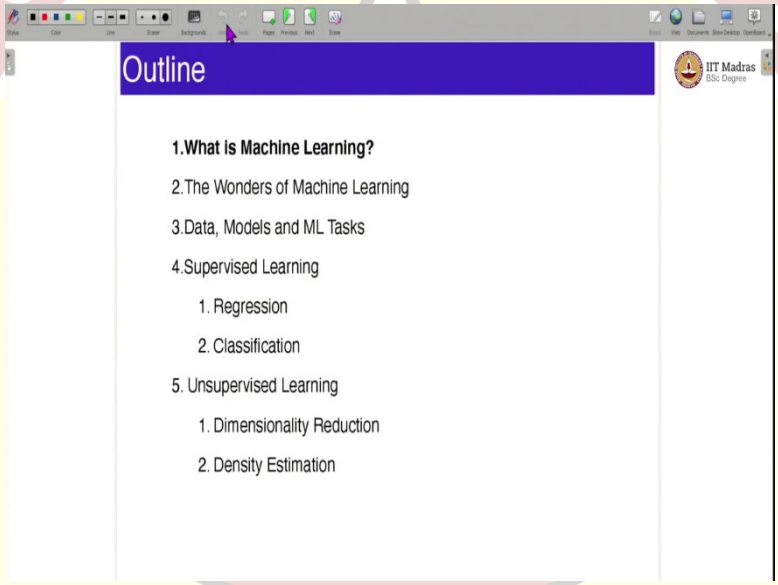
IIT Madras

ONLINE DEGREE

Machine Learning Foundations
Professor Harish Guruprasad Ramaswamy
Assistant Professor, Department of Computer Sciences & Engineering
Indian Institute of Technology Madras
What is Machine Learning?

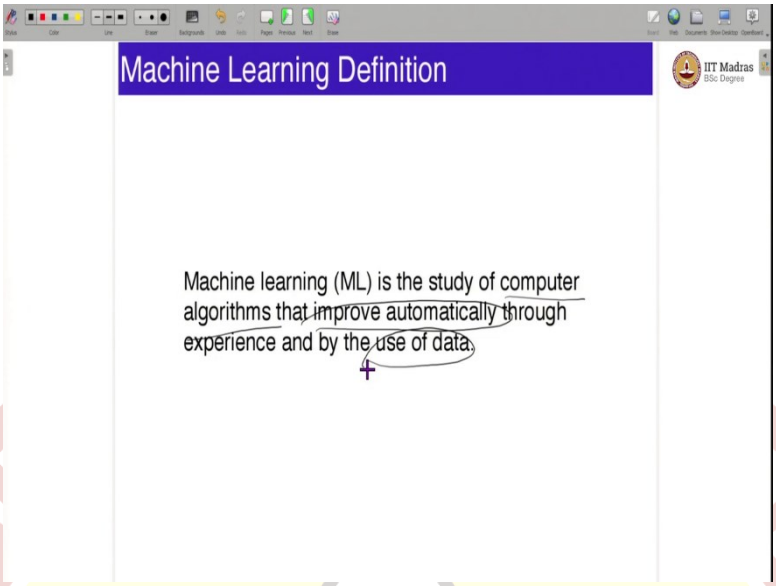
Hello, students, welcome to the course on Machine Learning Foundations. I am Harish Ramaswamy, and I am one of the three instructors who is responsible for teaching this course. The first week, we will take a brief look at the basic foundation, basic terminology and introduction of this course and set up some problems, which will be alluding it throughout the course.

(Refer Slide Time: 00:42)



First, here is a brief outline of the content for this week. The first question that everyone should be asking is, what is machine learning? You must, might have heard of this term, so bandied about in the press in the newspaper, in the TVs, but what is machine learning?

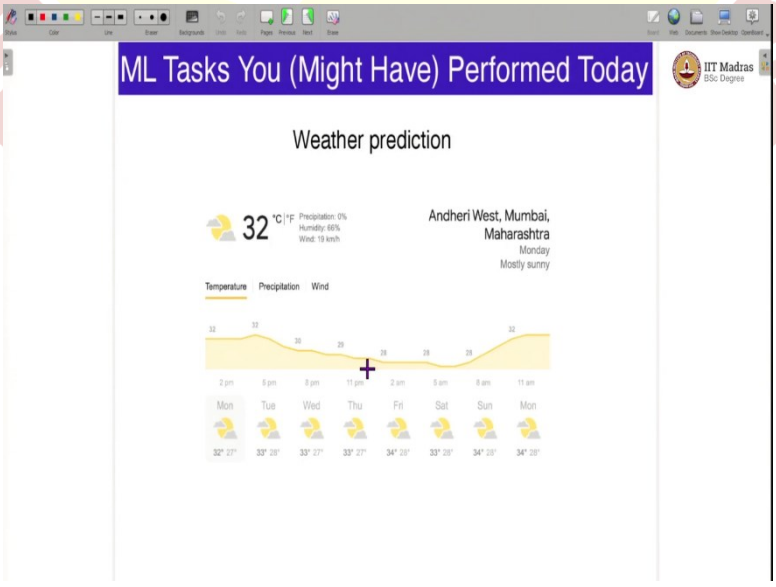
(Refer Slide Time: 00:57)



According to Wikipedia, this is the definition that you can find in Wikipedia. It says machine learning are, which is popularly abbreviated as ML, is a study of computer algorithms that improve automatically through experience and by the use of data. This is a pretty concise definition, and it does a pretty good job, but it uses more terms than can be understood easily.

For example, what is a computer algorithm? And what do we mean by improve automatically? And what exactly is the use of data? These terms are not fully clear, even though they are reasonably approximately clear. So, we will throughout the rest of this course we will make these terms clear.

(Refer Slide Time: 01:41)



Before we even make that attempt to just to impress upon you the importance and the ubiquity of machine learning tasks even before coming and watching this video today, you might have performed several machine learning tasks. I will give two examples. One example might be weather prediction.

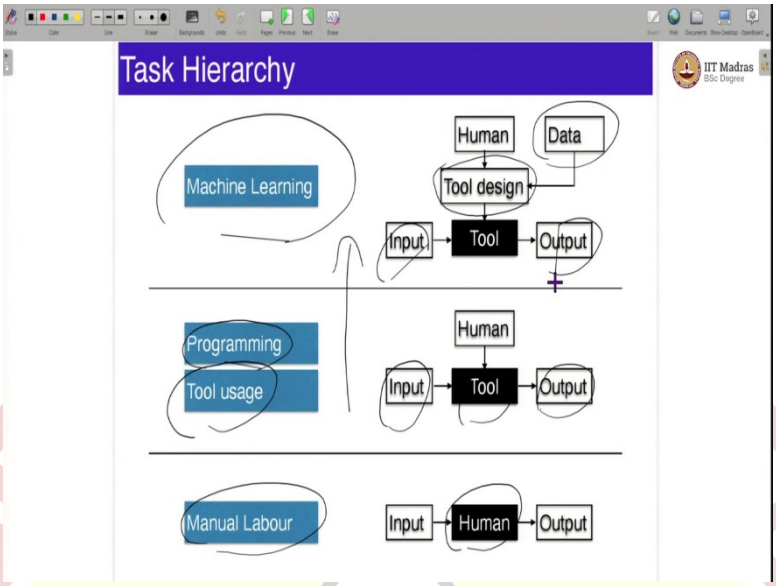
You might have just after you woke up you might have just Googled what is the weather now. And Google or some other weather site might have given you an output like this. While machine learning is by no means the only tool that is used for giving weather prediction it is a major reason why weather prediction is so much more successful these days than let us say 20, 30 years ago.

