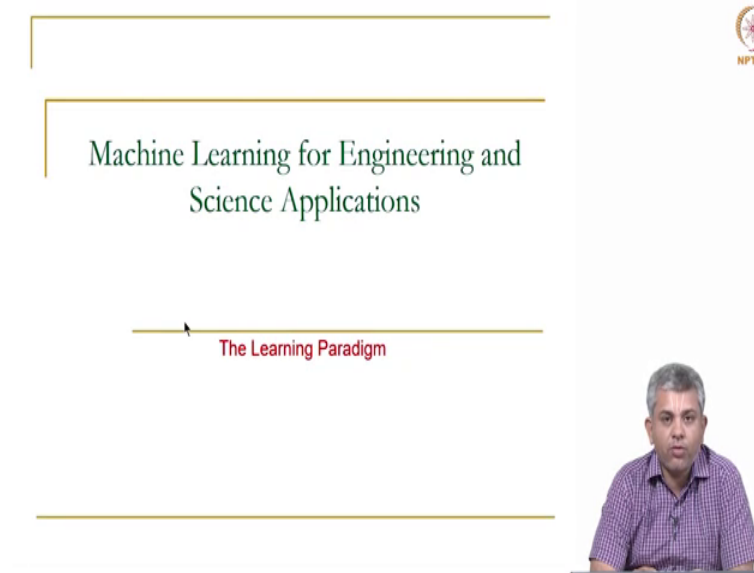


**Machine Learning for Engineering and Science Applications**  
**Professor Dr. Balaji Srinivasan**  
**Department of Mechanical Engineering**  
**Indian Institute of Technology, Madras**  
**The Learning Paradigm**

(Refer Slide Time: 00:15)




Welcome to week 4 of machine learning for Engineering and Science Applications for the first 3 weeks we were looking at some mathematical and computational preliminaries, this week we will actually start our excursion into machine learning. So before we do that I will be introducing you to the basic learning paradigm that will be used especially for the deep learning module. You might recall that we had split the course into essentially four large sections, the first 3 weeks were the mathematical preliminaries the next set which will be continuing until approximately week 8 would be deep learning ok which is the most popular machine learning algorithm or the machine learning family in use right now.

Ofcourse we never know which algorithm is going to make strong come back so we are going to concentrate on other machine learning approaches after we finished deep learning and finally will look at some advanced algorithms in the last couple of weeks. Now the paradigm that I am (discovering) describing right now in this video is particularly applicable to the deep learning set ok so you will see that the various models that we look at or the various algorithm we look at

will have a sort of standard template. So what I am describing right now is the general template so that you see the pattern when we repeat it.


Often it is easy to lose yourself into the algorithm without seeing the bigger picture so what I am describing right now before we get into deep learning is the overall paradigm or overall template that we will be using here.

(Refer Slide Time: 01:51)



### Minor changes in the syllabus

- Previously, we had planned to introduce applications at the end of each week.
  - As it appears in the syllabus
- This had several implementation problems (especially due to time)
- We will now remove advanced portions
  - Week 11 – Structured Probabilistic Models, Monte Carlo Methods
- And instead have one whole week of Deep Learning Applications to Science and Engineering
  - This will be either Week 9 or Week 10
- The lectures in the other weeks will cover theory (primarily)



