

Machine learning has developed over the years. Right from simple rote learning we have used different paradigms of machine learning to handle some very complex learning tasks. It has converted many so-called fictions into reality. Traditionally machine learning was confined to acquiring more information and retrieving the information timely. It was more like information acquisition to information retrieval. Let us take an example of a student who claims that he learned deep learning. What does that mean? Does he understand the concept or just few terminologies? Can he apply these concepts? Can he expand these concepts?

Am I getting confused between studying and learning? Learning should allow individuals to apply these concepts beyond conceptual space, coming up with allied concepts, and even transforming these concepts into a new form. Learning is actually expanding the ability to produce the results. In this case, even if student has studied the concept of deep learning and if it does not expand his abilities to produce the results, or solve some problems, which he could not solve, previously then it is not learning. Traditional paradigms of machine learning are typically based on knowledge acquisition. In case of student, he has acquired some knowledge and can reproduce same when needed. Next phase is he can apply it to the known scenario. Knowledge acquisition techniques evolve in different forms. In case of an answering bot, information is acquired by the bot in the form of answers and produced in case of appropriate questions. Most of the learning are involved in mapping these questions and answers.

In this chapter, we will discuss different paradigms of machine learning. While discussing different paradigms, we will introduce the new paradigm of creative machine learning. The knowledge innovation goes beyond acquisition. *Finding more through available knowledge association, meta-reasoning, exploration, expansion, transformation, and mapping*

Paradigm: Paradigm is a typical supposition about some product, situation, or solution. It can even be a typical example of pattern or model. In the following, we will discuss different learning paradigms with examples.

1. *Memorization paradigm* or rote learning was one of the most popular approaches where machine could produce exact result from repository. It has complete focus on storage and retrieval. This method was about exact problem mapping and selection of answers. It is a simple event-based retrieval and direct mapping. Simple answering bot is an example of this paradigm.
2. This evolved to *event-based machine learning*. We learn based on events in the past. Event-based learning typically refers to one or more reference events in the past. These events and their outcomes are used for learning. Event-based learning has very limited sample size and hence sometimes leads to surprising results. A focused event is at center in event-based learning.
  - Let  $e_1$  be an event in focus
  - Event is represented in the form of cause and effect relationship
  - If A happens then event  $e_1$  takes place and that is responsible for the outcome
  - The hypothesis is based on a single event.

Event  $e_1$  is represented as sets of input and output parameters

$$[i_1, i_2, \dots, i_n] \rightarrow [o_1, o_2, \dots, o_n]$$

3. Pattern-based learning is based on patterns based on historical series of events. These patterns help to learn about possible consequences. This is based on dominant patterns and their mapping to outcome. These patterns are classified based on properties in more than one classes. It is based on consistent and repeated occurrence of event resulting in same outcome. It boils down to effective relationship between variables. There can be a visible relationship among series of conclusive events occurring repeatedly. Pattern-based learning does not learn from noise but rather minimize noise with statistical averaging. Time series analysis and data stream analysis come under this supposition. Simple time series analysis comes up with forecasting for the next event. Let  $P_1, P_2, P_3, P_4, \dots, P_m$  be the time series associated with week  $W_1, W_2, W_3, \dots, W_n$

Moving average caters to creating series of averages of different subsets of given datasets. The subset size can be defined as number of entities. Now moving average defines the number of entities. Let  $m$  be the number of entities then moving average for  $r$ th element is the average of elements  $r$  and  $m - 1$  elements prior to  $r$ th element. Moving average removes the noise and can help in getting better forecast. Further learning in time series example, general trend, cyclic trend, DOW pattern index, Seasonal trend is also considered. The time-series-based data can be used for learning in association with other relevant parameters.

Code for time series forecasting is given below to highlight its typical input-output mapping and data-centric approach.

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.linalg as lin
import re

dataset=np.array(map(lambda l: map(float,filter(lambda singleSequence:
len(singleSequence)>0,re.split('\s+',l))),open('chpf_ts_data.txt')).T)
dataK=230
pattern=dataset[1,:dataK]
dataset=dataset[1,dataK:]

eta=10.
lam=.1
sequences=1
dimensions=2
sequencesLength=len(dataK)
singleSequence=np.ones( (sequencesLength+1,dimensions) )
singleSequence[0,1]=1
singleSequence[1,0]=dataK

def initMatrix(dataK):
    granularityStep=10
    eps=.1
    dataK=dataK[:granularityStep]
    hiddenStates=len(dataK)+1
    transitionMatrix=np.zeros( (hiddenStates,hiddenStates) )
    transitionMatrix[0,1]=1.
    roughCountsForTransitionMatrix=np.zeros( (hiddenStates,hiddenStates) )
    roughCountsForTransitionMatrix[0,1]=1.
    for i in xrange(1,hiddenStates-1):
        transitionMatrix[i,i]=(granularityStep-1.*eps)/(granularityStep*2*eps)
        transitionMatrix[i,i+1]=(1.*eps)/(granularityStep*2*eps)
        roughCountsForTransitionMatrix[i,i]=1.
        roughCountsForTransitionMatrix[i,i+1]=1.
        transitionMatrix[-1,-1]=(granularityStep-1.*eps)/(granularityStep*2*eps)
        transitionMatrix[-1,1]=(1.*eps)/(granularityStep*2*eps)
        roughCountsForTransitionMatrix[-1,-1]=1.
        roughCountsForTransitionMatrix[-1,1]=1.

    regressionWeights=np.ones( (hiddenStates,2) , dtype=np.float)
    regressionWeights[0,1]=dataK[0]
    regressionWeights[1:-1,1]=(dataK[1:-1]-dataK[1:])/granularityStep
    regressionWeights[-1,1]=(dataK[0]-dataK[-1])/granularityStep

    return transitionMatrix,roughCountsForTransitionMatrix,regressionWeights,hiddenStates

def expectedK(matrix):
    global transitionMatrix,regressionWeights,eta
    global forwardRecurrence,backwardRecurrences,emissions,residuals
    global expectedHiddenStates,expectedPairOfHiddenStates,nkk

    singleSequence=matrix[:1]
    y=matrix[1:,1]
    residuals=np.dot(singleSequence,regressionWeights.T) # (sequences,hiddenStates) <- (sequences,1) (sequences,hiddenStates)
    residuals=y
    emissions=np.exp(-eta/2*residuals**2,emissions)

    forwardRecurrence(matrix)
    backwardRecurrence(matrix)

    for t in xrange(len(emissions)):
        expectedHiddenStates[t,:]=forwardRecurrence[t+1,:]*backwardRecurrences[t+1,:]
        expectedHiddenStates[t,:]/=np.sum(expectedHiddenStates[t,:])
```

A bot learning based on time series patterns or data patterns is an example of pattern based learning.

4. The *behavioral patterns* across learning space are used at the next level of learning making the problem of learning a bit complicated. The behavioral pattern is specific to objects and is a depiction of its interaction with environment and change in state with reference to environment. It can be fairly dynamic. It is determined based on parameters of object and state transitions. Bots working on behavioral patterns come under this paradigm.