(Refer Slide Time: 02:25)

Here is one other example of a machine learning tasks that you might have performed today even before let us say watching this video. Let us say you wanted to take a group picture of your friends and family. And any reasonably advanced camera software these days would automatically detect the faces in the group of people, and it might put boxes so that it is easy for you to focus, it is easy for the camera to focus.

And autofocus would automatically focus on the faces in the image because most people who take pictures of faces, want the faces to be in focus and not the background. Machine learning is an important and in fact the most important tool that is used for doing this kind of task. I have been saying the word task several times over and over. What do we mean by that?

(Refer Slide Time: 03:18)



Task has an English meaning, as it is, but what when we say task. We mean it in the sense that a task is a process, which converts an input into an output that is the notion of abstraction that we will be working with. A task is something which converts an input to an output. So, a task can be performed at various levels.

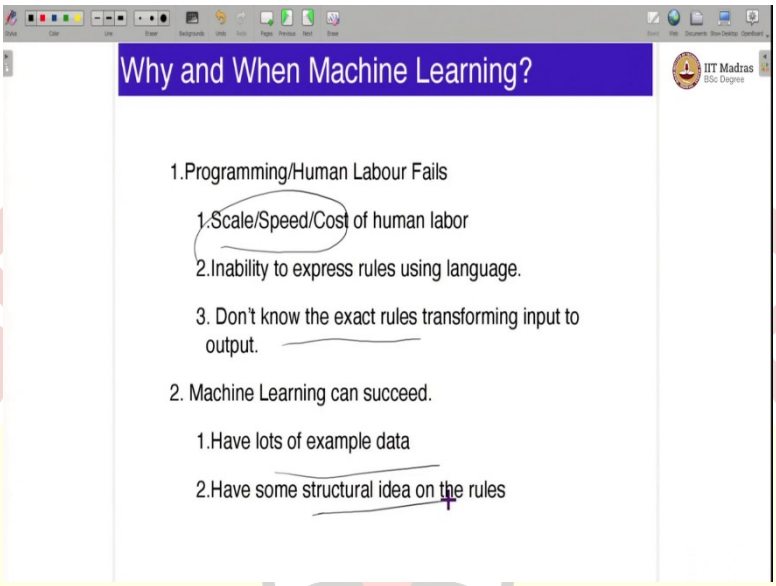
The lowest level is the level of manual labor where a human directly operates on the input and converts it to an output. So, that is the lowest level of that any task can be performed that is a manual labour. One level above that you have tool usage where the human does not directly operate on the input to create output, but rather the human constructs a tool, which in fact, operates on the input and converts into an output.

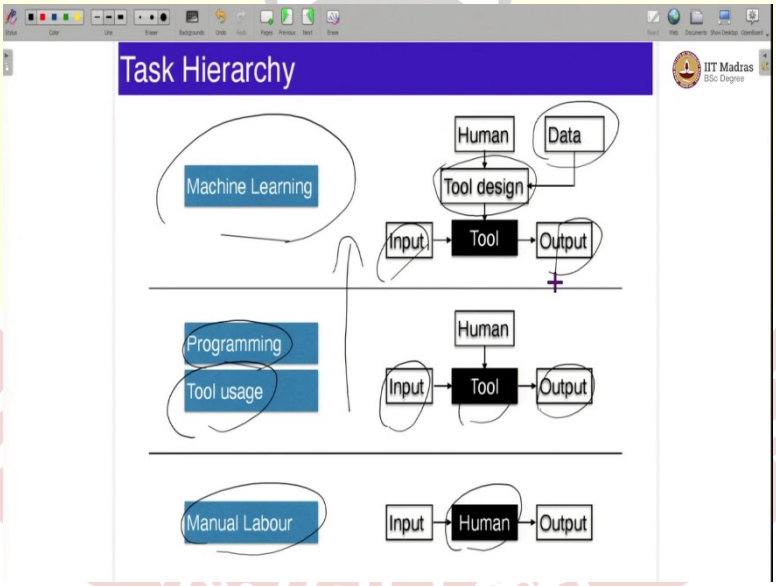
A special case of tool usage is simply programming where the tool happens to be a computer. And the way the human creates a tool is typically through programming and software. Then you have the next level of abstraction, which is machine learning. Here, the human actually does not even build the tool himself. What the human does is, actually just use a broad blueprint to a tool design box, which in fact, constructs the tool.

It also uses an additional source of data along with human inputs to make the tool. So, you can clearly see that the higher up you go the level of abstraction is increasing. At the lowest level the human directly operates on the input and converts it to the output at the highest level you have version where the human actually does not even construct the tool which converts the input into output.

The human just gives broad instructions to a tool design, which in fact uses the data and constructs the tool, which converts the input into an output. All of this might sound a little abstract. Now, we will give several examples and make this clear.

(Refer Slide Time: 05:17)





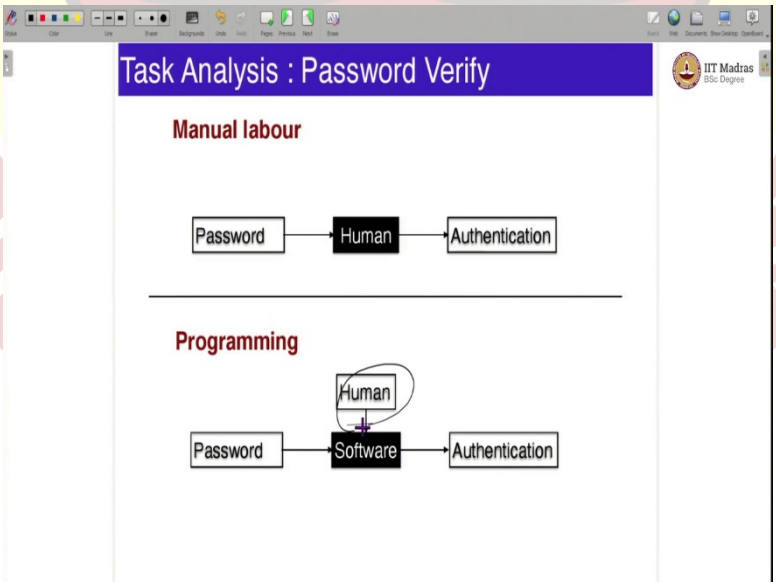
The first question before we do that, you should ask why should you use machine learning at all? Machine learning seems significantly more complex than manual labor or tool usage. Why should we even use machine learning? The answer to that is if you can use programming or human labor to do a task, you should not use machine learning, you should use simple programming or human labor.

