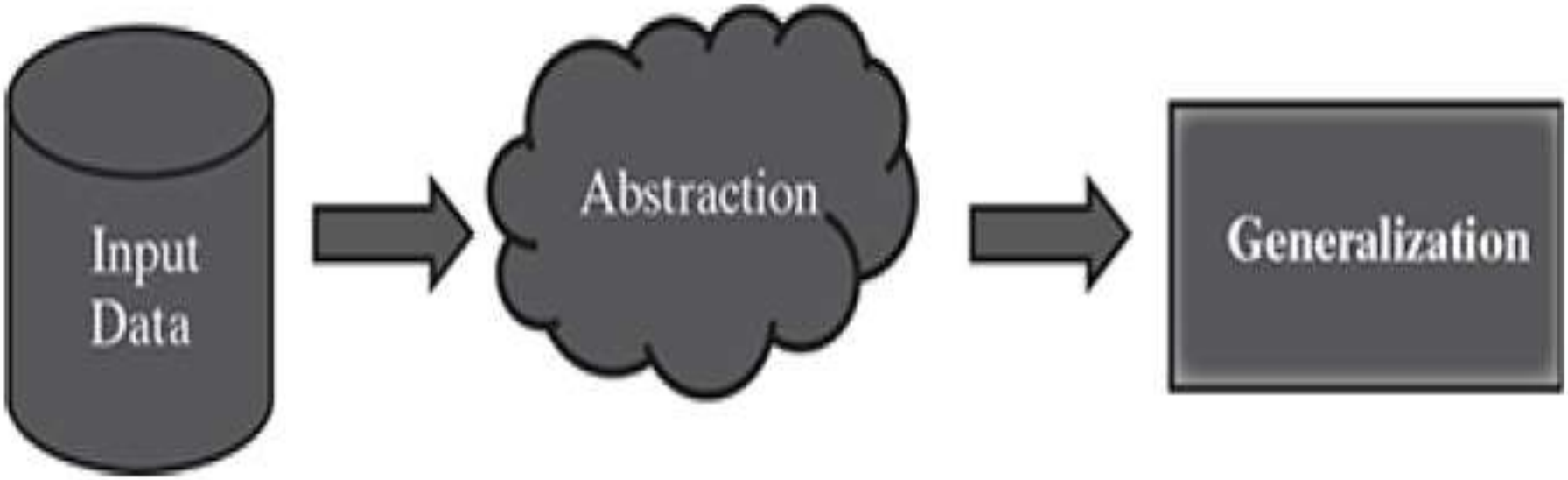
- **A robot driving learning problem:**

1. Task T: driving on public four-lane highways using vision sensors
2. *Performance measure P: average distance traveled before an error (as judged* by human overseer)
3. Training experience E: a sequence of images and steering commands recorded while observing a human driver

How do machines learn?

- The basic machine learning process can be divided into three parts.
  1. Data Input: Past data or information or training data is utilized as a basis for future decision-making
  2. Abstraction: The input data is represented in a broader way through the underlying algorithm
  3. Generalization: The abstracted representation is generalized to form a framework for making decisions

The following figure is a schematic representation of the machine learning process.



Process of machine learning

INPUT DATA

- Let’s put the things in perspective of the human learning process and try to understand the machine learning process more clearly.
- Let’s consider the situation of typical process of learning from classroom and books, and preparing for the examination.
- It is a tendency of many students to try and memorize as many things as possible.

- This may work well when the scope of learning is not so vast.
- Also, the kinds of questions which are asked in the examination are pretty much simple and straightforward.
- The questions can be answered by simply writing the same things which have been memorized.
- However, as the scope gets broader and the questions asked in the examination gets more complex, the strategy of memorizing doesn't work well.
- The number of topics may get too vast for a student to memorize.

- Also, the capability of memorizing varies from student to student.
- Together with that, since the questions get more complex, a direct reproduction of the things memorized may not help.
- The situation continues to get worse as the student graduates to higher classes.
- So, what we see in the case of human learning is that just by great memorizing and perfect recall, i.e. just based on knowledge input, students can do well in the examinations only till a certain stage.

- Beyond that, a better learning strategy needs to be adopted:
  1. to be able to deal with the vastness of the subject matter and the related issues in memorizing it
  2. to be able to answer questions where a direct answer has not been learnt
- A good option is to figure out the key points or ideas amongst a vast pool of knowledge is a data input.
- This helps in creating an outline of topics and a conceptual mapping of those outlined topics with the entire knowledge pool.

Abstraction

- During the machine learning process, knowledge is fed in the form of input data.
- However, the data cannot be used in the original shape and form.
- As we saw in the example above, abstraction helps in deriving a conceptual map based on the input data.
- This map, or a **model** as it is known in the machine learning paradigm, is summarized knowledge representation of the raw data.

- The model may be in any one of the following forms
  - **Computational blocks like if/else rules**
  - **Mathematical equations (Calculating probabilities)**
  - **Specific data structures like trees or graphs**
  - **Logical groupings of similar observations**
- The choice of the model used to solve a specific learning problem is a human task.
- The decision related to the choice of model is taken based on multiple aspects, some of which are listed below:

- The type of problem to be solved: Whether the problem is related to forecast or prediction, analysis of trend, understanding the different segments or groups of objects, etc.
- Nature of the input data: How exhaustive the input data is, whether the data has no values for many fields, the data types, etc.
- Domain of the problem: If the problem is in a business critical domain with a high rate of data input and need for immediate inference, e.g. fraud detection problem in banking domain.

- Once the model is chosen, the next task is to fit the model based on the input data.
- Let's understand this with an example.
- In a case where the model is represented by a mathematical equation, say '$y = c_1 + c_2 x$' *(the model is known as simple* linear regression which we will study in a later), based on the input data, we have to find out the values of c_1 and c_2 .
- Otherwise, the equation (or the model) is of no use.
- So, fitting the model, in this case, means finding the values of the unknown coefficients or constants of the equation or the model.
- This process of fitting the model based on the input data is known as training.
- Also, the input data based on which the model is being finalized is known as training data.