```

for t in xrange(len(forwardReurrence)-1):
    expectedPairOfHiddenStates=transitionMatrix*forwardReurrence[t,:][:,nmp.newaxis]*emissions[t,:]*backwardReurrences[t+1,:]
    expectedPairOfHiddenStates/=nmp.sum(expectedPairOfHiddenStates)
    nkk=expectedPairOfHiddenStates

transitionMatrix=roughCountsForTransitionMatrix+nkk
transitionMatrix/=nmp.sum(transitionMatrix,1)[:,nmp.newaxis]

for k in xrange(hiddenStates):
    ex=expectedHiddenStates[:,k][:,nmp.newaxis]
    dx=nmp.dot(singleSequence.T,ex*singleSequence)
    dy=nmp.dot(singleSequence.T,ex*y)
    dy.shape=(2)
    regressionWeights[k,:]=lin.solve(dx+lamb*nmp.eye(singleSequence.shape[1]), dy)

return forwardReurrenceIs[-1]

if __name__ == '__main__':
    #EM algorithm
    for i in xrange(20):
        print expectedM(singleSequence)

    roughBoundari=nmp.arange(len(expectedHiddenStates))[nmp.argmax(expectedHiddenStates,2)<4]

    #plot
    plt.plot(range(sequenceLength),singleSequence[1,0])

    yr=[nmp.min(singleSequence[:,0]),nmp.max(singleSequence[:,0])]
    for i in roughBoundari:
        plt.plot([i,i],yr,'-roughBoundari')

    plt.show()

#initialization step
transitionMatrix,roughCountsForTransitionMatrix,regressionWeights,hiddenStates=initMatrix(pattern)
emissions=nmp.zeros( (sequenceLength,hiddenStates) )
residuals=nmp.zeros( (sequenceLength,hiddenStates) )

#store the forward and backward recurrences
forwardReurrence=nmp.zeros( (sequenceLength+1,hiddenStates) )
forwardReurrenceIs=nmp.zeros( (sequenceLength+1) )
forwardReurrence[0,0]=1
backwardReurrences=nmp.zeros( (sequenceLength+1,hiddenStates) )
backwardReurrenceIs=nmp.zeros( (sequenceLength+1) )
backwardReurrences[-1,1]=1/(hiddenStates-1)

hiddenStates=nmp.zeros( (sequenceLength+1),dtype=nmp.int )
expectedHiddenStates=nmp.zeros( (sequenceLength,hiddenStates) )
expectedPairOfHiddenStates=nmp.zeros( (hiddenStates,hiddenStates) )
nkk=nmp.zeros( (hiddenStates,hiddenStates) )

def forwardReurrence(matrix):
    global forwardReurrence,emissions
    for t in xrange(sequenceLength):
        forwardReurrence[t+1,:]=nmp.dot(forwardReurrence[t:],transitionMatrix)*emissions[t,:]
        sm=nmp.sum(forwardReurrence[t+1,:])
        forwardReurrenceIs[t+1]=forwardReurrenceIs[t]+nmp.log(sm)
        forwardReurrence[t+1,:]/=sm
        assert forwardReurrence[t+1,0]==0

def backwardReurrence(matrix):
    global backwardReurrences,emissions
    for t in xrange(sequenceLength-1,-1,-1):
        backwardReurrences[t,:]=nmp.dot(transitionMatrix,backwardReurrences[t+1,:]*emissions[t,:])
        sm=nmp.sum(backwardReurrences[t,:])
        backwardReurrenceIs[t]=backwardReurrenceIs[t+1]+nmp.log(sm)
        backwardReurrences[t,:]/=sm

```

If we take look at all these methods, we realized there is a common thread among them. They are based on one simple principle and that is decoding relationship between input and output. The relationship may come in the form of nonlinear equations, probabilistic mapping, or closeness and association, or simple rules. To decode this relationship, we collect different input samples and corresponding output samples. The input samples may be confined to object and some extended cases may consider environment in which an intelligent agent is deliberating. The mapping between input and output is used for learning. In simple cases, it works based on a series of rules represented using decision tree. In complex cases, it can be featured by vector-based classification. This is a typical supervised learning or learning based on labeled data. In the absence of labeled data, we learn based on similarities and differences to form grouping of data points. This is called as unsupervised learning. While supervised learning has minimal freedom, unsupervised learning gives additional freedom. Unsupervised learning has hint of creativity since there is uncertainty and dynamic clustering can lead to number of different outcomes not defined beforehand. Semi-supervised learning is about

making use of unlabelled data for learning based on availability. The semi-supervised paradigm is very powerful and leads to experiments using different methods and techniques. Active learning is another variant of semi-supervised learning where experts take part in process.

In this chapter, we will cover these paradigms with the intent of locating ML opportunities. There are always a lot of questions in the mind of professionals and researchers—Is this a machine learning problem? Can I use machine learning here? Is it rule-based problem, simple analytics problem, complex analytics problem, or machine learning problem? Which machine learning method I should use?—Practitioners, professionals, and ML consultants are always keen on replacing rule-based systems with machine learning or intelligent system to build an obvious edge over their competitor. Sometimes, it is mandatory for business while sometimes it is need of time. Is this a right way to approach a problem? Shall I begin with existing solution, or use a new method or a new paradigm? Many such questions are at the base of this discussion. We will touch these questions related to machine learning opportunities in this chapter.

In first section, we will take a closer look at it with a few practical examples. While in later sections, we will try to unfold learning strategies and learning model selection practices.

---

## 2.1 Understanding Learning Opportunity (Catching Data Signals Right)

Advent of AI and success of big giants using machine learning motivated everyone to try it out. Big giants like Google, Amazon, and Microsoft decided to lead ML paths and even vouched on making some ML applications and platforms available for businesses. Other companies are looking at this challenge from a completely different perspective. When there is a problem in front of these companies, there are questions like—Is it a ML problem? Shall I replace my present system? Can I get rid of thousands of L1 and L2 customer executives trying to solve customer requests with intelligent systems and ML? Can I do automated reconciliation?

We are trying to put systems in place to solve problems. Sometimes, we are going for advanced and complex methods, sometimes we use machine learning to solve problems to build competitive advantage. I have these questions and request from many clients when I was trying to solve their problem. How can I say that a particular problem is a machine learning problem? Is it simple? Many times we divide these problems into three types

1. Problems that can be solved by rule-based Systems
2. Problems that need data analytics
3. Problems that need machine learning.

Actually all the three types mentioned here are related and in a way use some form of ML. Hence:

Every problem is a ML problem only you need to identify level of learning required!

One of my friends who is a machine learning researcher says—“Actually every problem can be machine learning problem but every problem is not a machine learning problem.”

He wanted to say that you can solve any problem with machine learning techniques and paradigm, but it is not necessary that you should solve it using ML. Any way that is a bit abstract but what is machine learning problem? Can I have test to determine this? Can we analyze to determine what level of intelligence we need to put in? Yes, we can. Any problem can be considered as machine learning problem, if it has one or more of the following properties:

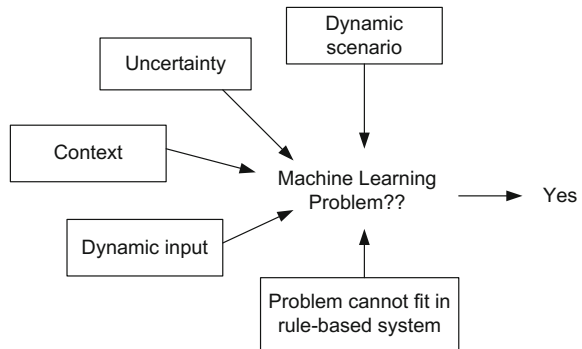
1. When it is dealing with dynamic scenarios: Which means environment is changing continuously and when agent is deliberating.
2. When it is acting in an uncertain environment: When there are certainty rules that are good to decide the direction. Larger uncertainty makes it necessary to keep on learning to solve problems. Simple rules devised in the past may not hold now in new scenarios.
3. The context makes sense but it is not just available and you need inference and association for the same. You need to work hard to build context.
4. Input is dynamic and you need to improve performance on every outing.
5. Simple rules fail to manage the overall scenario and you can sense their limitations.

These thumb rules help us while deciding approach to solve the problems. Figure 2.1 depicts machine learning aspects of the same.

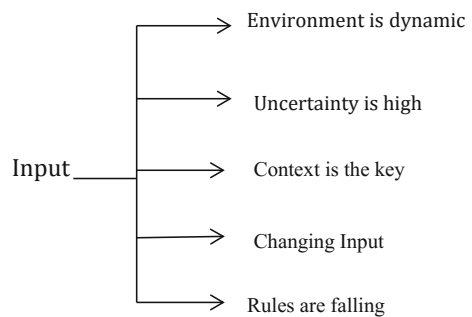
These properties are true for many problems. But one of the candidates attending my ML workshop asked me—“If environment is static then can I say—it is not machine learning problem? Do you mean then I cannot learn in static environment?”

Well, you can learn in any environment and that is why every problem can be ML problem. But in known space with known options, one can get decent performance with rule-based systems. Rule-based systems are typically not learning systems but mapping systems. Even rule-based system is a sort of simple intelligent system. Based on simple mapping, you can determine what action to take. Anyway to form the rules you have to learn. These rules are devised by experts and hence there is limited learning by machines. If you build a system where based on sample space system builds its own rules—that definitely qualify to ML. The rules can be determined by system and evolve to next set of rules. As the evolution is taking

**Fig. 2.1** Machine learning problem analysis



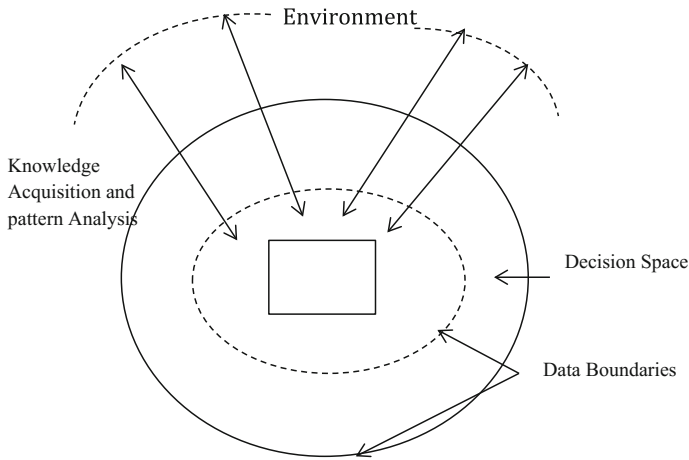
**Fig. 2.2** Thumb rules for machine learning



place yesterday's simple rule-based system may not qualify as machine learning systems. In a way that is correct. But can I have a systematic way to determine? Figure 2.2 depicts criterion and thumb rules for determining ML problems. We will further discuss a few models in subsequent sections.

Python code for decision-making is given below:

```
# Import Library
# Import other necessary libraries like pandas, numpy...
from sklearn import tree
# Considering you have predictor and Y (target) for training data set and x_test(predictor) of test_dataset
model = tree.DecisionTreeClassifier(criterion='info gain')
model.fit(predictor, target)
model.score(predictor, target)
predicted = model.predict(testPredictor)
```



**Fig. 2.3** Limited exploratory paradigm systems

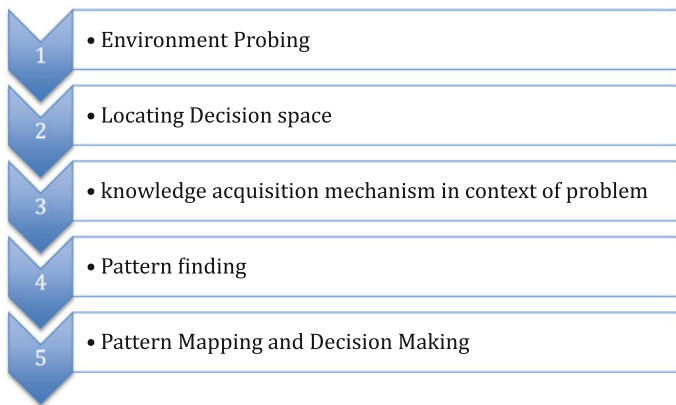
## 2.2 Knowledge Innovation Building Blocks of ML and Intelligent Systems

Learning systems are traditionally built around knowledge acquisition and data mining. It looks wonderfully great and these systems delivered wonderful results over the years, these systems are built from different paradigm. The paradigm was to do what human can do and produce results. It was typically looking at human's input-output patterns. Sometimes, the results for learning are provided by other experts while in other cases human efforts are taken to build human behavior patterns. If you look at these machines, they are built with *limited exploration paradigm*. These systems tried to mimic human mapping of decision-making to knowledge repositories. Hence, there was stress on data mining. Finding the patterns out of existing data was at the center of all these activities. Exploration paradigm was also built in context and strongly based on *limited adventure modes*. Here limited adventure mode generally helps incremental learning. Again that goes to paradigm of knowledge acquisition. If we look at the overall system, knowledge acquisition module is at heart of the system. A generic representation of these limited exploration paradigm (LEP) systems is depicted in Fig. 2.3.

## 2.3 Stages in Limited Exploration

Stages in data exploration are depicted in Fig. 2.4.





**Fig. 2.4** Stages in data exploration

Environment probing looks for environmental changes and parameters and senses them. Locating decision space is determined based on environment probing. In context of problem, knowledge is acquired and that is used for identifying patterns and building pattern repository. Pattern mapping is used for decision-making.

While mapping knowledge innovation paradigm and creative learning, the focus is on learning process. The creative learning is transformation from knowledge acquisition mode to knowledge innovation mode. While knowledge acquisition is about agglomeration—innovation is nonlinear transformation.

In any traditional learning methods, there is association among inputs and outputs. The relationship between input and output is represented using equations or statistical parameters. This formation of relationship drives machine learning model. The ML models are classified in to various types:

#### – Binary Versus Multiclass

Binary classification tries to classify the sample space into two classes. Classifier is trained for two distinctive classes. The features of the objects are used for this classification. A simple example is classifying food items into fruits and non-fruits items. Here, classification is based on features which are the distinguishing properties of the item. Binary classification algorithms with multiple iterations with different feature vectors can be used for multiclass classification. Here sample space is classified into more than two classes. A simple SVM classifier is given below:

```
#Import Library
from sklearn import svm
model = svm.svc()
model.fit(predictor, target)
model.score(predictor, target)
predicted = model.predict(testPredictor)
```

In case of learning bot for banks, the customers can be classified customer looking for opening bank accounts and customers looking for bank loan.

#### – Classification Versus Prediction

Classification and predictions are two different ways of looking at a problem; classification can be used for prediction. While classification predicts categorical labels, prediction tries to predict missing values or parameters. In some cases, these terms are also used for prediction values and predicting class labels and can be viewed as complimentary. Prediction looks for trends and cycles and classification looks for distinct properties and behavior of classes. But identification of classes based on these trends can help in prediction.

## 2.4 Mathematical Equations for Classification

Classification can be represented based on parameters and feature vectors. A typical feature vector (FV) may look like:

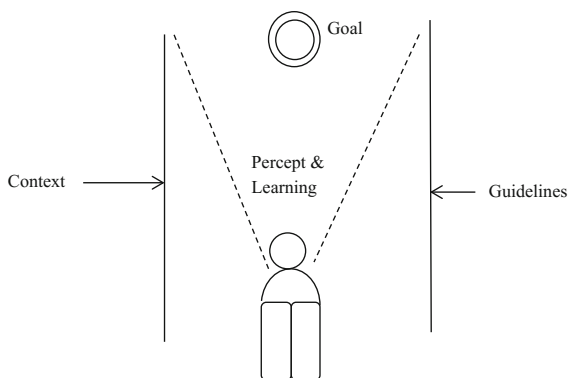
$$FV = W_1F_1 + W_2F_2 + \dots + W_nF_n$$

Let  $C_1$  and  $C_2$  be two classes and FV is mapped to classes based on association.

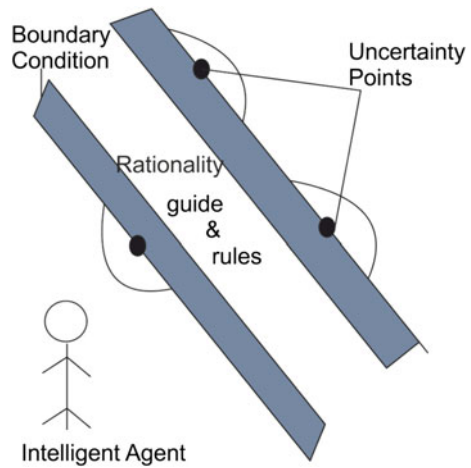
It is always assumed that intelligent agent is a rational agent and abides to rational guidelines. It is fairly correct but when it comes to creativity those are rather guidelines. Creativity requires timely suspension of rational principles. Hence it is not rules but guidelines. Knowledge acquisition based systems create rules—the rules may come in the form of decision systems standing for crisp rules, fuzzy systems looking for fuzzy rules. These systems are typically afraid of boundary conditions. Actually, boundary conditions and overlaps are opportunities to come with creative learning. Figure 2.5 depicts these guidelines with reference to rational agent.

Based on needs of problem there are various learning methods. There are boundary conditions alongside with guidelines. These boundary regions and uncertainty points are depicted in Fig. 2.6. In case of bot, these boundary conditions

**Fig. 2.5** Rationality and irrationality in intelligent agents



**Fig. 2.6** Rationality and boundary conditions

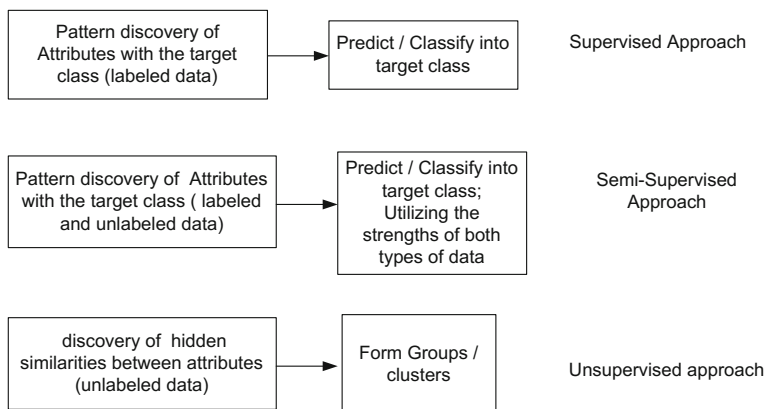


can be questions asked in different context by user or even a few unanswerable questions are raised.

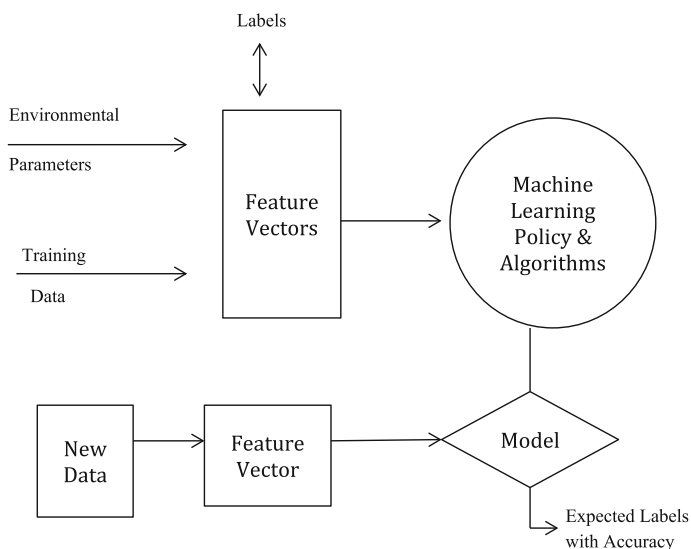
- Supervised versus unsupervised: As discussed in previous section, supervised machine learning is learning based on labeled data while unsupervised learning is learning based on unlabeled data. When we make use of both labeled as well as unlabeled data, it is called as semi-supervised learning. In supervised learning, a set of examples is provided to intelligent agent for learning. Based on these examples, a model is built to accommodate learning examples to produce desired results. Semi-supervised learning looks for bringing the unlabeled data in learn set. Figure 2.7 shows the learning approaches—Supervised, semi-supervised and unsupervised.

Figure 2.8 depicts typical supervised learning while Fig. 2.9 depicts semi-supervised learning.

- Active versus passive: Passive supervised learning is one where using data set and based on supervisor guidance, intelligent agent learns to classify or deliberate. Here, all unlabeled examples are verified by experts and are used for passive supervised learning. In semi-supervised passive learning, algorithm helps agent/algorithm to select relevant learning data from unlabeled samples. This is done after expert has given enough samples for initial level of learning. Active semi-supervised machine learning is next step to passive semi-supervised machine learning. Here, agent actively takes part in learning process. When learner or agent comes across unlabeled data which he has limited knowledge, it requests label from expert. An algorithm is built based on labels generated by experts to expand boundaries of classification. The active learning is better than passive learning in many ways—first of all it has fewer labels than that of passive learning. Further, it can be probing and can improve accuracy. It has flavor of evolution and active probing initiated by machine. Figure 2.10 depicts concept of semi-supervised learning.



**Fig. 2.7** Supervised, semi-supervised, and unsupervised learning



**Fig. 2.8** Supervised learning

Let labeled data be  $L_1, L_2, L_3 \dots L_n$

Classifier model  $M$  Input  $\{I_1, I_2, I_3 \dots I_n\} \rightarrow$  Output  $\{O_1, O_2, O_3 \dots O_n / \text{Labels } L_1, L_2, L_3, \dots L_n\}$

New Example  $NE_1 \rightarrow$  System.

Figure 2.11a, b depict the semi-supervised passive and active learning.

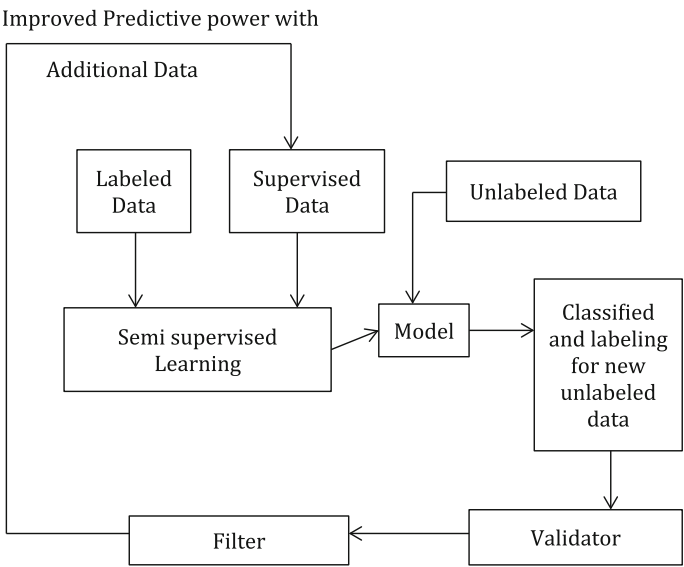


Fig. 2.9 Semi-supervised learning

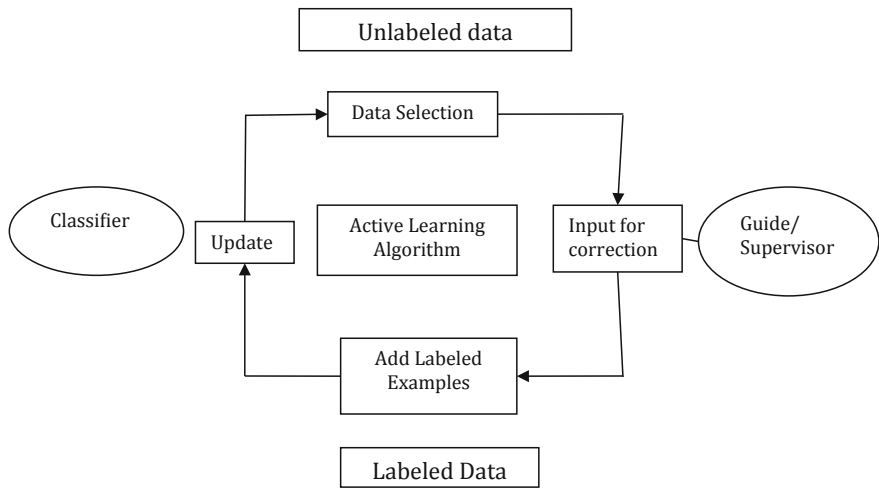
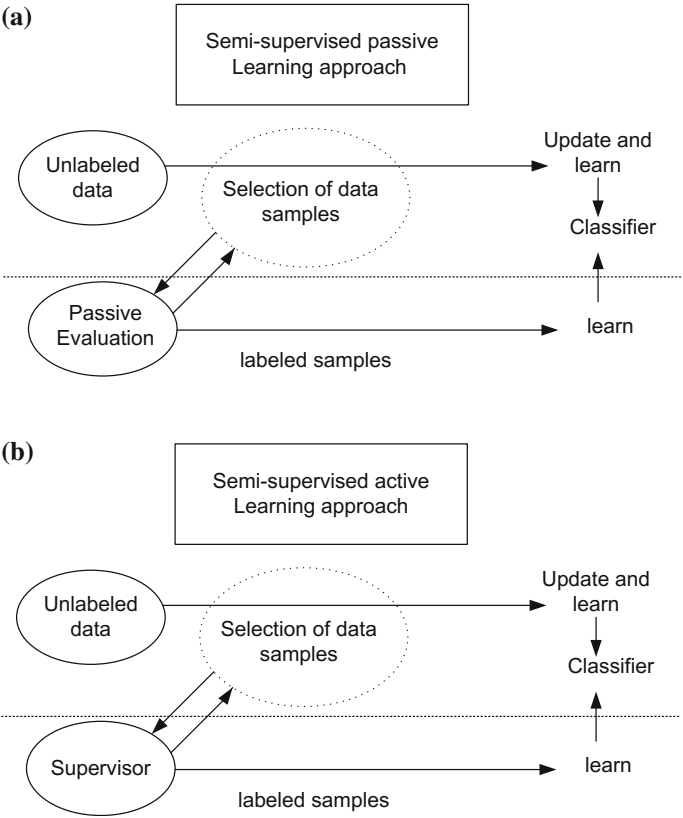


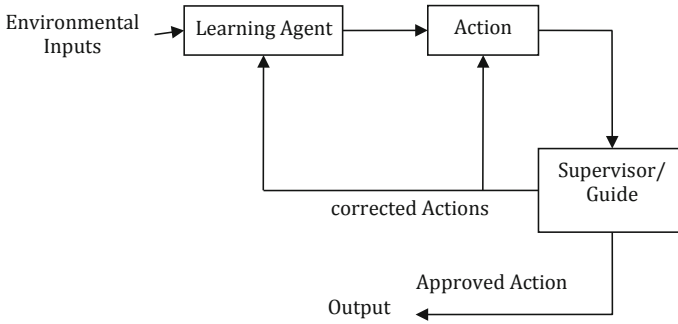
Fig. 2.10 Concept of semi-supervised learning

- Closely guided versus remotely observed learning: Closely guided learning is one where every activity and instance are watched very closely. It does look for micro details and gets those investigated. It tries to correct every mistake immediately. Knowledge acquisition based learning is typically closely guided



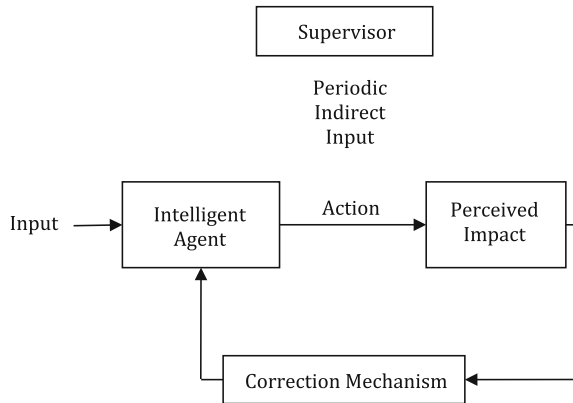