But there are several situations where programming or human labor will just fail. The main reason why human labor would fail is because it just does not possible to give, achieve the task at the scale, or the speed or the cost that is available to perform that task. This is the common reason why programming might fail in performing a task is that you might not be able to express the rules that convert the input into an output using say computer programming language.

Or in some cases, you do not even know the exact rules that transform the input into an output. So, these are all reasons why programming or human labor can fail at a task. But that is not sufficient for machine learning to succeed. For machine learning to succeed, we need one other thing, which is we need to have sufficient hope that machine learning can succeed.

And one of the main reasons we think machine learning can succeed is because if it has access to data. If you have access to a lot of example data, then machine learning can reasonably succeed. In addition to that, it is also nice if you have some structural idea on the rule. Even though you do not know the exact rule that transforms the input into an output, you must have some structural idea on the rules. So, all these points will make clear when we look at several examples now.

(Refer Slide Time: 07:05)



Let us consider a very simple task, first. Let us consider this, let us call this task password verify. So, let us say you are Google now, and the task is to verify the password input by some human somewhere. Can this be performed? Can this task be performed at the level of manual labor?

Conceivably yes, that is, when someone types a password, the password gets sent to a server and there is a human who is actually looking at the server and just checks the password and then sees whether the password matches the password on file and then gives authentication. It is in principle possible for a human to do it.

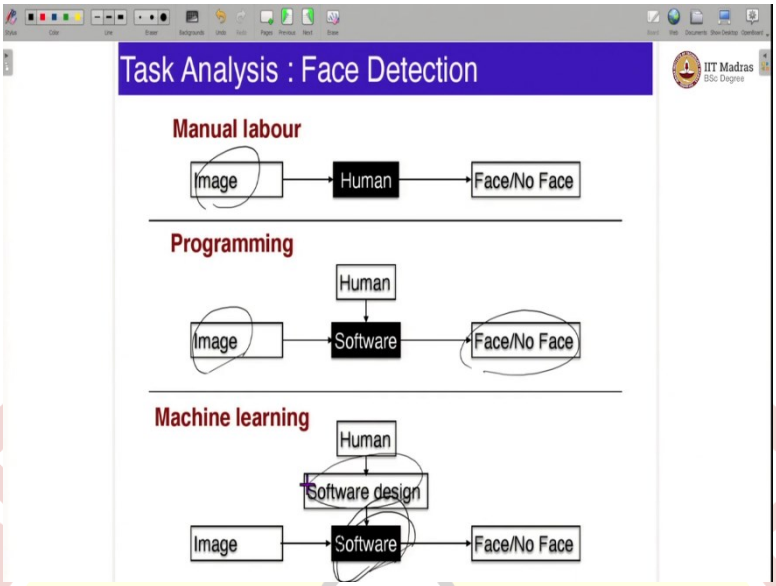
But you can clearly see it is not possible in scale that is we will need billions of humans just to do password verification, which is not clearly possible. And humans are more likely to make errors and so on. Then the, so this particular task is not suitable for to be performed at the level of manual labor. Can this task be performed at the level of say programming?

And in fact, the answer is yes. Because a human does not need to do this task of converting a password input into an authentication output directly. You can just write a piece of software, which in this case is the tool, which just checks whether the password that is input matches the password on file, and if it does, it gives the authentication and that well you can now log into your ID or if it does not match, then it can say well authentication is denied.

This is a particularly simple piece of software that a human can directly design and directly write. There is no need to go to higher level of complexity or abstraction where a human does not even directly write the tool, but because a human can easily write the tool in this case and can easily create the tool for solving the problem of this particular task.

So, this is an example of a task which is not suitable for manual labor, but it can be solved using programming. Because it can be solved using programming, there is no need to even consider going into the level of abstraction of machine learning.

(Refer Slide Time: 09:09)



Now, let us consider another task that we just discussed now, that is the task of face detection. In this case, the input let us imagine the thing the camera itself as the, let us say you are the camera itself, what is the input and what is the output? Well, the camera has to take the image as the input and it has to say whether well this particular image has it a face, does it have a face or not? If it has a face, where is the face?

It should, that is the output of this particular task. Can a human do this? Conceivably, once again, yes, a human can do it, that is, whenever you take your new Canon camera and you focus on a group photo that particular, whenever you press the button, whenever you are about to press the button that particular image is sent on a server to let us say where there is a human looking at all the photos that are about to be clicked, and he would just go and draw boxes on those images.

And those boxes are then somehow conveyed to the camera and then the camera once again displays the box. It is conceivable, it is possible to do it, but it is clearly not feasible or efficient because there are just not enough number of humans to draw boxes on all possible images. So, that rules out manual labor. Can this be performed using programming?

By that we mean that there is a human who writes a piece of software directly onto the camera. And what does this software supposed to do? Well, this software is supposed to take or let us say a programming function which is supposed to take an image as input and it is supposed to output the location of the faces in the image.

Is this possible? I mean, can a human write a program which will do this? Immediately you can see that it is not straightforward to write a function which does this. Why is that? Because while humans have an idea of what constitutes a face, they cannot really write it in a mathematical, mathematically precise way, what aspects of this pixels in an image constitute a face and which does not constitute a face.

You can easily, let us say, if you are teaching the concept of a face to a baby, you should, you would be able to easily do that. You can point to a baby and say this is a face, this is not a face, this is a face and this is not a face, and the baby would pretty quickly understand it that these are faces and these are not faces. And when you give to the baby an image containing several people the baby can easily identify which where the faces are in that image.

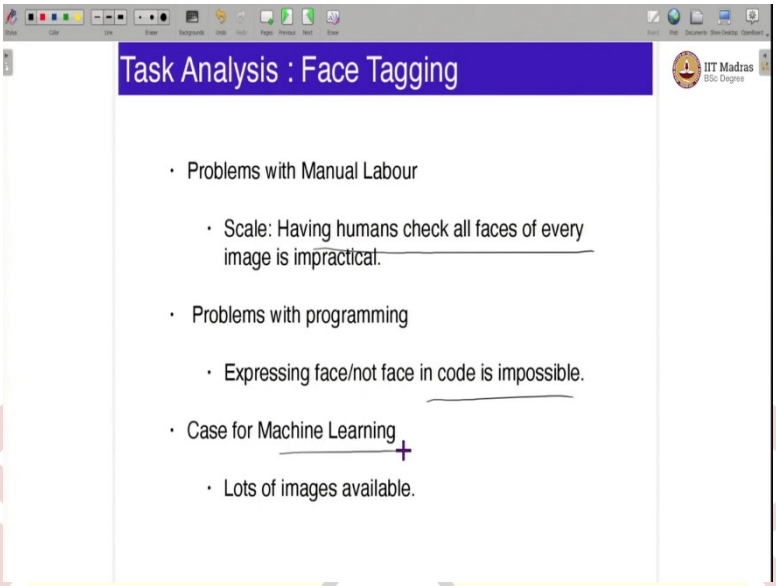
So, that is you know what a face is. It is just that you cannot convey that idea to a computer efficiently because computer does not look at it. As far as the computer is concerned, every image is simply a collection of pixels. The idea of a face cannot be conveyed efficiently to a computer, which just knows pixels, even though there exists, there clearly exists a function which can do that.