Generalization

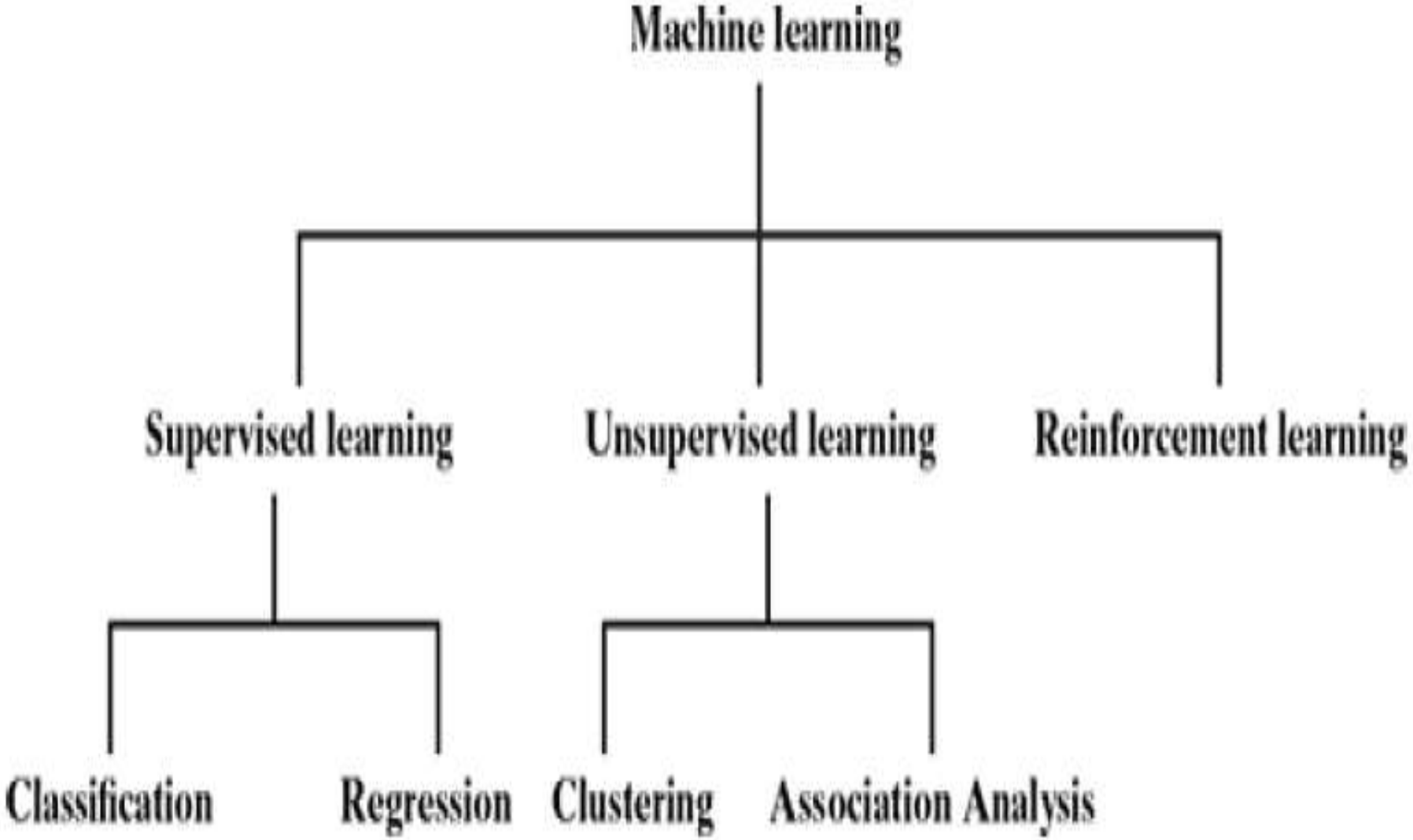
- The first part of machine learning process is abstraction i.e. abstract the knowledge which comes as input data in the form of a model.
- However, this abstraction process, or more popularly training the model, is just one part of machine learning.
- The other key part is to tune up the abstracted knowledge to a form which can be used to take future decisions.
- This is achieved as a part of generalization.

- This part is quite difficult to achieve.
- This is because the model is trained based on a finite set of data, which may possess a limited set of characteristics.
- But when we want to apply the model to take decision on a set of unknown data, usually termed as test data, we may encounter two problems:
  1. The trained model is aligned with the training data too much, hence may not portray the actual trend.
  2. The test data possess certain characteristics apparently unknown to the training data.

CATEGORIES or TYPES OF MACHINE LEARNING TECHNIQUES

- As given in Figure below, Machine learning can be classified into three broad categories:

1. Supervised learning – Also called predictive learning. A machine predicts the class of unknown objects based on prior class-related information of similar objects.
2. Unsupervised learning – Also called descriptive learning. A machine finds patterns in unknown objects by grouping similar objects together.
3. Reinforcement learning – A machine learns to act on its own to achieve the given goals.



Types of machine learning

Supervised learning

- The major motivation of supervised learning is to learn from past information.
- So what kind of past information does the machine need for supervised learning?
- It is the information about the task which the machine has to execute.
- In context of the definition of machine learning, this past information is the experience.

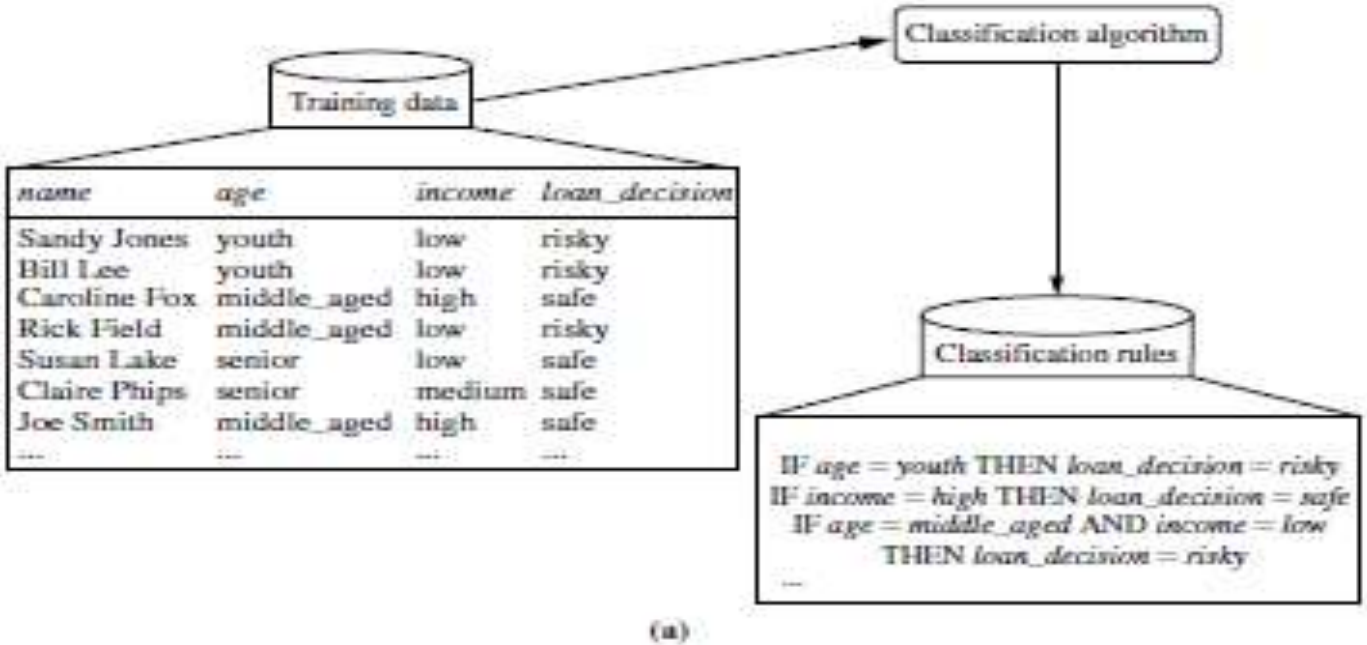
- Some examples of supervised learning are
  - Classifying texts such as classifying a set of emails as spam or non-spam
  - classifying loan application whether loan sanction to bank is safe or risky
  - Predicting the results of a game
  - Predicting whether a tumour is malignant or benign
  - Predicting the price of domains like real estate, stocks, etc.