**Fig. 2.11** a Semi-supervised passive learning. b Semi-supervised active learning

learning. Remotely observed learning allows the agent to make mistake and based on overall results observed over the time suggests the corrections. Closely guided learning is very good for short-term results while remotely observed learning works wonderfully for long-term results. Most of the exploratory learning algorithms come under remotely observed learning. Remotely observed learning allows agent to take decisions and look for own mistakes and improvements. It creates opportunity to enhance algorithm. The remotely observed learning can be combined with reinforcement learning. Here, time period of remote observation is set based on application. A typical temporal difference learning is an example of such learning. Even there are a few possible combinations of remotely observed learning with closely monitored learning and both together can be used for improvement of opportunities in creativity. Figure 2.12 depicts closely observed learning and Fig. 2.13 depicts remotely observed learning.



**Fig. 2.12** Closely observed learning

**Fig. 2.13** Remotely observed learning



- Systemic learning versus analytical learning: Learning in isolation has its own issues. In the absence of a complete picture, it learns in bits and pieces. It does not identify the system and give a complete picture. System is typically a set of interconnected objects that exhibit togetherness. In the absence of systemic picture, the fragmented learning tries to fix problems locally or based on limited knowledge without understanding aftereffects.

Learning in isolation suffers from systemic effects in the long run. Event-based and pattern-based learning works well for some set of problems but when it comes to complex, interdependent and associated problems we need to think, act, or rather learn systemically. Systemic learning is about learning with reference to intra-dependencies and inter-dependencies in the subsystems. It is also about knowing that cause and effects might be separated in time and space. Analytical learning is a very popular way of learning that deals with analytical recommendations. It enables to understand parts of the situation and focuses on detailed analysis of small part while synthetical thinking helps us to understand how

different parts work together. Analytical learning breaks things down into their component and parts; sometimes even in very small and minute parts where as synthetical learning finds the patterns, association and togetherness across those component parts. Hence, synthetical learning is more associative while analytical learning is dissociative. Analysis stresses on differences and synthesis on similarities. Synthetical thinking is about interactions and hence dynamic in nature and allows to cope with changes in system adaptively. Systemic learning is a combination of analytical learning and synthetical learning. It tries to look at problem from associative perspective with stress on relatedness.

- Holistic versus region specific: Holistic is about completeness. Though holistic is systemic it is more about complete information and not about relatedness among subsystems. Region specific is localized. There is difference between holistic and systemic learning—while holistic learning tries to acquire maximum information, systemic learning is more focused on system intricacies.
- Adaptive versus generalized learning: A particular way of learning may be good on a particular dataset. But it may not yield encouraging results in case of another data set. Even this is true in case of different problems. We learn in different way when dealing with language subjects and learn differently in case of mathematical subjects. Generalized learning tries to capture general properties of learning irrespective of dataset and tries to come up with a single solution across the problems. Adaptive learning tries to adapt to given data set and scenario and based on that dynamically decides the learning policies. Systemic learning should be adaptive and dynamic and should be able to cater to subsystems.
- Context-based versus multi-context learning: Context-based learning is learning based on context and is the special case of adaptive learning. Context is about understanding background, situations, and scenarios. It can include locations, region, time, date, and entities. A student trying to learn mathematics is the basic scenario. But the student is a boy, he is from Pune, he has already cleared National Mathematics Olympiad, his age is 17, and he loves probability. All these parameters form the context.

In case of an example of bot—customer is from Mumbai, he or she is of age 50 with diabetes and looking for long-term medical insurance, this forms the context. This context allows learning and identifying the best solution. Context-aware systems and context-based learning allows to take the best possible learning and decision-making with reference to context.

The problem is not limited to a single context to deal with. Same problem may arise in multiple contexts. Multi-context learning tries to cluster similar contexts and tries to come up with a common policy for the selected contexts. This can help in identifying similarity among multiple contexts and helps in building holistic view on one side while keeping track of various contexts on the other side.

- Online versus batch learning: Online learning is based on continuous data streams in real time. Batch learning typically collects data in batches and learns based on those batches. Batch learning is more data intensive while online needs to be more explorative.



## 2.5 New Paradigms in This Book

Machine learning evolved from storage to timely retrieval, crisp to fuzzy, single layered to multilayered, and exploitation to exploration. The new paradigm introduced and discussed in this book is focused on creativity and looking beyond available data.

