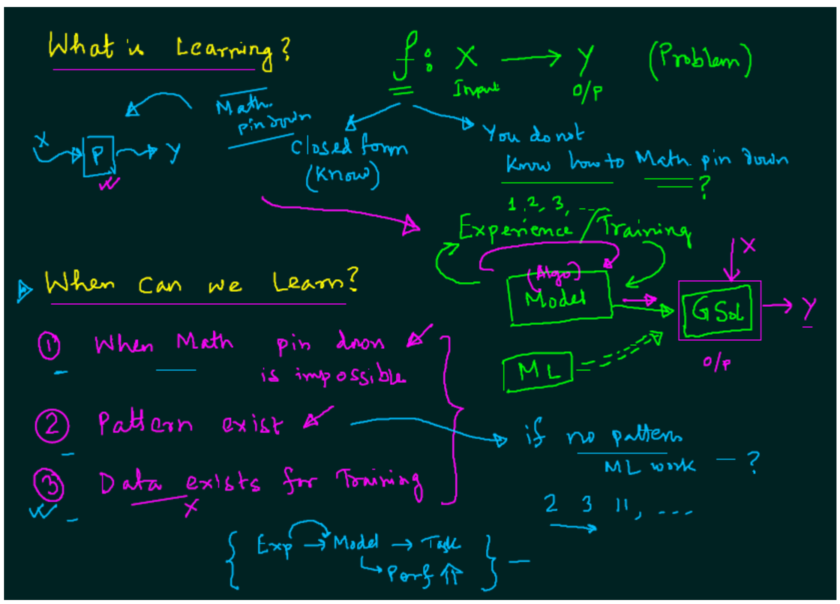
Lecture Scribe
for
Machine Learning (CS60050), Spring 2020-2021
Instructor: Prof. Aritra Hazra

Introduction
Date: 6^{th} January 2021

(Refer Slide Time: 00:36)



What is Learning:

Till now we have tried to solve a problem. There is something called inputs (say X), and there are certain outputs for that problem (say Y). When we have tried to solve that problem (say f), our primary objective is to find the solution to translate any input to its desired output.

$$f : X \to Y$$

Until now, whatever we have seen for this particular solution of the function 'f', we know that it has a closed-form either in mathematical or programming sense because we can iterate over the input and produce the desired output. So, in general, we know what the solution is, and we can mathematically pin down the solution. Therefore, we can write codes for the function and able to find the solution. So, this kind of thing when we mathematically pin down what we can do is that we write a program (say P) which can take arbitrary inputs (say X) and can produce desired outputs (say Y).

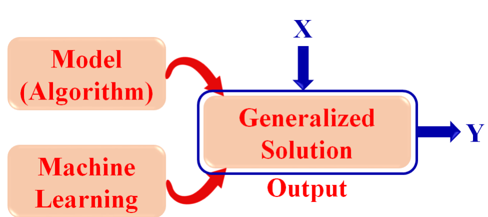
So, for a problem (f), we can solve it by mathematically pinning down to its closed or a known form which can deterministically tell the outputs by given the inputs.

There are another set of problems if you do not know how to pin down mathematically. So this set of problems are the concern in the whole course that we are going to learn.

That means we do not know the exact way of solving for every input, and also, we do not know all the varieties of the inputs that we'll get to solve the problem. So, therefore what we have to do is think now about how humans come and learn.

Think how a little boy or girl starts learning the color! We go and teach him something that this is red; this is yellow; this is green. One day, he tells everything wrong initially, then he corrects one/two things, and suddenly he makes everything correct. We called this '*Experience/Training*', which is essential to solving those problems that we do not know a mathematical answer. So, therefore now someone has to iterate over this experience and training in the human brain. Some model needs to iterate over this experience or to train one by one as the first example, the second example, the third example, and so on. And they go on iterating and finally coming up with a generalized solution of the understanding of the problem. This is the part that we called a '*Machine Learning*' (ML) where our solution is that generalized one.

So, it means that we had a set of training examples through some algorithm in such a way that finally come up with a model of the program itself, not directly doing the program as in other sense but we come up with a certain algorithmic way of the program. So, the generalized solution is the output of the learning. So, this program is now capable of taking 'X' and producing the desired 'Y'.



This is the fundamental part from where we are trying to deviate from a deterministic solution of a problem towards a learned solution of a problem, and that we what by mean by '*Learning*'.