- Though both of them are prediction problems, in one case we are trying to predict which category or class an unknown data belongs to whereas in the other case we are trying to predict an absolute value and not a class.
- When we are trying to predict a categorical or nominal variable, the problem is known as a **classification problem**.
- Whereas when we are trying to predict a real-valued variable, the problem falls under the category of **prediction or regression**.

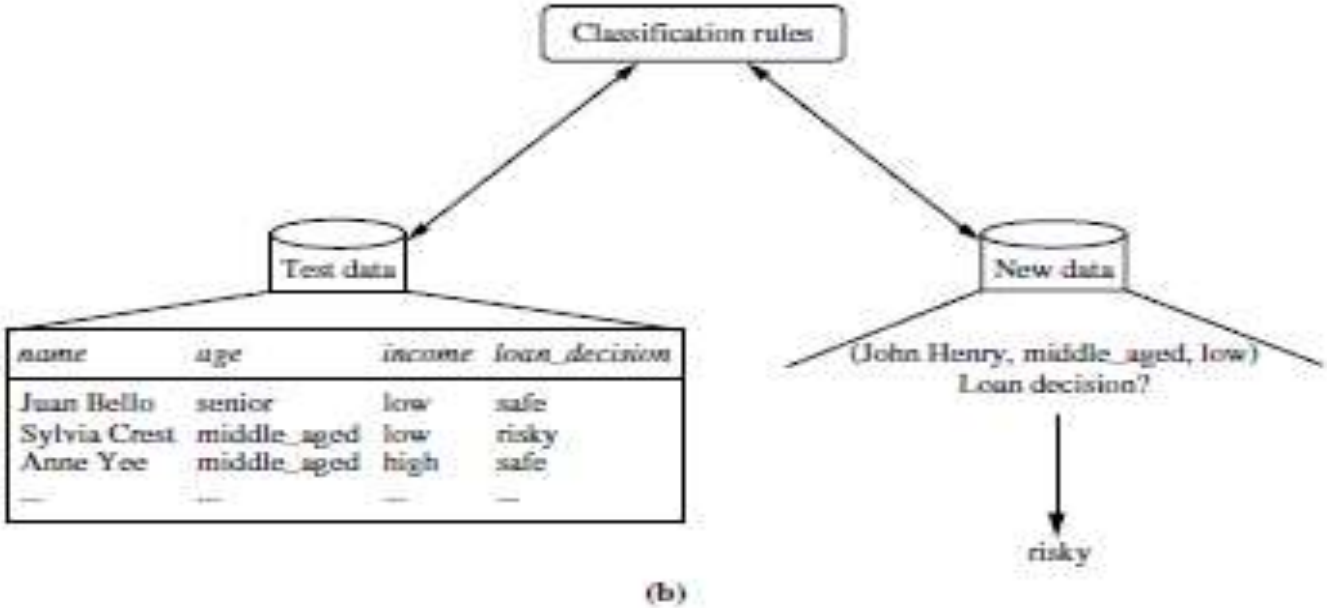
CLASSIFICATION

Data classification is a two-step process, consisting of a *learning step (where a classification model is constructed) and a classification step (where* the model is used to predict class labels for given data). The process is shown for the loan application data as given in Figure.

In the first step, a classifier is built describing a predetermined set of data classes or concepts. This is the learning step (or training phase), where a classification algorithm builds the classifier by analyzing or "learning from" a training set made up of database tuples and their associated class labels. A tuple, *X, is represented by an n-dimensional* attribute vector, *X = {x1, x2, : : : , xn }, depicting n measurements made on the tuple* from *n database attributes, respectively, A1, A2, : : : , An. Each tuple, X, is assumed to* belong to a predefined class as determined by another database attribute called the class label attribute. The class label attribute is discrete-valued and unordered. It is *categorical* (nominal) attribute means in each value serves as a category or class. The individual tuples making up the training set are referred to as training tuples and are randomly sampled from the database under analysis. In the context of classification, data tuples can be referred to as *samples, examples, instances, data points, or objects.*


Classification algorithm
Training data
name age income loan_decision
Sandy Jones youth low risky
Bill Lee youth low risky
Caroline Fox middle_aged high safe
Rick Field middle_aged low risky
Susan Lake senior low safe
Claire Phips senior medium safe
Joe Smith middle_aged high safe
... ...
Classification rules
IF age = youth THEN loan_decision = risky
IF income = high THEN loan_decision = safe
IF age = middle_aged AND income = low
THEN loan_decision = risky
...

(a)


Classification rules
Test data
New data
name age income loan_decision
Juan Bello senior low safe
Sylvia Crest middle_aged low risky
Anne Yee middle_aged high safe
... ...
(John Henry, middle_aged, low)
Loan decision?
risky

(b)

- There are number of popular machine learning algorithms which help in solving classification problems.
- To name a few, Naïve Bayes, Decision tree, and k-Nearest Neighbour, backpropogation algorithms are adopted by many machine learning practitioners.
- A critical classification problem in context of banking domain is identifying potential fraudulent transactions.
- Since there are millions of transactions which have to be scrutinized and assured whether it might be a fraud transaction, it is notpossible for any human being to carry out this task.