There exists a function, which can take in an image which just constant, which is just made up of pixels and say where the faces are, that clearly exists such a function, but just that a human cannot directly write a function because he is not able to convey the idea of a face to a computer directly. So, this rules out programming also.

Now, this is an example case where machine learning can make a huge advantage. Because here, the human does not need to even know, does not need to exactly convey the idea, the exact tool, because the human is not actually building the tool. The human is just giving the broad paradigms of the software design and the software is not directly built by the human.

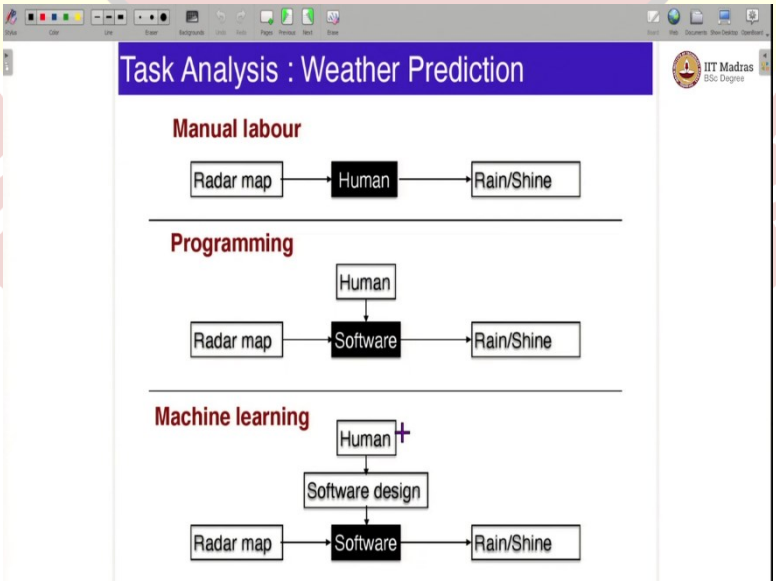
Of course, you need lots of data. And this is one example where face data is, face image data is practically unlimited on the internet, so there is no dearth of face data. So, you have a lot of hope of success with machine learning because there is lots of face data and humans do not need to exactly they do not have to build the tool from scratch because they do not have to write the software themselves, they only have to give inputs to a software design, which is what creates the software that converts an image into a face.

(Refer Slide Time: 13:30)



So, here is a summary of the problems of the face tagging task. That is the problem with manual labor is that having humans check all phases of every image is just impractical. And the problem with programming is that expressing face or not face in code is just impossible. The case for machine learning is that you do not have to express this face not face in code, as a function of pixels directly you just have to give lots of images, which are faces and lots of images which are not faces. This is sufficient for a machine learning paradigm to work.

(Refer Slide Time: 14:10)



Here is one other final task that we will be analyzing, which is considered the task of weather prediction. So, weather prediction, what is the input and what is the output of the task. The

input of the task is, a radar and map. Let us say, you want to predict the weather or let us say, you want to predict whether Mumbai is going to get rain or not the next day, that is your goal. And what is the input for making such a prediction.

Let us say you have the radar map of let us say the entire Arabian Sea, Bay of Bengal and India and so on. So, you have the radar map of containing the information about the wind, about the moisture and all that, you have that as the input for you and you want to predict whether tomorrow Mumbai is going to get rain or it is going to be sunny, that is the goal.

And in fact, in this case, it is not even possible for humans to do it because humans do not know, given the radar map, whether it is going to rain or not. Because there clearly exists such a function because physics says that there exists because whether it is going to rain or not tomorrow clearly depends on the moisture, the wind and other conditions that are prevalent today.

So, there clearly exists a function, which is what is implemented by nature when it decides to rain or not the next day, but just that humans do not know this function. So, in fact, in this particular case, the human cannot actually, this particular problem is not able to be solved using manual labor, because the human does not know the function which maps the input to the output.

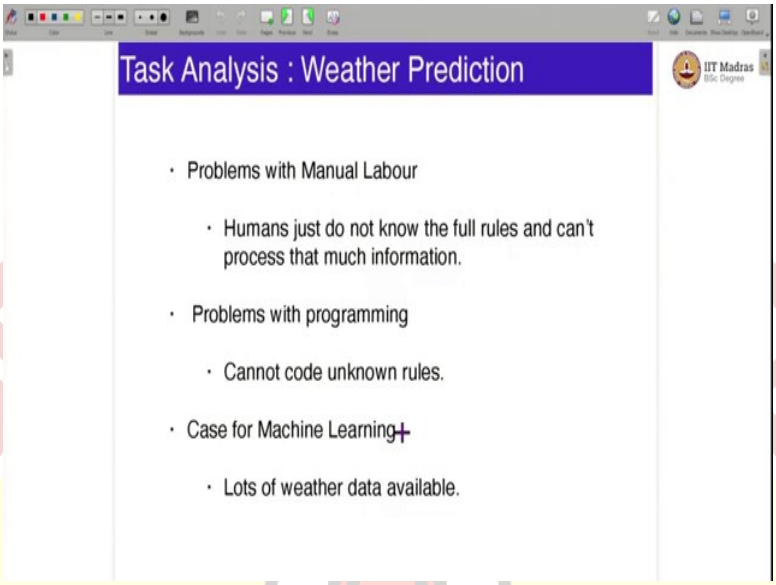
It is for that exact same reason programming will also fail, because if the human does not know what transforms the input to the output there is no way he or she can convey that idea to a computer. So, programming, this particular task cannot be solved with the level of programming as well.

At the level of machine learning, though, immediately it becomes possible because the only requirement is that there exists a rule which transforms the input, which is in this case the radar map to whether it is going to rain or not the next day, and we know such function exists because nature itself is implementing such a function. You do not have to know that function, you do not have to be able to convey that function to a computer and so on.

So, the only thing you need to know is that, there exists such a function. And in fact, you know that exists, so you do not have to worry about the problems that you encounter with manual labor and programming. Once again, the human just has to give the broad parameters of the software design, which in turn designs a software and uses some additional data of weather

maps and creates a software which will convert the let us say the local radar map into whether tomorrow we are going to get rain or shine.

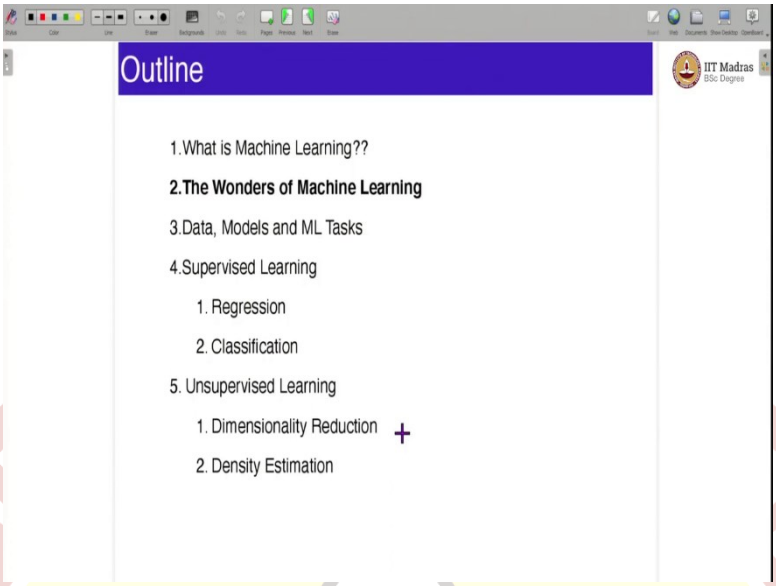
(Refer Slide Time: 17:09)



So, this is another example of a case where machine learning is an example of a good machine learning task, that is, you cannot solve it with manual labor and you cannot solve it with programming also, but you can solve it in machine learning because there is hope for solving it with machine learning, because there is lots of weather data available.