When can we Learn:

- When mathematical pin down is impossible, e.g., trying to recognize a face. While recognizing a face, we can not mathematically pin it down because everyone's face has certain other features. Another example is recognizing finger-prints for authentication, where some features are different.

- When a pattern exists. E.g., when a small kid is learning the color of a particular object, some pattern exists, which dictates the color. The same thing happens when someone is trying to recognize a face. So, in contrast with, if a function produces a random output, we can not learn it. So, those kinds of problems we can not learn.

- When data exists for training.

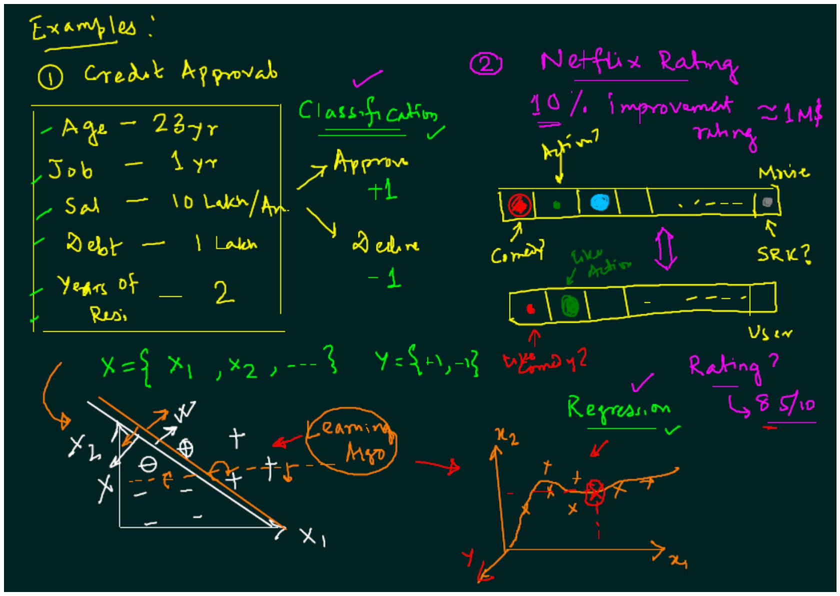
Now, see what happens if we relax these assumptions. If data does not exist, then forget about learning. So, data is required. But, let us say there is 'no pattern'. No pattern does not mean random, it is a wide variety, and we can not put it down in terms of a pattern. The ML will also work, but it is not working as well as a belief to be. As we do not have any basis for formulating the model, it may work in a little bit vague way. So, even if no pattern exists, we can apply the ML solutions or ML algorithms to build up a model, but it does not work as expected.

Secondly, can we do ML for the deterministic problem as well? That is when we mathematically can pin down. Let's say, can we determine a number whether it is prime or not? Suppose we have thousand of test examples. Eventually, due to these experiences being gathered in that model, we can somehow classify which one is prime and not. But that is not the goal of ML, and that can not establish the benefit of ML. When mathematically pin down is possible, it is not the tool we will use for ML. So, ML is not for those that we know we can mathematically solve the problem elegantly.

Definition of Machine Learning

Machine learning is a procedure through which, from the experience or the training, we can build up a model or an abstraction of a solution to solve a task with higher experience. The performance over solving a task improves.

(Refer Slide Time: 13:04)



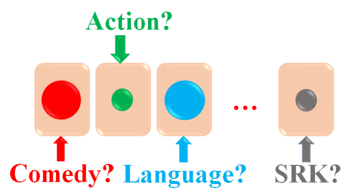
Example of Machine Learning Problems:

- **Credit Approval Problem:** Consider the profile as follows:

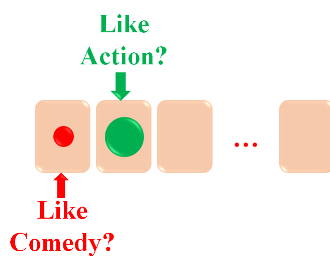
Age	– 23 Yr
Job	– 1 Yr
Salary	– 10 Lakh/Annam
Debt	– 1 Lakh
Years of Residence	– 2

  These are the inputs depending on which we need to classify whether they will approve for a loan or decline. Now see, this is the actual problem to learn as we can have some set of data in our hand, and unless a data I can not do anything. Here, we have data as we have prior customers, and we have learned for approval or declined for loans. In this case, we can not mathematically pin down because case to case basis may vary, and the number of attributes (such as Age, Job, Sal, and so on) can be huge. But there must exist a pattern. So, this is a good problem where we could apply ML over it. As we see, we want to decide as +1 for approval and -1 for declined. We called such kind of problem as '*Classification*'. Depending on the input attributes and their value, we want to classify whether we will pass it fail it. So, here 'X' is the attribute set, say '$X = \{x_1, x_2, \dots\}$' where x_1 is Age, x_2 is Job, and so on. And, 'Y' is only accept and declined, that is, '$Y = \{+1, -1\}$'.