- Machine learning is effectively leveraged to do this task and this is a classic case of classification.
- Based on the past transaction data, specifically the ones labelled as fraudulent, all new incoming transactions are marked or labelled as normal or suspicious.
- The suspicious transactions are subsequently segregated for a closer review.
- In summary, classification is a type of supervised learning where a target feature, which is of type categorical, is predicted for test data based on the information imparted by training data. The target categorical feature is known as **class.**

Prediction or Regression

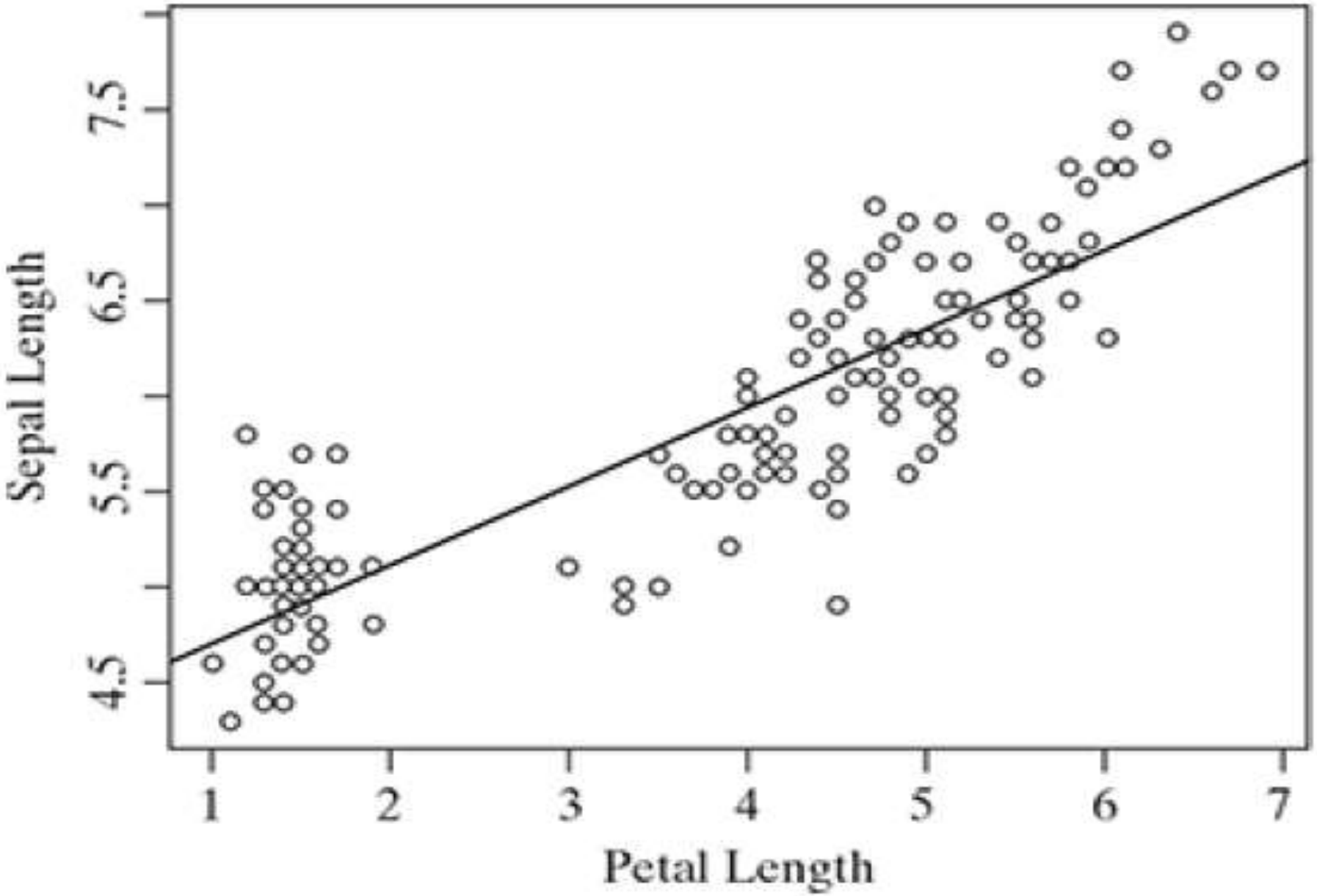
- In linear regression, the objective is to predict numerical features like real estate or stock price, temperature, marks in an examination, sales revenue, etc.
- The underlying predictor variable and the target variable are continuous in nature.
- In case of linear regression, a straight line relationship is ‘fitted’ between the predictor variables and the target variables, using the statistical concept of least squares method.

- As in the case of least squares method, the sum of square of error between actual and predicted values of the target variable is tried to be minimized. In case of simple linear regression, there is only one predictor variable whereas in case of multiple linear regression, multiple predictor variables can be included in the model.

- Obviously, the data related to past as well as the data to be predicted are continuous in nature.
- In a basic approach, a simple linear regression model can be applied with investment as predictor variable and sales revenue as the target variable.
- Figure below shows a typical simple regression model, where regression line is fitted based on values of target variable with respect to different values of predictor variable.
- A typical linear regression model can be represented in the form –

$$y=\alpha + \beta x$$

where *'x' is the predictor variable and 'y' is the target variable.*



Regression

- The input data come from a famous multivariate data set named Iris introduced by the British statistician and biologist Ronald Fisher.
- The data set consists of 50 samples from each of three species of Iris – *Iris setosa, Iris virginica, and Iris versicolor.*
- *Four features were measured for each sample* – sepal length, sepal width, petal length, and petal width.
- These features can uniquely discriminate the different species of the flower.

- The Iris data set is typically used as a training data for solving the classification problem of predicting the flower species based on feature values.
- However, we can also demonstrate regression using this data set, by predicting the value of one feature using another feature as predictor.
- In Figure above, petal length is a predictor variable which, when fitted in the simple linear regression model, helps in predicting the value of the target variable sepal length.