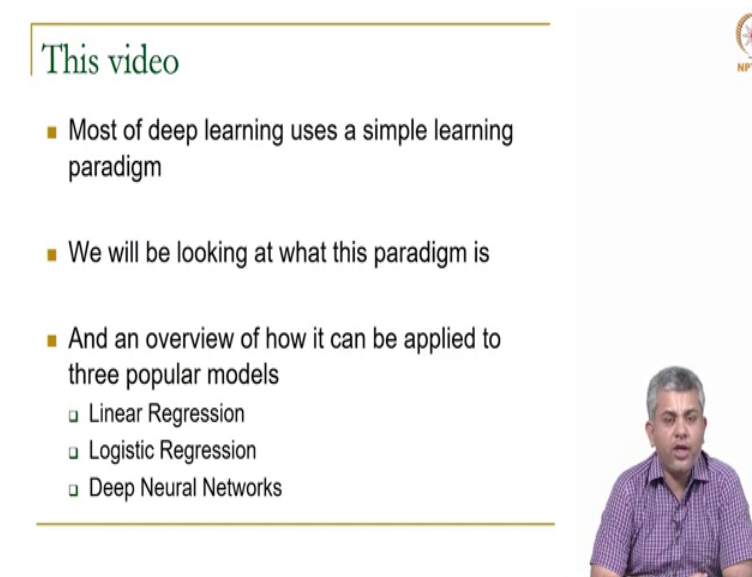
Before we go to the deep learning or the learning paradigm I would like to talk about some minute changes in the syllabus as announced in the course at the beginning of the course. So we have some minor changes, previously we have plan to introduce applications at the end of each week so we finish a week of lecture and then talk about some application of this as we step into the deep learning.

However this had several implementation problems especially due to the time that it takes usually to complete the weeks lectures that itself is around two and a half to three hours and if we start discussing applications who have taken much longer. So what we have done now is we have removed some really advanced portions which we had kept as week 11 of this course which is Structured Probabilistic Models on Monte Carlo methods anyway you can do this as part of other courses (and) instead we will now devote one full week this will be around week 8 or 9 or week 9 or 10 of this course and we will have one whole week after we finished the deep learning

portions will discuss all the possible engineering applications that have especially come up over the last few years.

There have been some old applications but especially new applications we will be discussing you will find some surprising instances of this in week 9 and 10 and we hope that you will enjoy that will have a concentrated discussion of applications ok. Other week lecture will ofcourse cover the primary theory and the primary computational ideas that you need in order to implement deep learning and we will take some standard examples and some engineering examples but the heavy load of applications will be relegated to week 9 or 10 of this course.

(Refer Slide Time: 03:31)



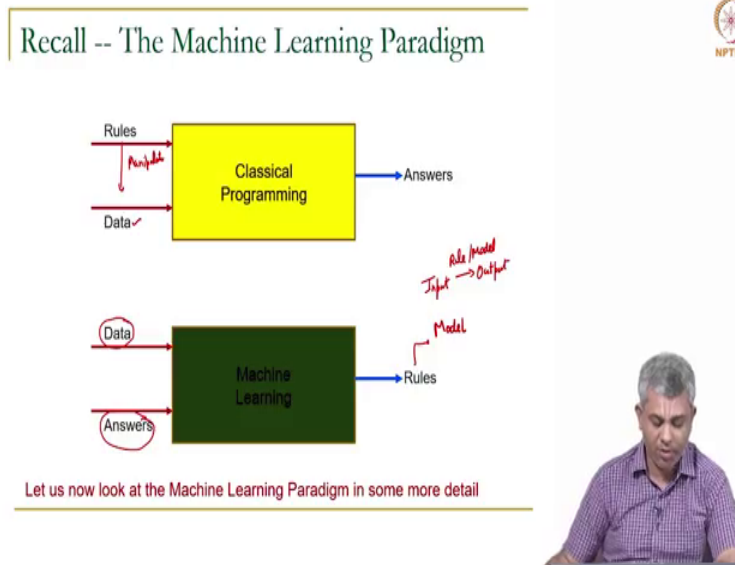
The slide features a title 'This video' in green text at the top left. Below it is a bulleted list of topics. To the right of the text is a small inset video of a man in a purple checkered shirt speaking. In the top right corner of the slide area, there is a circular logo with a star and the text 'NPTEL' below it.