And in fact, most weather prediction systems do use machine learning in some way, because of the immense amount of availability of weather data and it is just not possible to predict weather using either manual labor or with just simple programming. So, with that, I think you have a broad idea of what is machine learning and when you can apply machine learning.

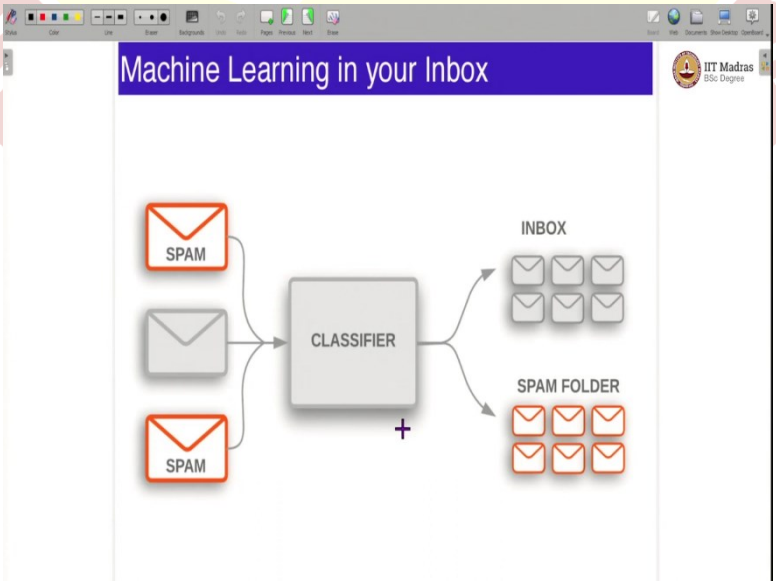
(Refer Slide Time: 17:56)



Then the question becomes, where is machine learning being used? This section is mostly an advertisement of machine learning. Machine learning really does not require an advertisement considering the ubiquitous nature of it, and you just cannot go even a few minutes without encountering machine learning, but I will still go ahead and do that to let you know where all machine learning can be formed.

This is by no means an exhaustive list. There are several applications. Any application that you can think of has a potential use case for machine learning. So, this is by no means an exhaustive list, but this is a couple of examples that you can find machine learning.

(Refer Slide Time: 18:35)



Let us get the easy example out of the way first, which is an example that you encounter every day in your email inbox. Any reasonable inbox system like Gmail or Hotmail or any reasonable email system actually has a very good spam classifier in it, that is, every time you receive an email the mailbox itself classifies that email as either spam or a regular email.

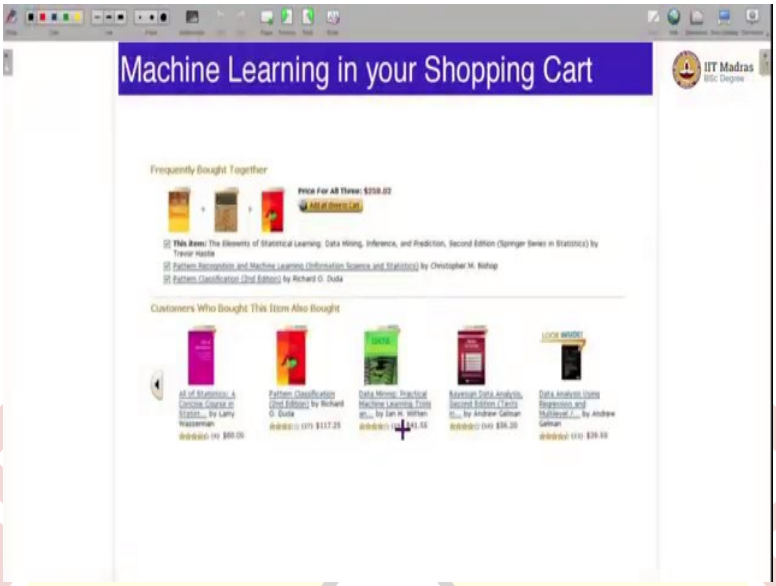
And all the spam emails get into a spam box, and all the regular emails just go to the inbox. In fact, this is one of the biggest success cases of machine learning where machine learning has largely solved the problem of spam. Even let us say 10 years ago, spam was a huge problem, that is, for every valid email that you get there would be 5 to 10 spam emails, which you would have to sift through to get to your regular email, but machine learning has tamed the problem of spam and it has successfully done so.

And why was this problem not? We can ask the question this is a machine learning task. Why cannot it be solved using human labor? Yes, it can be solved using human labor, but the problem with human labor is that it has the problems of scale and speed. The problem with programming is once again similar to the problem with the face detection, where, given an email how do you say with the program that this particular email is spam or not spam.

Because spam has the righty of characteristics, you cannot do any simple function which, just write a function which takes in an email and says whether it is a spam or not. There is no simple function, which can do that. Given an email, you can say whether it is mostly you can say whether it is spam or not, but it is just not possible to convey this idea to a computer.

But once again, this is easily solved using machine learning, because you do not need to convey that idea at all. The only thing you need to give to the system is lots of data, which in this case boils down to giving it examples of lots of spam emails, and examples of lots of non-spam emails, that is sufficient for the machine learning algorithm to design the tool that actually takes in these inputs, which are emails and classify them as spam or not spam.

(Refer Slide Time: 20:59)



So, and one other example that you encounter every day is in your shopping cart. Where you go to Amazon, and you say you buy a couple of books, so you will often see this extra thing, which will say customer who bought this item also bought these other books. This is an example of what is called a recommender system and recommender systems are another important task that is often solved using machine learning.

In fact, when you go to YouTube or when you go to Netflix, this is again, brought to it fore when you watch a TV show then you get a bunch of other TV shows that says people who watch this TV show who want who like this TV show also like these TV shows. And similarly for YouTube, when you watch one particular news video, it will automatically recommend other videos which other people similar to you have watched.