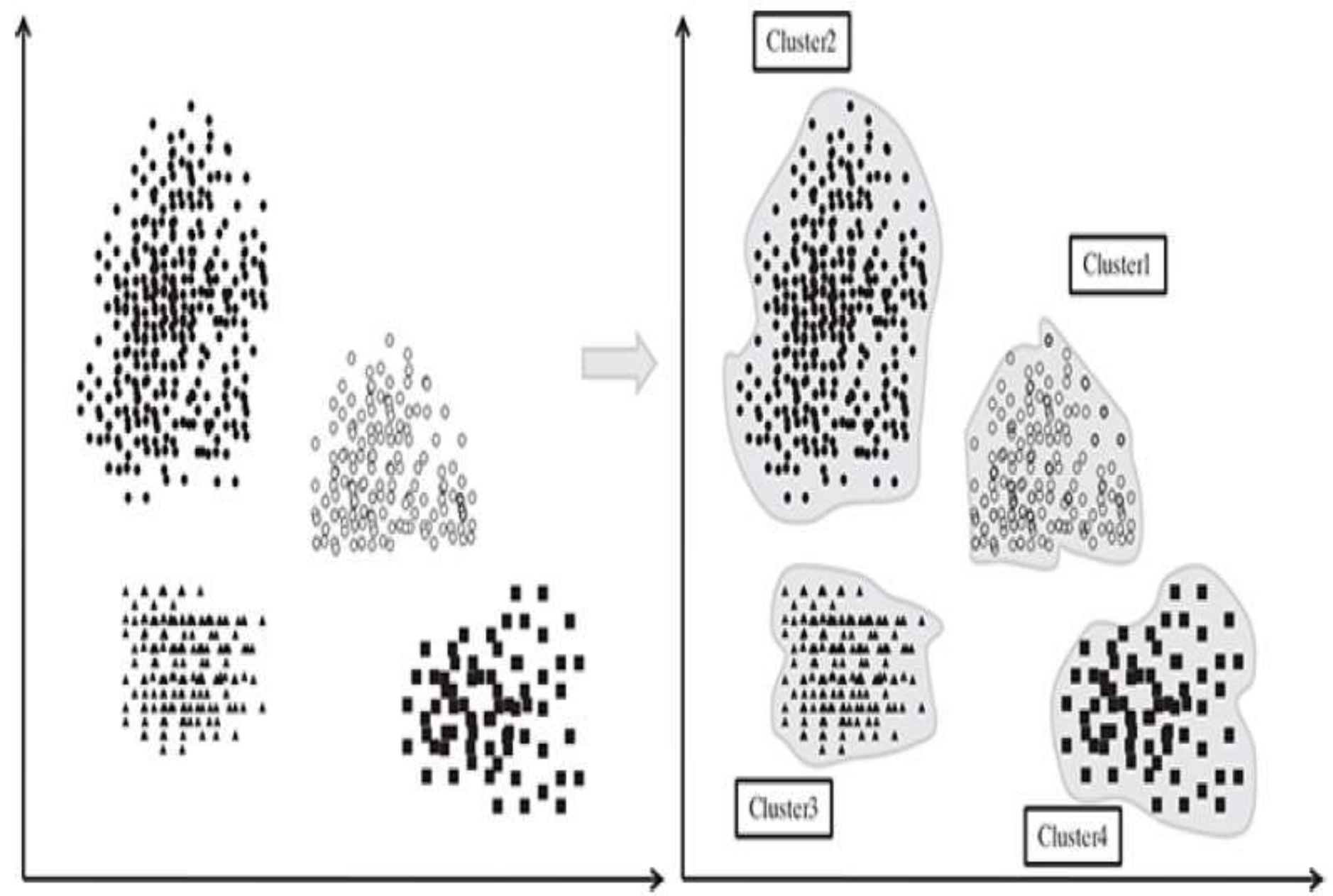
- Typical applications of regression can be seen in
  - Demand forecasting in retails
  - Sales prediction for managers
  - Price prediction in real estate
  - Weather forecast
  - Skill demand forecast in job market

Unsupervised learning

- Unlike supervised learning, in unsupervised learning, there is no labelled training data to learn from and no prediction to be made.
- In unsupervised learning, the objective is to take a dataset as input and try to find natural groupings or **patterns** within the data elements or records.
- Therefore, unsupervised learning is often termed as **descriptive model** and the process of unsupervised learning is referred as **pattern discovery or knowledge discovery.**
- One critical application of unsupervised learning is customer segmentation.

Clustering

- Clustering is the main type of unsupervised learning.
- It intends to group or organize similar objects together.
- For that reason, objects belonging to the same cluster are quite similar to each other while objects belonging to different clusters are quite dissimilar.
- Hence, the objective of clustering to discover the intrinsic grouping of unlabelled data and form clusters, as depicted in Figure below.
- Different measures of similarity can be applied for clustering.



Distance-based clustering

- One of the most commonly adopted similarity measure is distance.
- Two data items are considered as a part of the same cluster if the distance between them is less.
- In the same way, if the distance between the data items is high, the items do not generally belong to the same cluster.

Association Analysis

- Other than clustering of data and getting a summarized view from it, one more variant of unsupervised learning is **association analysis.**
- As a part of association analysis, the association between data elements is identified.
- Let's try to understand the approach of association analysis in context of one of the most common examples, i.e. market basket analysis as shown in Figure below.
- From past transaction data in a grocery store, it may be observed that most of the customers who have bought item A, have also bought item B and item C or at least one of them.

TransID	Items Bought
1	{Butter, Bread}
2	{Diaper, Bread, Milk, Beer}
3	{Milk, Chicken, Beer, Diaper}
4	{Bread, Diaper, Chicken, Beer}
5	{Diaper, Beer, Cookies, Ice cream}
…	…

Market Basket transactions

Frequent itemsets → (Diaper, Beer)