**This video**

- Most of deep learning uses a simple learning paradigm
- We will be looking at what this paradigm is
- And an overview of how it can be applied to three popular models
  - Linear Regression
  - Logistic Regression
  - Deep Neural Networks

So in this video we will be discussing the following ideas most of deep learning uses one simple learning paradigm or one simple template by which it learns ok.


(Refer Slide Time: 03:45)



We will now look at somehow it applied to linear regression logistic aggression I will give you a brief idea and then we will get into the details of this. So recall we had looked at this in week 1 itself the standard machine learning paradigm is different from what happens in classical programming. In classical programming you give some data and you if that rules for manipulating data and that gives the final answer that you require ok.

On the other hand in machine learning we actually give the data the input data we give you the final answers that are required and you are supposed to figure out what the rules are, this kind of rules is what I call model. Model means basically how is the input related to the output. So if I speak and you are hearing words even though all I am doing is making sound waves in air ok, how is it that you are translating the sound waves into some words that is a model ok so that is the model that is data terms it is basically you take a set of numbers and turn it into another set of numbers through a map ok.

(Refer Slide Time: 05:06)



**General Paradigm**

Input Data  $x$  → Machine Learning → Parameters/Weights ( $w$ )

Output Data  $y$  ← Machine Learning


Relationship? (between  $x$  and  $y$ )

Deep Learning (handwritten note)

- We wish to learn the relationship between the input and the output data.
- For now, we will think of this relationship as a function
  - We can call this function the **model** or the **hypothesis function**
- The function has two parts
  - Form of the function →  $e.g. y = h(x) = w_0 + w_1x_1 + w_2x_2 + w_3x_1x_2$
  - Parameters of the function
- Typical machine learning **learns only the parameters**
  - The form is provided by the ML engineer. Requires domain knowledge

This modeling involves two processes – **Feedforward** and **feedback**

Handwritten notes:  $x = [x_1, x_2]$ ,  $w_0, w_1, w_2, w_3$  are unknown weights/parameters, found by the algorithm.



So now let's look at this in some more detail, so let's look at now specifically the machine learning paradigm. So what you have is what comes in is an input data you also give the output data ok and what you figure out are what are called parameters or weights please notice this word this will be recurring for a long time until we go till week 9 or classical deep learning is all about learning the parameters or the weights ok. So our past is to learn the relationship between this two, this is what we don't know ok. So if we see a bunch of pixels and we identify this as human being what is it that is happening there, that is what we want to learn.

In class in engineering or science terms we want to know the model ok, we want to know the model for you know why is it, what is the relation between today's temperature and yesterday's temperature, so you can treat today's temperature as  $y$ , yesterday's temperature as  $x$  and you want to know what is the relationship between this two ok that would be a model. So we are using the same term as big umbrella term as a model which relates one thing to the other thing ok. So for now we will think of this relationship as if it is a function is the standard model we will be using till we come to the end of deep learning ok.

We call this function, what is this function? This function that relates  $x$  to  $y$  we will call this as the model or the hypothesis function ok. Remember in science we postulate or we have a hypothesis ok this is probably what is happening so you have a hypothesis something like the relationship between the two forces of earth and sun is  $jmm$  by  $r$  square so that it is a hypothesis

so it is a mathematical hypothesis it is a function. Now a certain part is every function has two parts ok so let us take one function, ok so let us say  $x$  is a vector and it has two components  $x_1$  and  $x_2$  ok and I have my function  $y$  is  $h$  of  $x$  and is given as  $w_0$  plus  $w_1 x_1$  plus  $w_2 x_2$  plus  $w_3 x_1 x_2$  ok.

Now this function itself this function here has two parts, one is the form of the function and another is the parameters of the function, what is meant by form? Form is the fact that the first term is a constant the second term is linear, third term is linear and the third and the fourth term is non-linear  $x_1 x_2$ , this is called the form, the parameters are this  $w_0, w_1, w_2, w_3$  ok. So when you have some  $x$  and some  $y$  you can have infinite number of forms you can have linear you can have quadratic, you can have cubic, exponential whatever it is and you also have in each one of this you have some unknowns, this are called the weights or the parameters or sort of the knobs that you turn for this function to look different each time.