- Knowledge acquisition based learning versus Knowledge innovation based learning: Knowledge acquisition paradigm looks for building knowledge base and retrieval of required knowledge. Knowledge innovation paradigm is building new knowledge from the acquired knowledge. It is also about meta-knowledge mapping.
- Pattern-based learning versus creative ML: Pattern-based looks for pattern in past and stores patterns in repository. Creative ML tries to go beyond patterns. It looks for uncertainty and freedom points.
- Forward hypothesis versus reverse hypothesis learning: This book introduces new paradigm of *Reverse Hypothesis Machines*. While ML is striving for Forward Hypothesis based on mapping, reverse hypothesis tries to decode mapping, relationships, and use uncertainty points for exploring new pathways.
- Data volume driven learning versus optimal learning: Data Volume—well, big data is buzzword. The big data paradigm is about Volume, Velocity, Veracity, Variety and many other ‘V’s’. Optimal learning on other side tries to minimize data and learns optimally. It has three Os—Open, Optimal, and Original. Open is about freedom and uncertainty, Optimal is about not more and not less, and Original is about novelty and not directly driven from data.

How Aqua Chill used traditional ML in nontraditional way:

### Systemic Knowledge Innovators: Aqua Chill—Gajanan Khot

It was the time when big giants—big name banner companies—were holding their stake in the air conditioning industry and there was hardly any space left for others. Two friends—now partners who made their mark in the air conditioning business—had faced tremendous challenge that time to achieve what they have today.

One of the partners could understand and feel the necessity of having “an intelligent cooling” for them to probe in the market. He acquired the knowledge, understood the market need from energy consumption to power reduction before they could proceed.

They converted their customer into knowledge partners to achieve this.

There were big companies who were about to start their plant in India; they looked at working with existing established companies for air conditioning. The partners convinced them and got their first opportunity.

But it was not a cakewalk. They put to practice their knowledge of ML to come up with an economical and custom-made solution. They analyzed the working pattern of the employees, their other working habits to devise an intelligent cooler. The ML approach—supervised one helped them to save energy while offering effective cooling. The pattern discovery process assisted them to a great deal to offer solutions to the problems at hand.

So, it is not just technology that they put to practice but innovating with the dynamic scenario and changes depending on the working environment. There was uncertainty involved—hence creativity!!!

They slowly went ahead of time to innovate, grabbing innovation awards and further innovating new intelligent systems.

On similar ways, they were able to capture many renowned companies contact simply by—understanding the environment, adequate market study, analysis of behavior pattern, the people, and most important, the risk factor too!!

They accepted new challenges where, there was power cut issues, and where companies did not want water logging due to water tower. They could effectively overcome this problem with intelligent solutions—ML!!

They understood the need to innovate, and put into work to come up with knowledge-based intelligent systems...

With sheer systemic knowledge innovation and associative knowledge building Aqua Chill has become one of the major knowledge organizations. They innovated knowledge and expanded knowledge boundaries to create knowledge value.

---

## 2.6 iknowlation's IDEA Matrix for Machine Learning Opportunity Evaluation

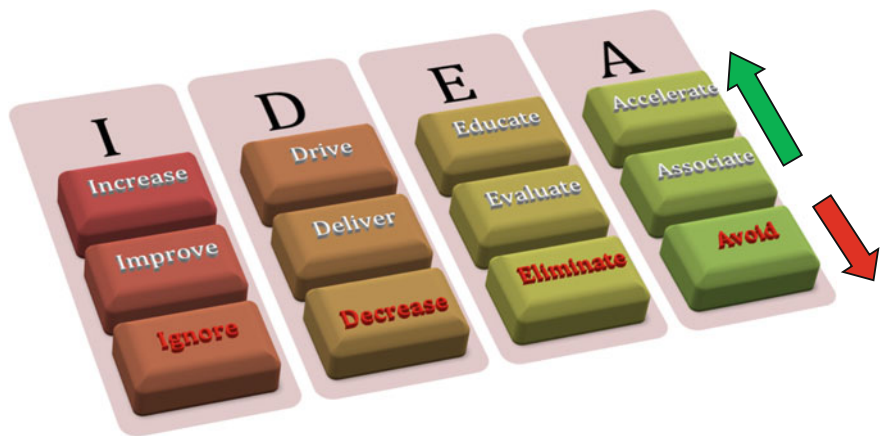
Evaluation of learning opportunity remains a challenge. There are many problems and one needs to evaluate ML opportunities. It is applicable for learning as well as problems. What machine learning tries to achieve? Machine learning tries to improve learnability and build ability in machine to deal with new, uncertain scenarios along with regular situation. This improves a few abilities in machine while optimally using the data ignores a few aspects. In exploration, it comes to evaluate new scenarios and hence it needs to evaluate continuously.

Learning has two aspects:

It improves and increases capabilities that are necessary while it decreases and deletes pointers that are unnecessary. To measure these two aspects in systematic way, we have introduced Learning IDEA Matrix.

A solid framework for learning opportunity evaluation and knowledge innovation is required otherwise locating ML opportunity would become more difficult. To build and practice new paradigms of machine learning with Knowledge Innovation IDEA Matrix framework is developed. This framework is applied to more than one dozen successful ML projects in different domains to locate ML opportunities. This includes industries from financial advisories, agriculture, teaching learning, and health care. This section introduces this learning IDEA framework for systemic evaluation of problem to identify and evaluate ML opportunities. This framework focuses on highest leverage points of knowledge building and knowledge flow optimization to locate and evaluate machine learning opportunities. The framework is based on flexibility and simplicity while applying the knowledge concepts. Learning IDEA matrix is based on learning experiments. Here idea is that one should identify opportunities to evaluate and improve learnability and effectively the over all performance. Idea matrix helps to identify learning problems.