Possible association: Diaper → Beer

Market basket analysis

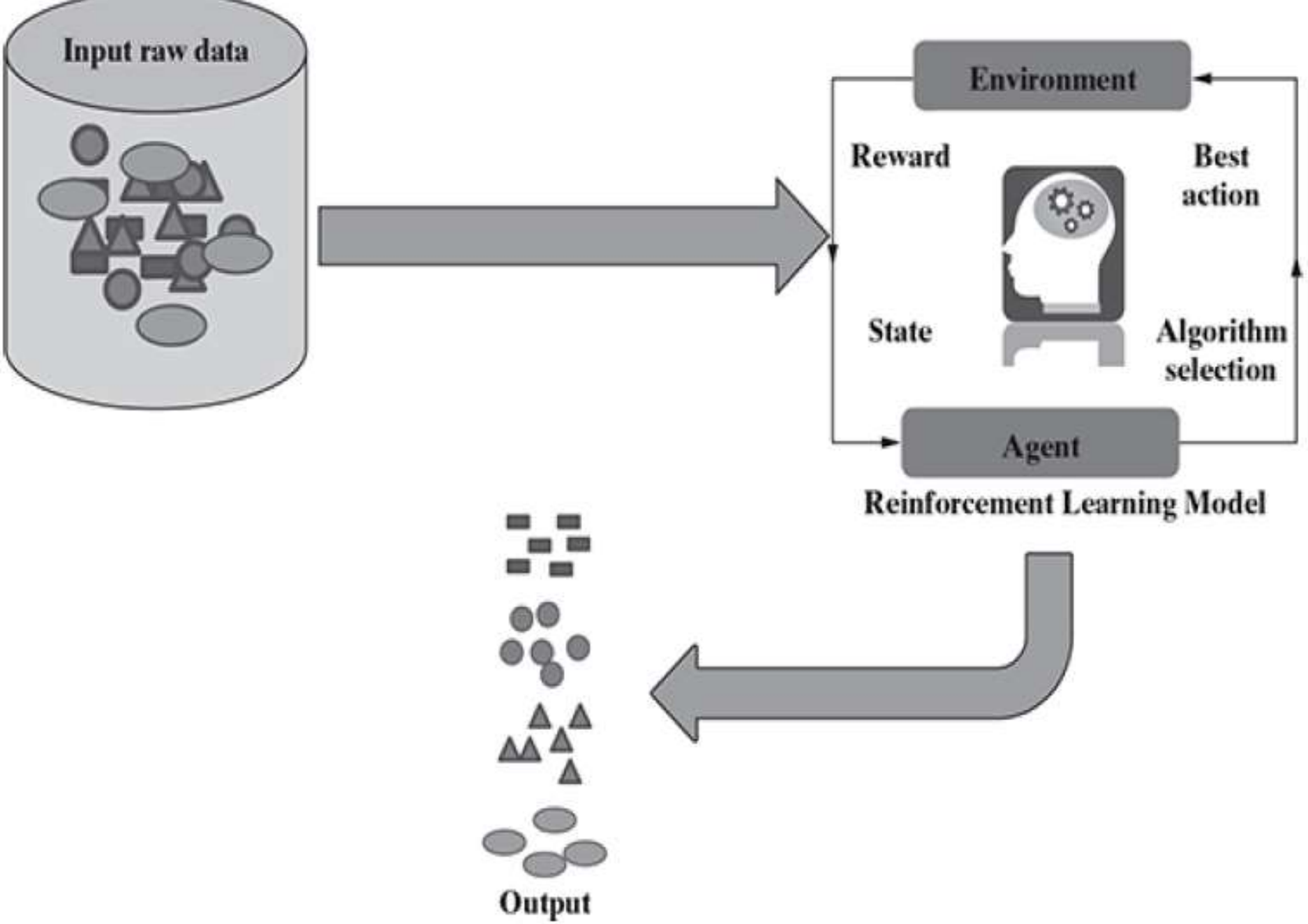
- This means that there is a strong association of the event ‘purchase of item A’ with the event ‘purchase of item B’, or ‘purchase of item C’.
- Identifying these sorts of associations is the goal of association analysis.
- This helps in boosting up sales pipeline, hence a critical input for the sales group.
- Critical applications of association analysis include market basket analysis and recommender systems.

Reinforcement learning

- We have seen babies learn to walk without any prior knowledge of how to do it.
- Often we wonder how they really do it.
- They do it in a relatively simple way.
- First they notice somebody else walking around, for example parents or anyone living around.
- They understand that legs have to be used, one at a time, to take a step.
- While walking, sometimes they fall down hitting an obstacle, whereas other times they are able to walk smoothly avoiding bumpy obstacles.

- When they are able to walk overcoming the obstacle, their parents are elated and appreciate the baby with loud claps / or may be a chocolates.
- When they fall down while circumventing an obstacle, obviously their parents do not give claps or chocolates.
- Slowly a time comes when the babies learn from mistakes and are able to walk with much ease.
- In the same way, machines often learn to do tasks autonomously.
- Let’s try to understand in context of the example of the child learning to walk.

- The action tried to be achieved is walking, the child is the agent and the place with hurdles on which the child is trying to walk resembles the environment.
- It tries to improve its performance of doing the task.
- When a sub-task is accomplished successfully, a reward is given.
- When a sub-task is not executed correctly, obviously no reward is given.
- This continues till the machine is able to complete execution of the whole task.
- This process of learning is known as **reinforcement learning**.



Reinforcement learning

- One contemporary example of reinforcement learning is self-driving cars.
- The critical information which it needs to take care of are speed and speed limit in different road segments, traffic conditions, road conditions, weather conditions, etc.
- The tasks that have to be taken care of are start/stop, accelerate/decelerate, turn to left / right, etc.
- Reinforcement learning is getting more and more attention from both industry and academia.

1.2 DESIGNING A LEARNING SYSTEM

Some of the basic design issues and approaches to machine learning, consider designing a program to learn to play checkers, with the goal of entering it in the world checkers tournament. We adopt the obvious performance measure: the percent of games it wins in this world tournament.

1.2.1 Choosing the Training Experience

The first design choice we face is to choose the type of training experience from which our system will learn. The type of training experience available can have a significant impact on success or failure of the learner. One key attribute is whether the training experience provides direct or indirect feedback regarding the choices made by the performance system.

For example, in learning to play checkers, the system might learn from *direct training examples consisting of individual checkers* board states and the correct move for each. Alternatively, it might have available only *indirect information consisting of the move sequences and final outcomes* of various games played. In this later case, information about the correctness of specific moves early in the game must be inferred indirectly from the fact that the game was eventually won or lost. Here the learner faces an additional problem of *credit assignment, or determining the degree to which each move in* the sequence deserves credit or blame for the final outcome.

Credit assignment can be a particularly difficult problem because the game can be lost even when early moves are optimal, if these are followed later by poor moves. Hence, learning from direct training feedback is typically easier than learning from indirect feedback.

A second important attribute of the training experience is the degree to which the learner controls the sequence of training examples. For example, the learner might rely on the teacher to select informative board states and to provide the correct move for each. Alternatively, the learner might itself propose board states that it finds particularly confusing and ask the teacher for the correct move. *Or the* learner may have complete control over both the board states and (indirect) training classifications, as it does when it learns by playing against itself with no teacher present. Notice in this last case the learner may choose between experimenting with novel board states that it has not yet considered, or honing its skill by playing minor variations of lines of play it currently finds most promising.

A third important attribute of the training experience is how well it represents the distribution of examples over which the final system performance P must be measured. In general, learning is most reliable when the training examples follow a distribution similar to that of future test examples. In our checkers learning scenario, the performance metric P is the percent of games the system wins in the world tournament. If its training experience E consists only of games played against itself, there is an obvious danger that this training experience might not be fully representative of the distribution of situations over which it will later be tested. For example, the learner might never encounter certain crucial board states that are very likely to be played by the human checkers champion.

To proceed with our design, let us decide that our system will train by playing games against itself. This has the advantage that no external trainer need be present, and it therefore allows the system to generate as much training data as time permits. We now have a fully specified learning task.