Ok if you have a linear function  $w_0$  plus  $w_n x$  you have two weights, if you have this sort of a function you have 4 weights. So then we say that machine learning learns and what it give out is actually only the parameters or the weights. So within the deep learning module this is the only thing will be doing we as users already give the form ok so suppose I say this is the form of the function, machine learning is not smart enough to say ok let me also try you know  $x_1 (x) x_1 x_2$  square it cannot try any such thing it can only try within the limit that you have given it ok so user defines the form.


User or the programmer or the machine learning engineer gives this form and the parameters are found by the algorithm ok. So when we look at the learning algorithm all it learns are just the parameters ok. Now how do you decide under form? That usually requires domain knowledge ok that is you need to know it we will see some examples shortly but we need to know what it looks like. So if you have the relationship between let us say you know the voltage and the current you kind of know from intuition or from your physics knowledge that it supposed to be linear. Somebody working in solid mechanics stress and strain, they know it kind of bilinear or maybe a small variation on that ok.

So the form is usually decided by what you already know from science or from engineering about the function and the parameters or the weights are given by the machines. In some sense


though this is not quite fair as you will see in some sense sophisticated curve fitting is what is going on in much of learning machine. Now this modeling process which is going to figure out these parameters or weights involves two separate processes, one is called feed forward and one is called feedback let us see what these are.


(Refer Slide Time: 10:41)

### Forward Modeling



- A model or hypothesis is simply an educated guess at what the relationship between input and output is.
- As mentioned before, it has two pieces
  - Form of the function – Linear, Quadratic, Exponential, etc
  - Parameters of the function
- We sometimes use the notation  $y = f(x; w)$ 
  - Given  $x$  and a choice of  $w$ , we can find a corresponding  $y$
- The function  $f$  going from  $x$  to  $y$  is called the forward model
  - The process is sometimes called feedforward





Ok so let us look at feed forward or forward modeling, so let us say you have a process of this sort you give  $x$  you also have to give the weights and what comes out is  $y$  is  $f$  of  $x ; w$ , so model or a hypothesis as we discussed in the previous slide is simply an educated guess at what the relationship between the input and the output is in some cases it is kind of obvious some cases it has not ok so you usually have two pieces as I mentioned just now you have the form which is linear quadratic exponential etc and then you have a parameters the unknown constant adjustable constant that are sitting there.

So we sometime use the notation  $y$  is  $f, x ;$  what it means is this is a parameter, whatever comes after the semicolon in this notation basically means it is a parameter ok. Now notice that whenever I give you  $x$  suppose this is given suppose I give you some choice of  $w$  ok I am even guessing  $w$  ok you can find a  $y$  if you are given  $f$  if you are given the form of the function  $x$  and  $w$ ,  $y$  is fully determined ok. This process of going from  $x w$  and coming out with  $y$  ok is called the forward model ok. We will see the significance of this as we go on in the course ok, this process is sometimes also called feed forward please remember the terms. Feed forward is

nothing but putting  $x$  and  $w$  inside this box and then getting out  $y$  ok, so that is simple forward model.

(Refer Slide Time: 12:35)

### Learning the parameters via feedback

To learn the parameters, we follow this paradigm

- Collect lots of data pairs (Input Vector, Output Vector) =  $(x, y)$
- Guess for the form of the hypothesis function  $h(x; w)$ 
  - Example:  $h(x; w) = w_0 + w_1x_1 + w_2x_2$  → Over-fitted (Does not change)
- For an arbitrary guess for  $w$ 
  - We will get some  $\hat{y} = h(x; w)$  which will not match the ground truth  $y$
- Define a cost function  $J(y, \hat{y}(w))$  depending on the difference
- Find optimal  $w$  by minimizing  $J(w)$  → Gradient Descent (Optimization process)
  - By using some optimization procedure such as Gradient Descent