Figure 2.14 depicts IDEA framework for machine learning and systemic knowledge innovation. It looks for different parameters marked by *I*, *D*, *E*, and *A* depicting impact and need of machine learning.



**Fig. 2.14** IDEA framework. Ref: Book by Parag Kulkarni—“Knowledge Innovation Strategy” Bloomsbury 2015

**I (Innovate)**

Increase: Opportunity to increase knowledge and insights with reference to problem (Knowledge innovation opportunity)

Improve: Identifying the need of transition from present state to improved knowledge state (Learnability Opportunity)

Ignore: Opportunity to ignore data points and coming up with relevant data and knowledge pointers through learning (Optimal Learning Opportunity).

**D (Differentiate)**

Drive: Opportunity to drive systemic understanding to create holistic picture through learning (Systemic Opportunity)

Deliver: Opportunity to deliver knowledge value through exploration—  
Deliver improvement through learning (Exploration Opportunity)

Decrease: Opportunity to decrease what is additional (Crisp).

**E (Expand)**

Educate: Opportunity to educate other intelligent agents in the system and creating learning opportunities (Cooperative- and scenario-based learning)

Evaluate: Opportunity and means to evaluate learnability and reward collection (Deep Exploration)

Eliminate: Opportunity to eliminate using crisp rules, and identifying necessary, and unnecessary through learning (Freedom and Uncertainty).

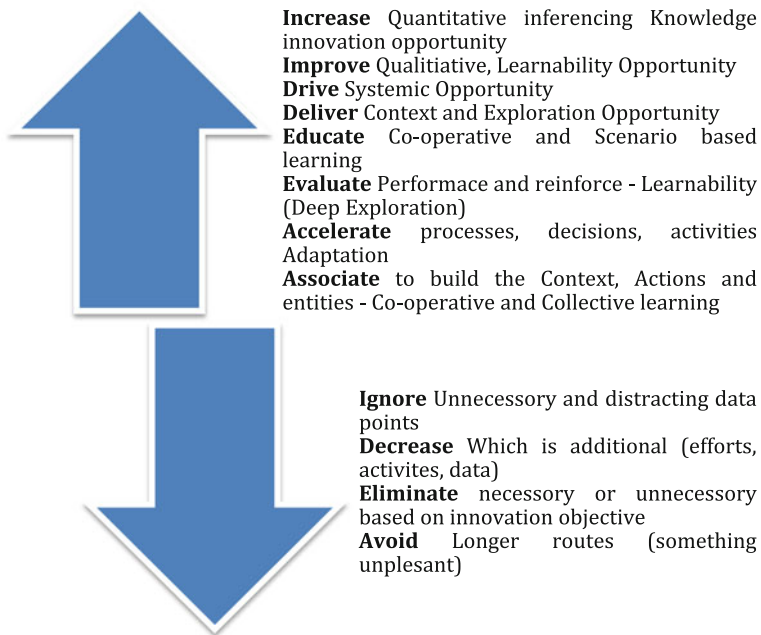
**A (Associate)**

Accelerate: Opportunity to accelerate transitions through adaptability and learning (Adaptive Learning)

Associate: Opportunity to associate different systems, intelligent agents, and knowledge opportunities (Cooperative and Collective learning)

Avoid: Opportunity to avoid the guided routes, extra context (Creative Learning).

Figure 2.15 depicts IDEA matrix learning significance.



**Fig. 2.15** IDEA matrix learning significance

## 2.7 Using IDEA Matrix to Identify ML Opportunity

Machine learning opportunity function:

$$O = f\{I, D, E, A\}$$

Here,  $O$  is Opportunity function. Factors  $I$ ,  $D$ ,  $E$ , and  $A$  can be calculated through simple weighted summation across the system.  $I$  can be represented like:

$$I = \sum_{i=1}^n \text{increase} + \sum_{i=1}^n \text{improve} + \sum_{i=1}^n \text{ignore}$$

Similarly,  $D$  can be represented as:

$$D = \sum_{i=1}^n \text{drive} + \sum_{i=1}^n \text{deliver} + \sum_{i=1}^n \text{decrease}$$

$E$  can work in a similar fashion.

$$E = \sum_{i=1}^n \text{Educate} + \sum_{i=1}^n \text{ievaluate} + \sum_{i=1}^n \text{ieliminate}$$

A is represented as

$$A = \sum_{i=1}^n \text{acceleerate} + \sum_{i=1}^n \text{associate} + \sum_{i=1}^n \text{avoid}$$

Total IDEA value further used to calculate effective opportunity index.

1. General learning from observation and using it in some problem scenarios when required
2. Another aspect is learning with reference to solving a particular problem  
Is it possible to combine these two aspects? In case possible how can I do that?
3. These parameters help to understand learning opportunities.

Learning opportunity index is function of {IDEA}

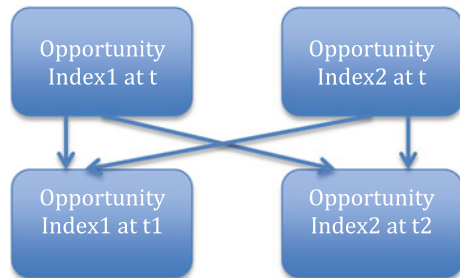
Let  $U$  represent freedom or uncertainty then opportunity index can be depicted as:

$$O = \sum_{t=k}^T \binom{T}{k} U^k \{I, D, E, A\}^{T-k}$$

Opportunity index can be calculated using effective crowd sourcing from stakeholders. This further helps to build ML opportunity graph. Since no opportunity is absolute. There is association among number of such opportunities. Since the association is typically directed it can be represented a directed acyclic graph (DAG) in Fig. 2.16.

The relative temporal relationships among machine learning opportunities help in prioritizing ML opportunities.

**Fig. 2.16** Directed acyclic graph for opportunity association



## 2.8 Self-evaluation of Learning

Evaluation is the most important aspect of learning. Inappropriate or wrong evaluation criterion can harm your learning and even lead to wrong learning strategy. Intelligent agent that can evaluate progress and performance of its learning algorithm and has ability to correct it is something that needed to build a real intelligent system. In previous section, we discussed about learnability and its appropriateness. Many researchers used term learnability interchangeably with usability. A more learnable system can accumulate more knowledge and can accomplish complex tasks and solve complex problems swiftly. Here by learnability we mean—ability of intelligent system to adapt new things and exhibiting creativity with reference to unknown scenario that is not just part of knowledge base. Learnability is the ability to produce better results by learning from the same set of data. Learnability is measured in terms of state transition. It is the transition from one knowledge state to other.

Learnability index is defined based on state transition from state  $S_1$  to  $S_2$ . Here, assumption is that due to learning there is transition of state. Learnability is defined with reference to specific situation and environmental parameters. How can we measure learnability? In case of same data if the machine learning algorithm allows transfer to new state  $S_n$  where  $S_n$  has higher freedom and can be mapped to the better learning state then algorithm can said to improve learnability.

---

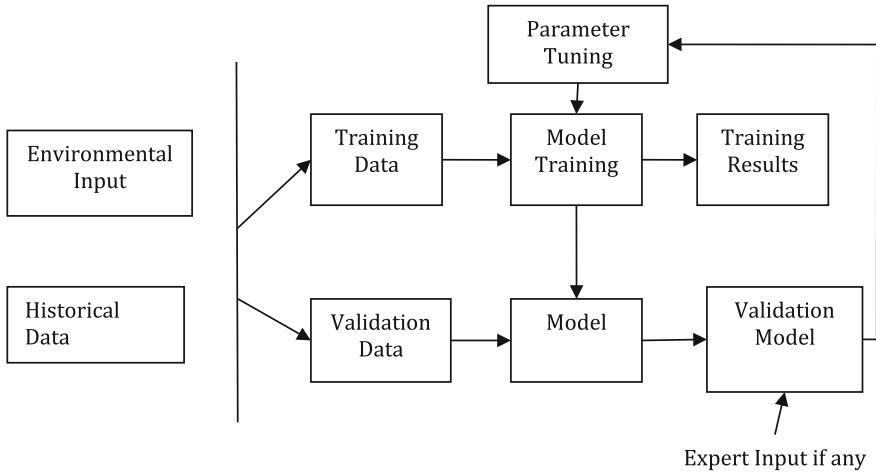