Consider a checkers learning problem:

- *Task T: playing checkers*
- *Performance measure P: percent of games won in the world tournament*
- *Training experience E: games played against itself*

In order to complete the design of the learning system, we must now choose

1. the exact type of knowledge to be learned
2. a representation for this target knowledge
3. a learning mechanism

1.2.2 Choosing the Target Function

The next design choice is to determine exactly what type of knowledge will be learned and how this will be used by the performance program. Consider with a checkers-playing program that can generate the *legal moves from any board* state. The program needs only to learn how to choose the *best move from among* these legal moves. This learning task is representative of a large class of tasks for which the legal moves that define some large search space are known a priori, but for which the best search strategy is not known. Many optimization problems fall into this class, such as the problems of scheduling and controlling manufacturing processes where the available manufacturing steps are well understood, but the best strategy for sequencing them is not.

Based on this setting where we must learn to choose among the legal moves, the most obvious choice for the type of information to be learned is a program, or function, that chooses the best move for any given board state. Call this function *ChooseMove and use the notation ChooseMove : B -> M to indicate* that this function accepts as input any board from the set of legal board states *B* and produces as output some move from the set of legal moves *M, keeping in mind to* improve the performance P at task T to the problem of learning some particular *Target function such as ChooseMove. The choice of the target function will* therefore be a key design choice.

The target function *V among the many that produce optimal play,* this will make it easier to design a training algorithm. Let us therefore define the target value *V(b) for an arbitrary board state b in B, as follows:*

1. if *b is a final board state that is won, then V(b) = 100*
2. if b is a final board state that is lost, then V(b) = -100
3. if b is a final board state that is drawn, then V(b) = 0
4. if b is a not a final state in the game, then V(b) = V(b1), where b1 is the best final board state that can be achieved starting from b and playing optimally until the end of the game.

1.2.3 Choosing a Representation for the Target Function

Next we must choose a representation that the learning program will use to describe the function V^1 that it will learn. As with earlier design choices, we again have many options. For example, allow the program to represent V^1 using a large table with a distinct entry specifying the value for each distinct board state. Or we could allow it to represent V^1 using a collection of rules that match against features of the board state, or a quadratic polynomial function of predefined board features, or an artificial neural network. In general, this choice of representation involves a crucial tradeoff. On one hand, we wish to pick a very expressive representation to allow representing as close an approximation as possible to the ideal target function V.

On the other hand, the more expressive the representation, the more training data the program will require in order to choose among the alternative hypotheses it can represent. To keep the discussion brief, let us choose a simple representation: for any given board state, the function V^1 will be calculated as a linear combination of the following board features:

- *xl: the number of black pieces on the board*
- *x2: the number of red pieces on the board*
- *x3: the number of black kings on the board*
- *x4: the number of red kings on the board*
- *x5: the number of black pieces threatened by red*
- *X6: the number of red pieces threatened by black*

Thus, our learning program will represent $\hat{V}(b)$ as a linear function of the form

$$\hat{V}(b) = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 + w_5 x_5 + w_6 x_6$$

where w_0 through w_6 are numerical coefficients, or weights, to be chosen by the learning algorithm. Learned values for the weights w_1 through w_6 will determine the relative importance of the various board features in determining the value of the board, whereas the weight w_0 will provide an additive constant to the board value.

1.2.4 Choosing a Function Approximation Algorithm

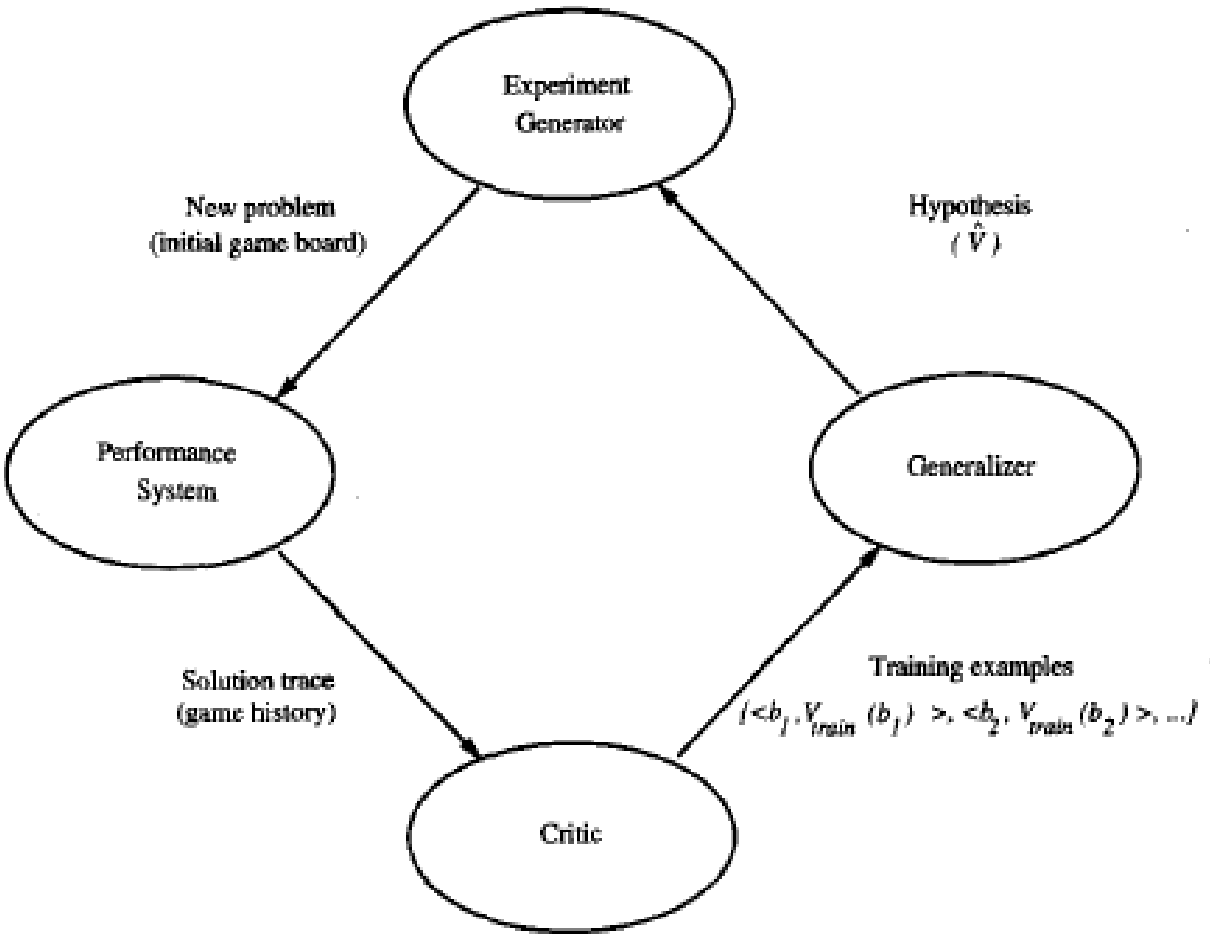
In order to learn the target function $\hat{V}$ we require a set of training examples, each describing a specific board state b and the training value $V_{train}(b)$ for b. In other words, each training example is an ordered pair of the form $\langle b, V_{train}(b) \rangle$. For instance, the following training example describes a board state b in which black has won the game (note $x_2 = 0$ indicates that red has no remaining pieces) and for which the target function value $V_{train}(b)$ is therefore +100.