Now however as I said earlier what we are interested in knowing is  $w$  ok so this is the way it works. So let us say you collect lots and lots of data ok so you collect  $x$  a corresponding  $y$ , suppose we are doing a voltage and current thing so you do one experiment current you know with a particular resistor find out voltage current, do a second experiment voltage current ( $v_1$ )  $i_1$ ,  $v_2$   $i_2$  so on and so forth to collect lots of data. So we will call the input as usual  $x$  and output as  $y$ . now I am also going to use the term ground truth ok, what does ground truth mean? This is what came out of experiment ok. Suppose a standard machine learning example is suppose we are trying to find out you know house prices in a particular area?

Ok so you are guessing that house prices in that particular area are cost let us say 1500 rupees per square foot but you will actually have to go to the house and see, measure the area, measure the house price and the price might actually be different from what you guessed ok so the owner might say no I am going to charge you 2000 rupees per square foot ok. Ground truth is what actually exists out there in reality. Similarly if a person is doing a radiologist or somebody is trying to identify cancerous tumor. You go to the radiologist, show the images the person will look at the scan and say ok this person has cancer that is the ground truth.



Now you have your model which is separate from the ground truth, what is the model going to do? It is going to do a mathematical process we are going to use a hypothesis function  $x$  stands for hypothesis that is you do not know how the radiologist came up with cancer or no cancer, you are going to guess a function ok so we give a function and for that the user has to give a form ok. For example we say that  $h$  of  $x$   $w$  is  $w_0$  plus  $w_1 x_1$  plus  $w_2 x_2$  this is a guessed model ok. Now we do an iterative method just like we did gradient descent ok. So what do we do we initially guess for some arbitrary value of  $w$  ok.

Now once I know  $x$   $w$  and I also know the form of the function this is remember already user fixed this will not change during our learning ok. In the middle of the learning we don't suddenly say ok from linear let me switch to quadratic ok once you have a learning that optimal learning will happen for a particular form of the function ok. Now what will happen is, you will get some  $h$   $x$   $w$  we will call it  $\hat{y}$  in order to distinguish it from  $y$ . What is that suppose the radiologist said cancer and you said no cancer ok based on your hypothesis function or based on the parameters that you have chosen this is  $\hat{y}$  ok so this is experimental truth, this is our guess ok.

So what happened was  $x$  was fixed  $w$  was guessed  $y$  is fixed because that is the truth and  $\hat{y}$  depends on  $h$  ok. So our guess gives us some  $\hat{y}$  the ground truth is something else so you are going to find out that there is usually a difference between the ground truth as will see in several examples shortly in this week. There will be some difference between the ground truth and our hypothesis ok. So what we do is cleverly we say we define a cost function ok so what the cost function does, is that it tell you when ground truth was 1 and you said 0.8 so it is going to cost you ok to have this difference between 1 and 0.8 ok so you can have various different cost functions. So once  $y$  and  $\hat{y}$  are given you can actually find out  $J$  which will be the difference between in some sense a difference between  $y$  and  $\hat{y}$ .

In general what is supposed to happen is if  $y$  is equal to  $\hat{y}$   $J$  should be zero ok and there are several function which satisfy this which will see again as the course proceeds ok. So dependent on the difference between  $y$  and  $\hat{y}$  we define a  $J$ . Now what we do is we find out that  $w$  which minimizes that cost ok so this is the general idea, this is a very important idea so the learning idea of finding of  $w$  is now reduced to a optimization problem and this is why we did all this optimization in week 3 ok. So we use some procedures such as Gradient descent.