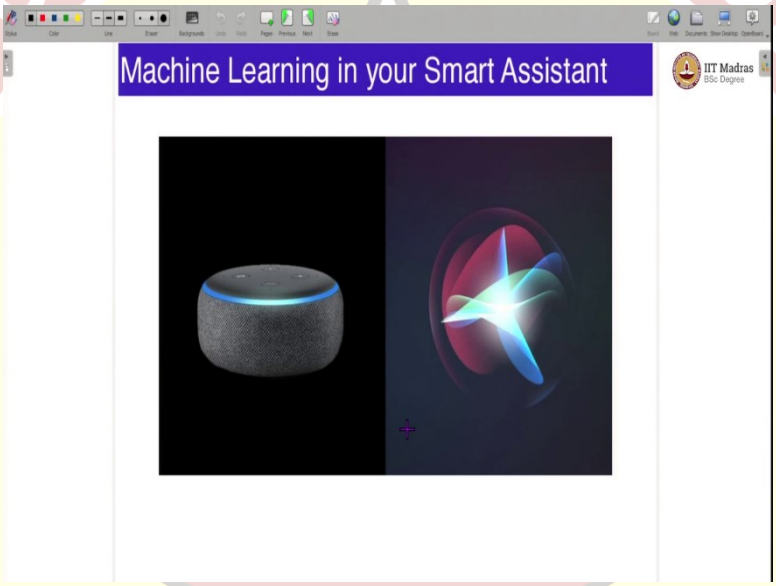
So, this is how does machine learning do that we will not be attacking that particular problem in this course, but that is an example of machine learning. Once again, can this be solved using humans? Potentially, yes, that is a human can just do a psychological analysis of each person, and then say, okay, this person likes these videos, so this is a type A kind of person, so he will like these kinds of videos also.

But you can clearly see this is not scalable to all humans, and it seems like a wild violation of privacy. And this cannot be done using programming because you might; there is no clear rules on which video someone else is going to like given that he has watched these other videos. Even though you might have a hunch, it is difficult to translate this hunch into at the level of programming where you can convey that to a computer directly.

But to solve this task at the machine learning level, you need neither. You just need to have examples of several examples of videos that this person has watched and other people have watched, and we just use that to come up with a tool which takes in the input that is your history of videos that you have watched our books that you have purchased and predict future videos that you will like are future books that you will like.

So, this is an input, output, which is not possible efficiently with humans and not at all possible with program and with the level of programming, but it is possible to solve this particular problem also at the level of machine learning.

(Refer Slide Time: 23:22)



Another example, that you might have encountered very often is see your smart assistant is your Alexa, is your Siri, is your Google Assistant and so, on all of these use machine learning very, very integral part. Right from the part that when you speak, your sound waveforms are converted into text and which in turn is converted into commands.

As far as the computer is concerned, a sound waveform is just a waveform. It is a bunch; it is a time varying signal, that is it. It is not, I mean, it is possible to solve this at the level of humans where you hide a small human inside every smart speaker and that small human decides whether what you are saying is Alexa or not.

But you can clearly see the ridiculousness of it, but you cannot, this solution is simply not scalable. But you so, well, can you solve this problem using programming. Once again, you run into the same problem that you had with face tagging, with spam detection and so on. While

you might be able to take here a piece of here a sound and say whether that sound was Alexa or not Alexa. How do you do that at the level of programming?

How do you write the program? How do you write a function, which takes in a time varying signal as input and output whether the word Alexa was said or not said in that particular signal? It is immediately the problem is that you cannot convey the idea of the sound Alexa to a computer efficiently.

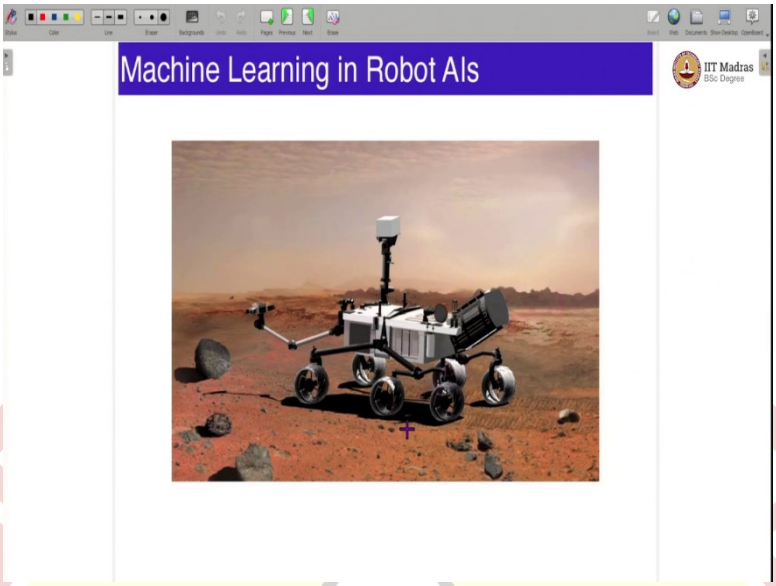
Once again, machine learning has no such issues with doing that because you only need to give several examples of Alexa and several examples of not Alexa, and the machine learning tool design will automatically do the job for you and it will convert the, it will construct a tool, which will convert your input audio into whether the sound was Alexa or not Alexa.

And of course, smart assistants use ML even more than this that is it, they do not just convert sound to text. So, they also convert text into commands that can be actually performed by the smartest. Let us say if you said 'increase volume by 10 percent'.

Let us say that is what you said, it has to first convert what you say that is increase the waveform corresponding to increase volume by 10 percent, into the words increase volume by 10 percent and then it has to detect that the command that you have given is corresponds to increasing volume by 10 percent, because not everyone might say it in the same way.

They might say things like, raise sound by one point. They might say, which might pretty much mean the same thing, colloquially, but how do you convert all of that into a reasonable command that an assistant can perform on the mobile or on any system. So, machine learning is used for that as well, machine learning is used for converting the text command, a text into a command that can be performed by the assistant.

(Refer Slide Time: 26:15)



And one other example, that used to be all the rage that used to be one of the main motivations behind developing machine learning itself was building a robot artificial intelligences. If you have seen any movie like Terminator or Robocop or iRobot or anything like that, you know that these used to be all the rage in pop culture.

And they are still one of the main goals of machine learning, but learning has formed so many other successes that it is not the only goal anymore. But it is still useful in several robots. For example, when you build, let us say, in this case, this is the I think the perseverance rover in Mars.

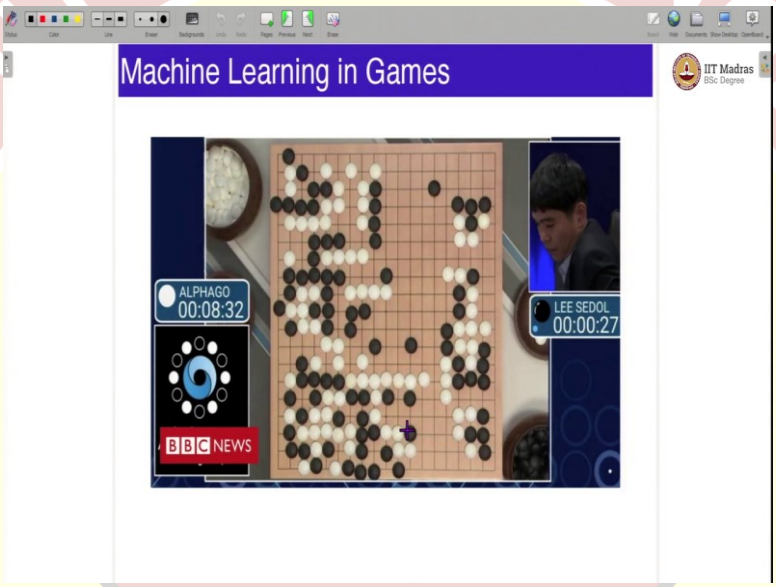
So, when you build a robot for, let us say, operating in Mars, it is just not possible for humans to do the job simply because it is going to take 10 to 15 minutes for just the radio signals from Mars to reach Earth. So, you cannot remote control the robot using, it is just not possible. So, there must be, you can do some level of programming, you can perhaps pre-build a route of let us say you want the rover to move from crater A to crater B.