$$\langle \langle x_1 = 3, x_2 = 0, x_3 = 1, x_4 = 0, x_5 = 0, x_6 = 0 \rangle, +100 \rangle$$

This involves a procedure that first derives training examples from the indirect training experience available to the learner, then adjusts the weights *wi to best fit these training examples.*

1.2.5 The Final Design

The final design of checkers learning system can be described by four distinct program modules that represent the central components in many learning systems. These four modules, summarized in the figure are as follows:



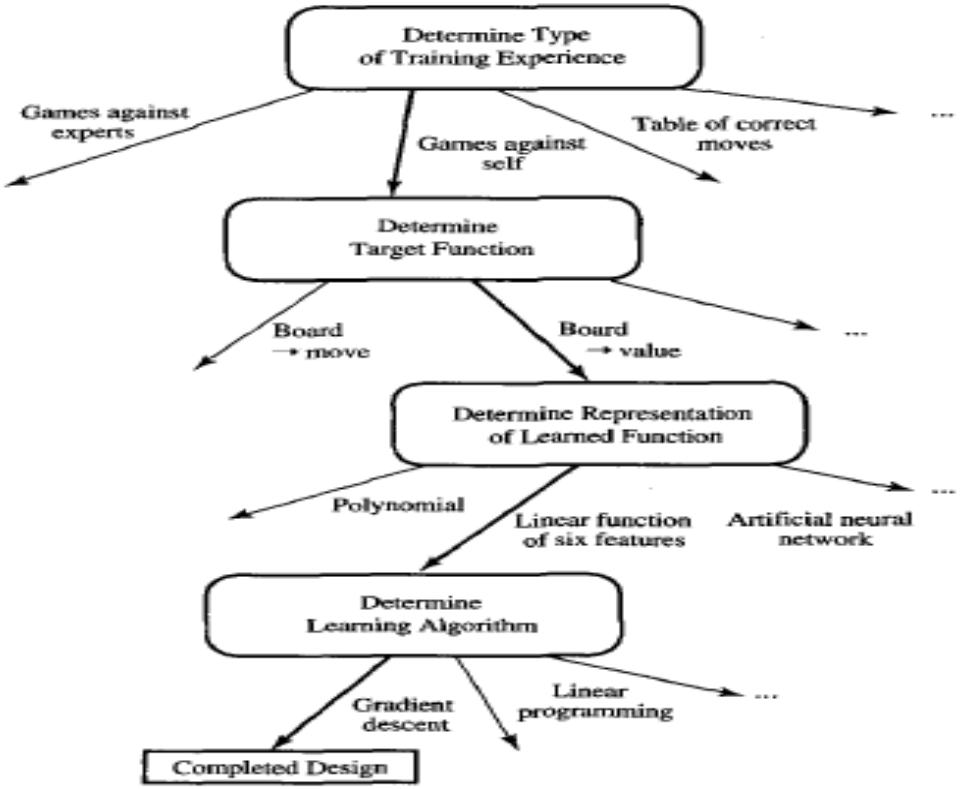
• The Performance System is the module that must solve the given performance task, in this case playing checkers, by using the learned target function(s). It takes an instance of a new problem (new game) as input and produces a trace of its solution (game history) as output. In our case, the strategy used by the Performance System to select its next move at each step is determined by the learned p evaluation function. Therefore, we expect its performance to improve as this evaluation function becomes increasingly accurate.

• The Critic takes as input the history or trace of the game and produces as output a set of training examples of the target function. As shown in the diagram, each training example in this case corresponds to some game state in the trace, along with an estimate *Vtrain of the target function value for this* example.

• The Generalizer takes as input the training examples and produces an output hypothesis that is its estimate of the target function. It generalizes from the specific training examples, hypothesizing a general function that covers these examples and other cases beyond the training examples. In our example, the Generalizer corresponds to the LMS algorithm, and the output hypothesis is the function f described by the learned weights *wo, . . . , w6.*

•The Experiment Generator takes as input the current hypothesis (currently learned function) and outputs a new problem (i.e., initial board state) for the Performance System to explore. Its role is to pick new practice problems that will maximize the learning rate of the overall system. In our example, the Experiment Generator follows a very simple strategy: It always proposes the same initial game board to begin a new game. More sophisticated strategies could involve creating board positions designed to explore particular regions of the state space.

The sequence of design choices made for the checkers program is summarized in Figure**.**



1.3 PERSPECTIVES AND ISSUES IN MACHINE LEARNING

One useful perspective on machine learning is that it involves searching a very large space of possible hypotheses to determine one that best fits the observed data and any prior knowledge held by the learner. For example, consider the space of hypotheses that could in principle be output by the above checkers learner. This hypothesis space consists of all evaluation functions that can be represented by some choice of values for the weights *wo t*hrough *w6. The learner's task is thus to* search through this vast space to locate the hypothesis that is most consistent with the available training examples.

The LMS algorithm for fitting weights achieves this goal by iteratively tuning the weights, adding a correction to each weight each time the hypothesized evaluation function predicts a value that differs from the training value. This algorithm works well when the hypothesis representation considered by the learner defines a continuously parameterized space of potential hypotheses.

Issues in Machine Learning

Our checkers example raises a number of generic questions about machine learning. The field of machine learning, is concerned with answering questions such as the following:

- What algorithms exist for learning general target functions from specific training examples? In what settings will particular algorithms converge to the desired function, given sufficient training data? Which algorithms perform best for which types of problems and representations?
- How much training data is sufficient? What general bounds can be found to relate the confidence in learned hypotheses to the amount of training experience and the character of the learner's hypothesis space?
- When and how can prior knowledge held by the learner guide the process of generalizing from examples? Can prior knowledge be helpful even when it is only approximately correct?
- What is the best strategy for choosing a useful next training experience, and how does the choice of this strategy alter the complexity of the learning problem?
- What is the best way to reduce the learning task to one or more function approximation problems? Put another way, what specific functions should the system attempt to learn? Can this process itself be automated?
- How can the learner automatically alter its representation to improve its ability to represent and learn the target function?