For example suppose we know  $j$  then you can keep on iterating and find out what is the optimal  $w$  which minimizes  $J$ . So this idea we will use again and again. So just to summarize  $h$  is user defined  $x$  and  $y$  are data you guess a  $w$  run it through  $h$  it gives you a  $\hat{y}$ . Now you find out  $y$  and  $\hat{y}$  are different it gives you a cost  $J$  then once again using gradient descent for some such method you improve your  $w$ . This process of going from  $J$  to an improved  $w$  is called feedback. This is very-very similar to gradient descent. I have written gradient descent there because as far as this course is concerned will only be looking at gradient descent and its variants. Some variants will consider the next week but will look at gradient descent and its variants for this course.


(Refer Slide Time: 18:53)

### What the DL engineer must provide

- Appropriate decisions for input and output vectors  $(x, y)$ 
  - Recall : All problems are data, all solutions are functions/maps
- Choosing appropriate datasets *(Should be usually large)*  
 *$(x_1, x_2, \dots, x_n)^T$*
- Some appropriate form of the forward model  $\hat{y} = h(x; w)$ 
  - Example : Linear Model  $\hat{y} = w_0 + w_1x_1 + w_2x_2 \dots + w_nx_n$
- Form of the Loss function  $\rightarrow$  *Domain Knowledge*
  - Example : Least Squares  $J(y, \hat{y}) = (y - \hat{y})^2$
- Optimization algorithm – Example: Gradient Descent
  - And associated hyperparameters such as  $\alpha$

Machine Learning is not magic! It requires a lot of input from engineers






Now all this process tells you is that what all do you have to provide us an engineer. So first of all you have to decide what is an appropriate  $x$  and what is it appropriate  $y$  for this problem. So that is an important part as I have said much earlier in week 1 itself. Please remember all problems any problem even a seemingly qualitative problem can be kind of reduced to a data problem and all solutions as far as we have concern will be shown as functions or maths. You have to choose an appropriate and usually a large data set. A lot of this is interaction sometimes you might choose some nice  $x$  and  $y$  for the problem some nice input and output vector but you might not have it available.

Ok so you might have to do something a little bit more clever in order to find out so you have to find out  $x$  and  $y$  so that you need a large data set ok so finding out a good data set is a very-very important part of the process. Now you need to decide on some appropriate form of the forward model. One example is the linear model which I have shown and which is the first case we will be considering so if  $x$  was a vector  $x_1, x_2 \dots x_n$  then  $\hat{y}$  could be this which is a simple linear model, the fourth thing you have to decide is what is the form of the loss function ok one simple loss function called the least square loss function is simply  $y - \hat{y}$  squared ok so this is simple cost ok.


And finally you have to decide how you are going to do the optimization ok so I like I said we will be sticking to gradient descent and its variants an important thing to note is that you will have to give the hyper parameters alpha etc is that machine learning is not magic ok so there is no magic going on here it requires a lot of input for example as we have discussed earlier this might require domain knowledge (21:08) try to take this away but really even there it is as we will see later on in the application things it is usually very-very useful if you have some amount of domain knowledge ok.

(Refer Slide Time: 21:22)



### A look ahead at Deep Learning – 1

- We will be looking in the next few weeks at various types of hypothesis functions  $\hat{y} = h(x; w)$ .
- Each of these have their own purpose and domain where they work well
  - Linear Regression – For simple polynomial regression problems
  - Logistic Regression – For two-class (binary) classification problems
  - Deep Neural Networks – For any general, non-linear problem
    - There is a theorem that assures us that sufficiently large neural network will approximate any function
  - Convolutional Neural Networks – For vision/image based problems
  - Recurrent Neural Networks – For sequential/time-series problems
- There are also appropriate loss functions for each.
- It is possible that you might have a better model than these for your problem. The general procedure outlined here remains the same.