You might have already computed the route and said, broadly, this is the thing that you should follow. But there are several unexpected things that can happen, and a programming just cannot handle it. So, for example, in this case, an input situation, input might be some environmental situation, and output might be what action to take.

The action might be, let us say, move forward, more left, stay there, switch on your parachute or huddle up or there might be several actions that a robot can do that might be the output and the input would be the current environment.

Once again, it is not possible to hard code program, hard code a program for all possible variations simply because you do not know what is going to happen, but machine learning can handle this to a certain extent, because you just need to give enough data to it so that it can effectively find the right or a good thing to do for every possible situation.

(Refer Slide Time: 28:26)



And one other recent huge success on machine learning has been in games. We know that in chess already in the late 90s there was human built chess system which beat the best humans, best human, just Garry Kasparov. But these systems have had even more success in recent times where for example, GO, was supposed to be an impossibly complex game that computers were not supposed to make for the next 10, 20 years.

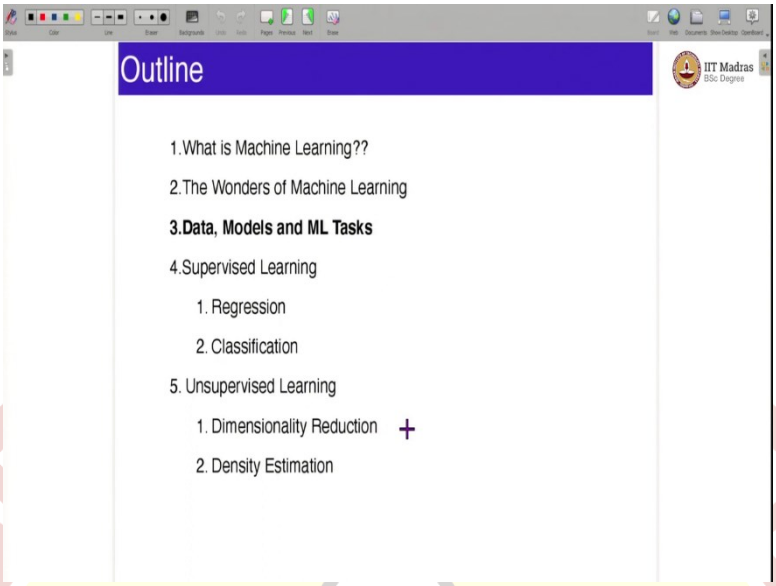
But suddenly, immediately the, this thing happened in the quite recently in just few years ago where a GO playing system which was primarily built around machine learning beat the best human GO play. And this is eventually going to happen for pretty much all games with complete information. Artificial Intelligence or machine learning agents are the most powerful players.

(Refer Slide Time: 29:28)

And in practical industry applications, machine learning is used in for example, in things like marketing. In social media marketing where you might have a limited budget. Let us say you have a budget of 100 dollars and Facebook is asking 10 dollars for a single ad, you might that means that you have a budget for 10 ads. But whom do you advertise to?

So, this is another, if you are a brand you have limited budget, but you want to reach as many people as possible, how do you decide whom to advertise to so that you can maximize your impact? So, this, how do you market across, let us say some social networks, how do you market across Twitter or YouTube or how do you do this efficiently is a very complicated question, and machine learning has made huge strides in solving such problems.

(Refer Slide Time: 30:20)



So, with that, I think this helps you to understand what is machine learning and when can you apply machine learning. And finally, you also know several areas where machine learning has been applied. So, this gives a brief introduction into machine learning. We will jump into the nitty gritty details of what makes machine learning work in the next section. So, for this, that is all we have for this video. Thanks.

Machine Learning Foundations

Introduction, Terminology and Setup

Harish Guruprasad Ramaswamy
IIT Madras

Outline

1.What is Machine Learning?

2.The Wonders of Machine Learning

3.Data, Models and ML Tasks

4.Supervised Learning

    1. Regression

    2. Classification

5. Unsupervised Learning

    1. Dimensionality Reduction

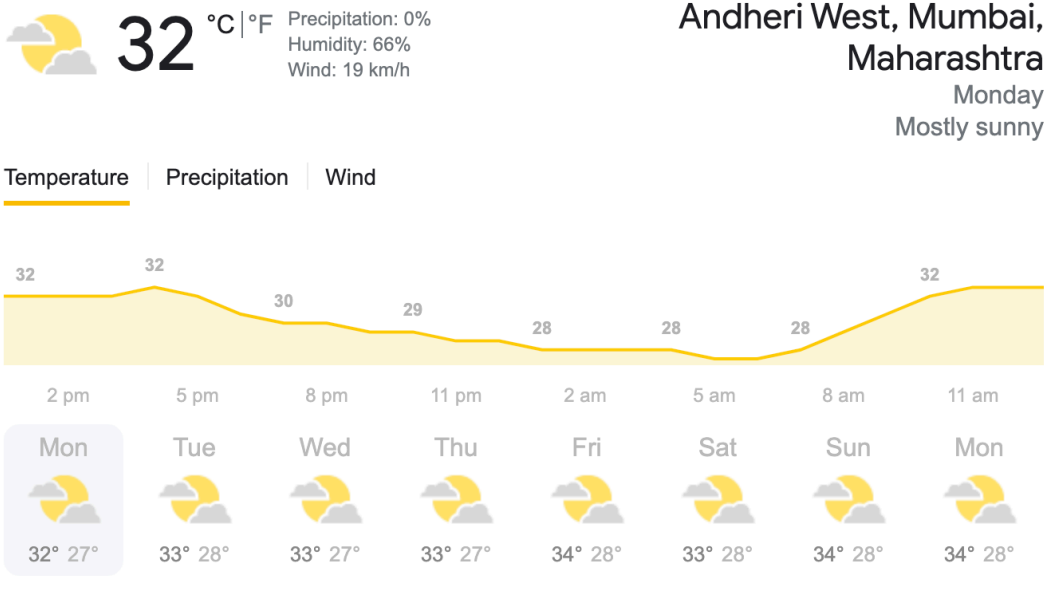
    2. Density Estimation

Machine Learning Definition

Machine learning (ML) is the study of computer algorithms that improve automatically through experience and by the use of data.