## 2.9 Mathematical Model of Learnability

Knowledge Canvas: Knowledge canvas depicts learning opportunities in the solution space. These are determined using freedom index and uncertainty. The machine learning opportunity points are represented on knowledge canvas. These points are associated to build overall learning opportunity. These opportunities are ranked in terms of importance, criticality, and freedom. So learnability is on set of provided limited data for learning coming up with better results, surprising solutions, and novel answers.

Let  $D$  be the data set for learning and is provided in the context of problem  $P$ . For a new scenario  $N$  if learning algorithm comes up with  $m$  solutions or partial solutions such that relevance of these solutions is greater than threshold  $T$  and association index with obvious solutions is below  $R$ . Based on higher association index the weight is associated with each of the solutions. Then learnability is given as follows:

Here,  $G$  represents gain,  $W_i$  is weight of solution, and  $A$  is Association with routine index.

$$L = \sum_{i=1}^m W_i(1 - A)_i G_i$$



**Fig. 2.17** Machine learning model

## 2.10 Building Machine Learning Models: Your Foundation for Surprising Solutions

Machine learning model is the deciding way to learn. Traditional models try to establish relationships between inputs and outputs. These models are different from typical mathematical equations. A typical ML model is depicted in Fig. 2.17.

The learning models are created using different ML opportunities. There is association between more than one ML opportunities. These opportunities together build a learning canvas for the system.

## 2.11 Opportunity Cycle

Learning opportunities are mapped to different learning stages. In the opportunity cycle, as knowledge acquisition takes place, new learning opportunities emerge. The machine learning opportunity points are accumulated like rewards in reinforcement learning to decide area of interest. The association among multiple opportunity points helps to represent the overall ML opportunity. The next stage is opportunity ranking. Based on opportunity, ranking and association, decision about problem selection is taken. The different learning opportunities and their association form opportunity cycle.



## 2.12 ML Big Landscape

This ML landscape includes large number of applications. Theoretically speaking every problem is a machine learning problem. To expand this landscape, we need to introduce uncertainty and expand the conceptual space.

Deliberate Uncertainty: For creativity there is need of uncertainty. For this purpose uncertainty is introduced. This is typically called deliberate uncertainty. These uncertainties are typically introduced at appropriate place based on requirements.

Learning paradigms are based on available learning resources and environment. There is learning from single event—which is referred as event-based learning. Similarly, episodic learning is learning based on episodes. These episodes are stored in memory for the reference. Learning based on single event limits the understanding and also vulnerable to noise. Hence, it is recommended to learn from multiple events. Learning from multiple events works very well to give broader view of problem. Repetitive occurrence of events and particular pattern across the events, data can be used for learning. The pattern-based learning thus goes beyond events to find out pattern. These patterns may be derived based on historical information. These all paradigms discussed so far work in controlled environment. These paradigms are majorly driven by historical pattern. But these methods do not work effectively in case of dynamic environment. Dynamic scenarios demand exploratory learning. Reinforcement learning is a type of exploratory machine learning. In exploratory learning, an intelligent agent explores in new scenarios to learn in such a way that rewards are maximized. These multiple agents can cooperate with one another. Cooperative learning is about learning together and improving the learning. Holistic learning tries to acquire the complete information. Multicenter learning looks for learning from multiple scenarios at multiple locations.

---

## 2.13 Context-Based Learning—Respect Heterogeneity

Knowledge context is important for learning. Context-based learning is about understanding the context where context is about situation, place, scenario and associated meta data. It learns based on context to act effectively in given scenario. This allows the system to act effectively in the given context. The context-based learning can be made possible using scenario-based learning. Here, representative scenarios are given for the sake of learning. Same action in a particular context may mean something different hence context based learning maps actions to context. Meta context helps in understanding more details about the given context.

## 2.14 Summary

Machine learning opportunities are everywhere. Understanding ML opportunity and selection of machine learning model are important tasks while building machine learning applications. The IDEA matrix can help us to understand ML opportunities and to evaluate problems. This evaluation helps in many ways. This helps in first identifying that it is a ML problem—further it helps in prioritizing ML problems. Different ML opportunities can be associated while going for ML model building. ML is after all building abilities in machine to solve problems. It is about learnability rather than precision. Learnability of an ML algorithm can be measured based on stage transition. The new paradigm of knowledge innovation based machine learning is focused on learnability, meta-knowledge, and continuous exploration.

<http://www.springer.com/978-3-319-55311-5>

Reverse Hypothesis Machine Learning

A Practitioner's Perspective

Kulkarni, P.

2017, XVI, 138 p. 61 illus., 9 illus. in color., Hardcover

ISBN: 978-3-319-55311-5