So let us look ahead at what will be doing in deep learning so will be looking at specifically various different types of hypothesis functions ok. Now that we know that all that is required in order to learn  $w$  is have a hypothesis and to learn it will now be looking at various types of

hypothesis. So each of these have their own purpose ok so the pipe of hypothesis that will be looking at have their own purpose and there is a specific domain where there will be work this is like any other model. So you can think of what we have what we will be discussing as various tools in your tool box ok. So just like you would not use a screw where something else is required, where a screw driver is required ok.

So you can have a hammer, you can have a screw driver, you can have a nut, you can have a bolt all this could be sitting in your tool box sometimes a spanner ok. Now each one of these are different and it is your intuition and your knowledge of the domain that lets you use an appropriate thing at an appropriate place. So what we will be doing very often is tell you ok the use of a spanner is this, this is what it will do and you have to find out where it applies. Similarly will be discussing various models each of these models is a different model ok so linear regression so usually it is used for simple feet square feet's, logistic regression is used for classification problems.



Deep neural networks are used for almost any general non-linear problem but sometimes it might be overkill ok so the reason why deep neural networks work and probably why many people are in this particular course so is that we know that any function can actually be approximated by a deep neural networks or even by a reasonably shallow neural network ok it will approximate any function at all so we look at this next week ok then there are something called CNN's which will be week 6 and 7 of this course this is for vision based problems. Then you have recurrent neural networks for time series sequential problems ok so all these are like I said different tools in your tool box will be going through sequentially what the properties are of this.

There are also approximate loss functions for each and every case it is also possible this important warning for you that none of these might be the best models for your problem ok and it is possible that you might have some good idea of what the model looks like and still the overall paradigm that I am describing in this video actually works for you. So as long as you have some idea about the problem use as much knowledge as you have will come to some concluding remarks towards end of the deep learning section on where to use it and where not to use it but please do remember use as much knowledge as you have about the fundamental problem that you are trying to solve usually that is a good idea.

(Refer Slide Time: 24:14)

## A look ahead at Deep Learning – 2

- While looking at each of these models, we will be specifically discussing the following aspects each time.
  - What is the mathematical expression for  $h(x; w)$ ?
    - Linear regression --  $\hat{y} = h(x; w) = Wx + b$  → sigmoid (non-linear)
    - Logistic regression --  $\hat{y} = h(x; w) = \sigma(Wx + b)$
  - What is a good loss function  $J(y, \hat{y})$  for this model?
    - Least squares  $J(y, \hat{y}) = (y - \hat{y})^2$
    - Binary cross entropy  $J(y, \hat{y}) = -y \ln \hat{y} - (1 - y) \ln(1 - \hat{y})$
  - What are the efficient ways of calculating the gradient  $\nabla_w J$ ?
    - The **backpropagation algorithm** for neural networks is an example of this
  - We will start our discussion with linear regression in the next video

So why we are going to look at each of these models will be specifically discussing the following aspects ok. So what is a mathematical expression for  $h$  of  $x$   $w$ ? This is the first thing that will be discussing. So linear regression for example is simply if you have  $w$  and  $x$  a linear model  $w \cdot x$  plus  $b$  logistic regression is this function is called  $\sigma()$  (24:36) will be looking at details of this later this week this is a non-linear function. You will see that it looks similar to linear regression and this is the usual process will follow. A linearity followed by a non-linearity that is logistic regression. Next thing will be looking at is what is a good loss function? Remember your  $y$  and  $\hat{y}$  will usually be different and you want to find out you know how do I tell the machine that I have not got the answer that I want it and that is usually through the loss function.

So the loss function if it is non-zero tells you that there is a difference between your hypothesis and ground truth. So a least square function is  $(y - \hat{y})^2$  this looks a little bit more complicated but it is actually a fairly simple function will be looking at this, this is called binary cross entropy ok then we will be looking at what are efficient ways of calculating the gradient, since you are going to get a gradient when we do gradient descent then we do optimal parameters search. So for neural networks the algorithm which we use is called the back propagation algorithm. So in the next video we will be starting our discussion with the first model for this course which is linear regression the linear model, thank you.