- **Netflix Movie Rating Problem:** Here Netflix asked for a 10% improvement in its prediction operating. So, some movie comes out, and people have to predict what is the rating of that movie will going to be. And, Netflix has said that if somebody predicts in such a way that the existing algorithms they do, let's say they predict the rating and they make the improvements of the prediction depending on the closeness of the actual rating, they will give a 1M$ for the prize. And it took ten years to get the 10% improvements. Now, let us see the features of this problem. A movie has a set of attributes in it. E.g., a movie may contain a comedy for too much, or it has less action, language is English accepted by a broad customer, or maybe there is an attribute that tells whether Shahrukh Khan is participating and how much screen presence that he has. So, every movie has specific features as follows:



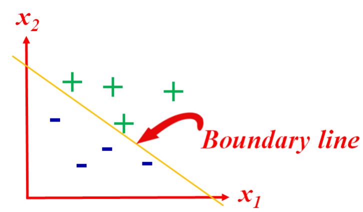
If we try to think about the user who rates the movie, he also has specific parameters to rate it. Such as user likes comedy minimal, user likes action more, and so on as follows:



Concerning the Netflix rating problem, we are trying to co-relate the attributes of the movie with the user's attributes to find out our ultimate goal that is the rating of the movie. Here, in this case, unlike the classification problem, it is not approved or declines. Rather we are trying to predict, let us say IMDb 8.5/10; that is, we are trying to predict something. We called such problem as '*Regression*'.

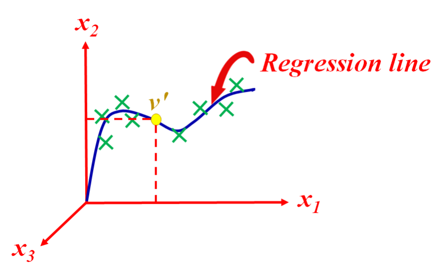
The basic difference between Classification and Regression: In the case of classification, the output is 'Yes' or 'No'. Now, in the case of regression, instead of such 'Yes' or 'No', we want a value-based answer.

If we see these examples in the ML algorithm's view, then we will find certain dependencies. Consider the Credit Approval example; there is an attribute feature space, and suppose there are only two attributes x_1 and x_2. So, depending on this feature space, some is classified as good, which we can approve (in the following figure it is marked as '+'), and some of them are classified as bad those we can reject (in the following figure, it is marked as '-'). Now, our job as a learner is to take one by one example and try to classify a boundary to distinguish between good and bad entities with respect to the problem, here it is the loan approved or not.

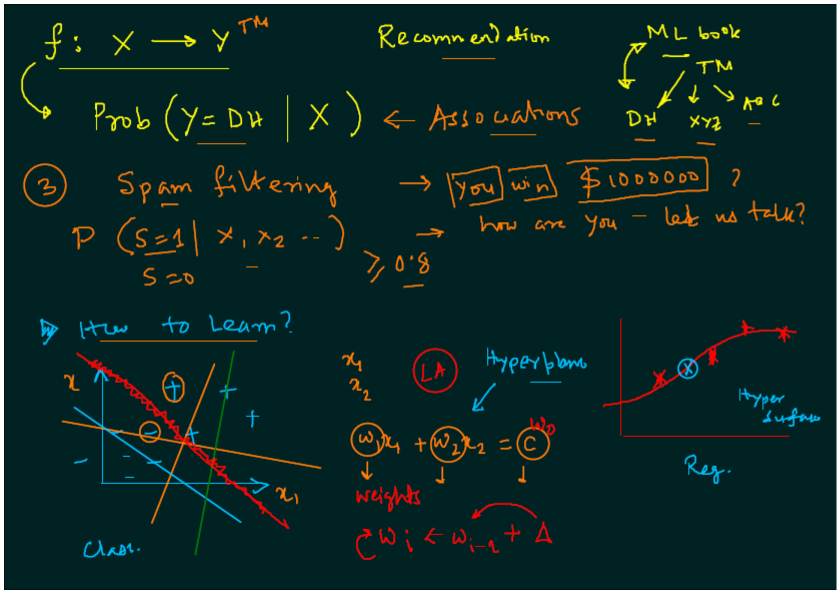


In our learning algorithm, we should start with an arbitrary boundary where we should misclassify something. Then we gradually try to switch it up/down or rotate this line accordingly with its slope and try to develop the fitting line that can classify the training examples correctly. That is what our '*Learning Algorithm*' do for the classification examples.

Now, consider the regression example. Suppose, consider only two/three features that lead a two or three-dimensional space, respectively. Suppose the three features are x_1, x_2 and x_3. The regression is trying to do that. Suppose we have some prior knowledge that it will take at the third dimension. Now, we are trying to fit some curve in between those values so that whenever we are trying to judge a value, say v', we can judge what the rating value would be. See the example as follows:



(Refer Slide Time: 22:45)



- **Online Recommendation List Problem / Spam Filtering problem:** An online website, say Amazon, will give us certain recommendations when we buy something. This kind of problem is called recommendation problems. Suppose we have bought ML book by Tom Mitchell, they give us certain recommendations like we may also buy Duda-Hart, XYZ, or ABC. This problem is an association problem between the existing we purchased and the other that we have. Until now, we learn the function (say f), which has a set of inputs (say X) and a set of outputs (say Y). I.e., $f : X \rightarrow Y$. So, concerning this recommendation problem, effectively if we solve the problem f, the other we could look into the probability of Y being Duda-Hart, given that the set of attributes (here X) through which I bought an ML book, i.e., $Prob(Y = DudaHart|X)$. So, this is the other answer that we also might look for. This is called '*Associations*'. So, all these suggestions you see in the recommendation kind of a thing in any online sites are the top list probability values through which is being associated with our existing buying option. The most interesting example of this kind of probabilistic association is Spam Filtering. Here depending on the association of some specific words and their positioning, we judge whether the mail is spam or not. It is a classification problem, but the classification decision is more governed by the probability analysis of it, which we called the posterior probability.

Therefore, we can see the scope of Machine Learning is enormous. If we try to think about such a problem, we can immediately figure out what can be learnable and recognize how we can use a learning model.

How to Learn:

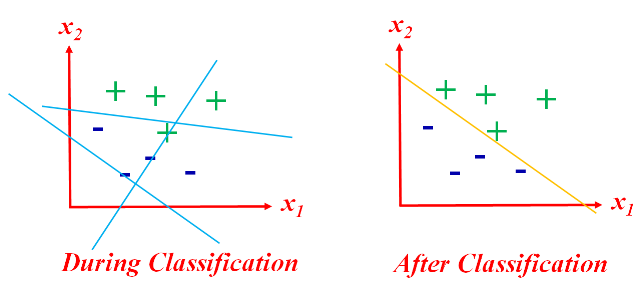
Consider the previous example of the classification. We have certain attributes, say x_1 and x_2, and depending on that; we want to classify. To classify in 2-dimension, we need to put a line. But the line has to misclassify some of the points. Since we consider only two attributes, we have to consider the line; else it will be hyper-planes. Now, consider the equation of a line:

$$w_1x_1 + w_2x_2 = c, \textit{ where } c \textit{ is a constant}$$

Here in this equation, w_1 and w_2 are the weights, even if the constant c can be written as w_0. Now, the learning algorithm tries to change each of these weights with its earlier value and with some updates of these weights, say Δ. It can be mathematically written as follows:

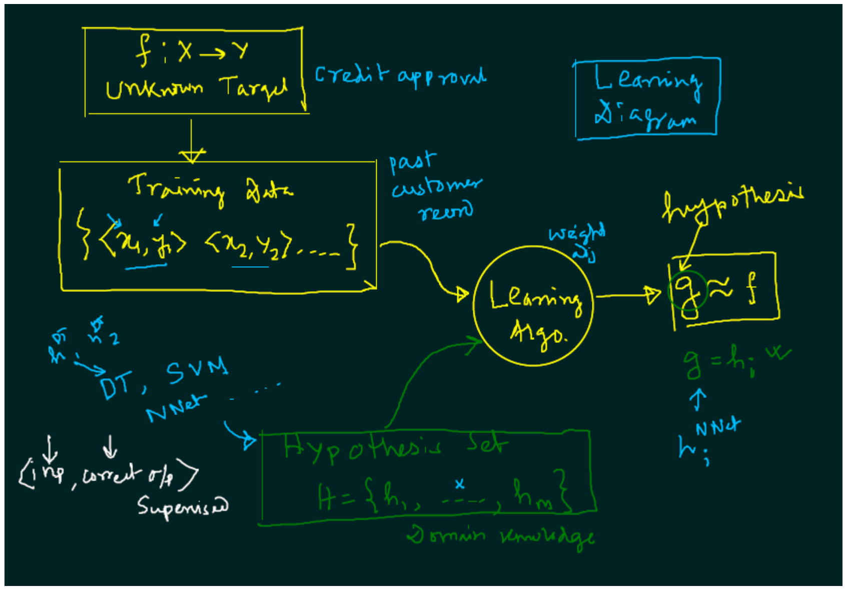
$$w_i \leftarrow w_{i-1} + \Delta$$

Now keep on modifying these weights iteration-wise unless you come up with a proper classifying line as follows:



Now, in the case of regression algorithm, here we try to fit in something which minimizes the error with the training, where the error is how much it deviates from the training data, and the curve should properly fit within it. And depending on the good we fit, new data can be given the value of the better solution. Notably, in the case of a higher dimension, it will be a hyper-surface instead of a curve.

(Refer Slide Time: 30:38)



Ingredients of Learning or the Learning Schematic Diagram:

Consider the schematic diagram as follows:

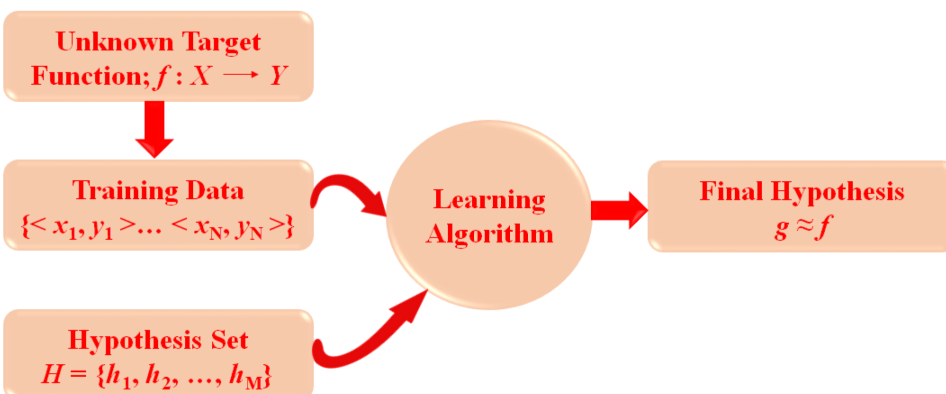


Figure: Learning Schematic Diagram

Here we want to learn a function $f : X \to Y$, the unknown target. In our hand, we have the training data set, i.e., $\{< x_1, y_1 >, < x_2, y_2 > \cdots < x_N, y_N >\}$. We are now trying to write our learning algorithm, which takes the training data and tries to find out a function, say g, which approximately relate f, and we denote this g as the hypothesis. Specifically, we want to choose the function g from the hypothesis set H, where $H = h_1, h_2, \ldots h_M$. Therefore, we may say $g = h_i$, where $i \in 1\ to\ M$. Choosing the value of g from the set H provides a certain kind of bounds from the learnability aspects. Secondly, the set H also governs from the domain knowledge. For example, consider the previously discussed credit approval problem. If we redraw this problem concerning the learning schematic diagram, it looks as follows:

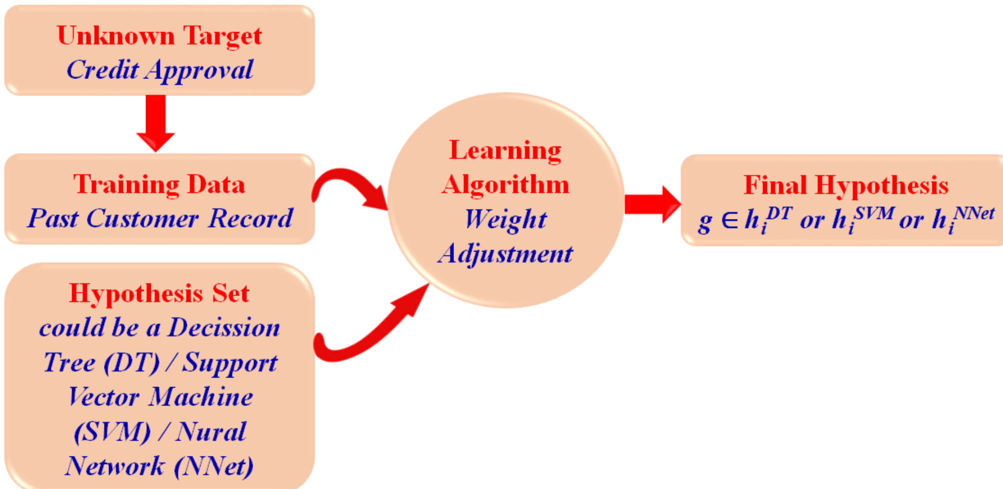
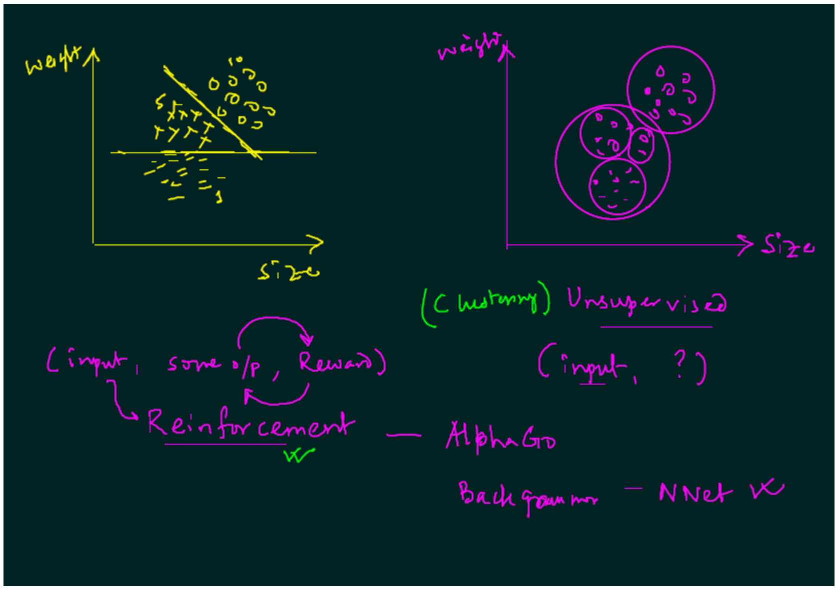


Figure: Learning Schematic Diagram for Credit Approval Problem

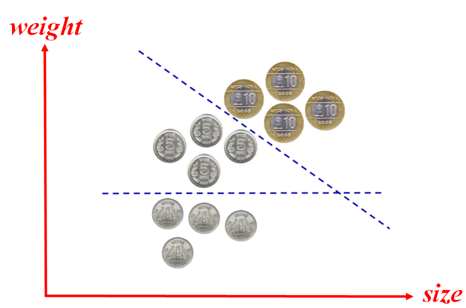
Depending on the kind of learning data, we can determine what kind of learning it is. Therefore, when the training data set consists of input and correct output, it is called '*Supervised Learning*'. Another one is '*Unsupervised Learning*'.

(Refer Slide Time: 30:38)

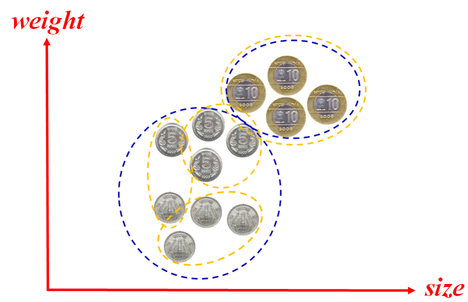


Supervised, Unsupervised and Reinforcement Learning:

Here we should discuss the concept of these three types of learning. Suppose there are three types of coins: 10 rupees, 5 rupees, and 1 rupee. Now, consider the 2-dimensional graph plotting the size of the coin at X-axis and the coin's weight at Y-axis. Here we plot the coins at this 2-dimensional space according to their size and weight. Now, in the case of '*Supervised Learning*', we can easily categorize the graph as we know which of them are 10 rupees coin, or 5 rupees coin, or 1 rupee coin. Here, the training data set looks like $< input, correct\ output >$. We can describe this concept as follows:



In the case of '*Unsupervised Learning*', we try to find out clusters. As here, we have not any supervision. Therefore the number of clusters is not in our hands. Here, we have only some data in our hands, and we don't know about the output. So, we need to categorize them. The training data set looks like $< input, ? >$. We can describe this concept as follows:

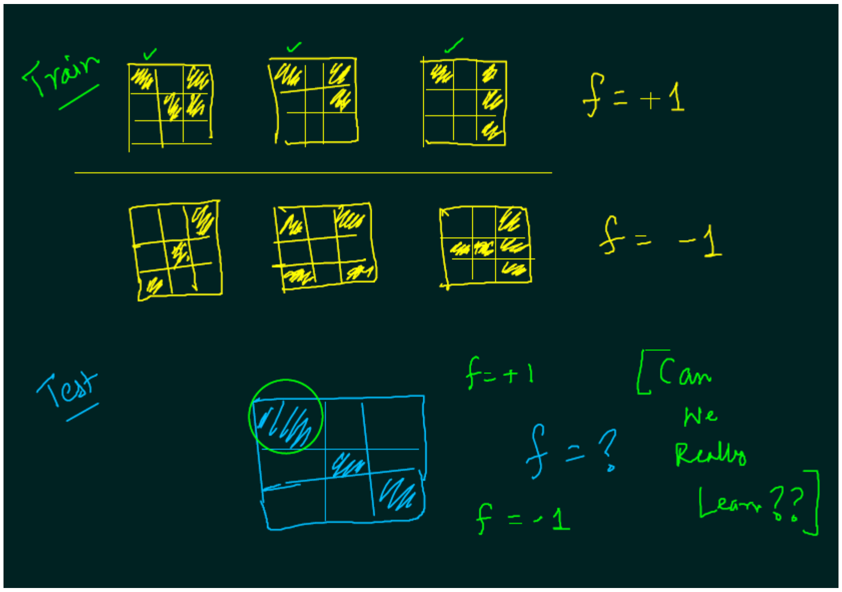


In the case of '*Reinforcement Learning*', the training data set looks like $< input$, *some output*, *Reward* $>$. To understand the reward, we can consider the example of a child reaching his hand in front of a hot cup, then he touches and pulls back his hand. So, here the reward is negative. So, after some time, the child may figure out that may be the smoke coming out of the cup is a very bad thing. This is called reinforcement. Some of the beautiful examples of this kind of learning is the game playing 'AlphaGo' or 'Backgammon'.

Summary

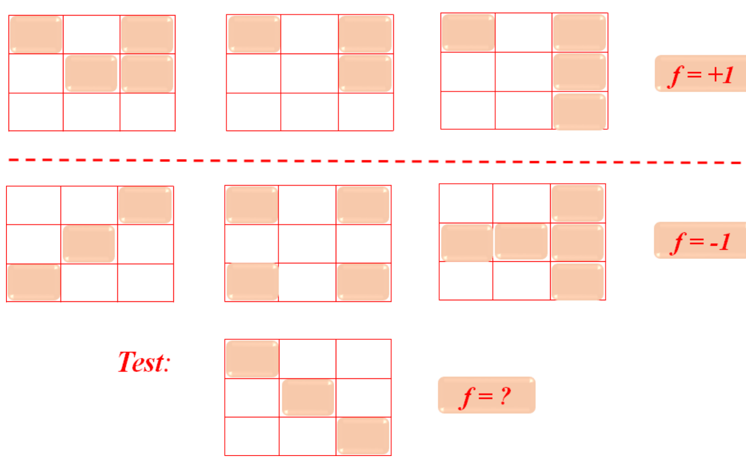
- **What is Learning**
- **When can we Learn**
- **Classification, Regression, and Association Problems**
- **Characterize the learning Diagram**
- **Types of Learning: Supervised, Unsupervised, and Reinforcement Learning**

(Refer Slide Time: 41:20)

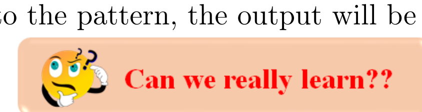


The Puzzle:

Consider the puzzle as follows. Here a set of training examples and the corresponding outputs are given. In the test case, another pattern is given, and we should find the corresponding output.



Answer: If we classify it concerning the leftmost upper cell, the output will be $f = -1$. If we classify it with respect to the pattern, the output will be $f = +1$. So,



Scribe prepared by Tapadyoti Banerjee (20CS91R13)