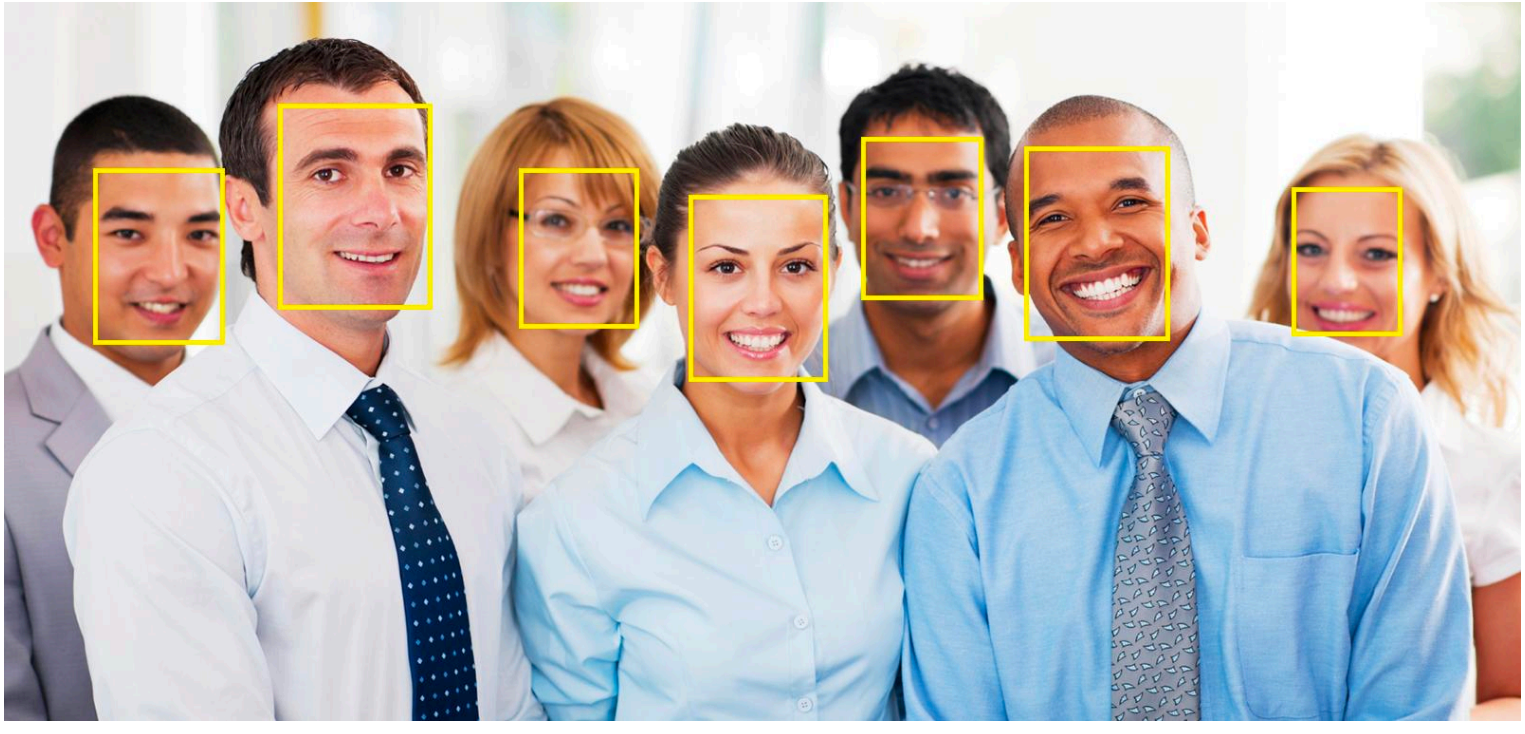
ML Tasks You (Might Have) Performed Today

Weather prediction



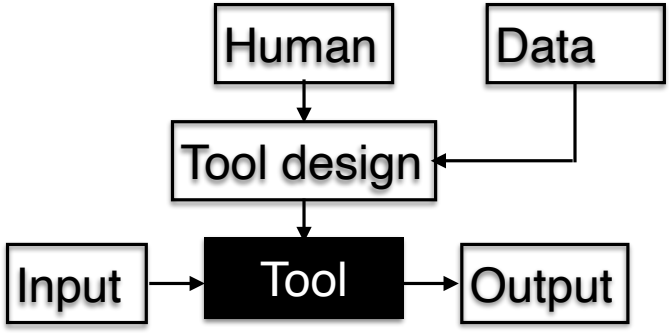
ML Tasks You (Might Have) Performed Today

Face Detection



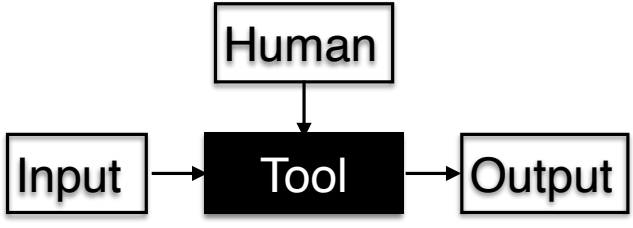
Task Hierarchy

Machine Learning



Programming

Tool usage



Manual Labour

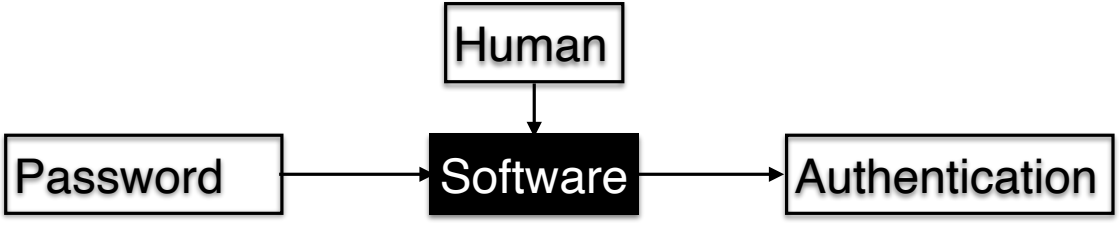
Why and When Machine Learning?

1. Programming/Human Labour Fails
    1. Scale/Speed/Cost of human labor
    2. Inability to express rules using language.
    3. Don't know the exact rules transforming input to output.
2. Machine Learning can succeed.
    1. Have lots of example data
    2. Have some structural idea on the rules

Task Analysis : Password Verify

Manual labour

Programming



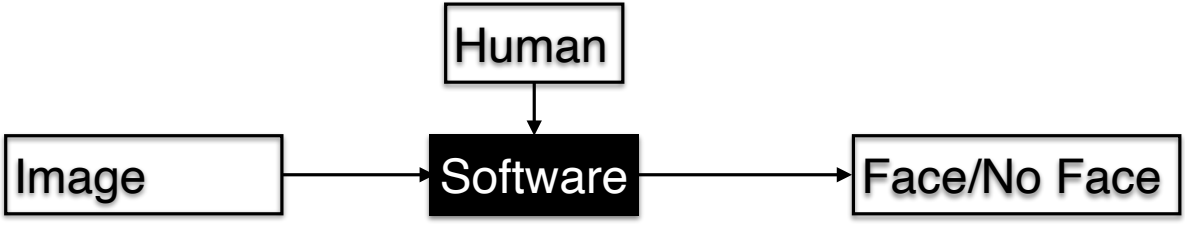
Task Analysis : Password Verify

- Problems with Manual Labour
    - Scale: Having humans check login details of every login is impractical.
- Problems with programming
    - None.
- Machine Learning not required.

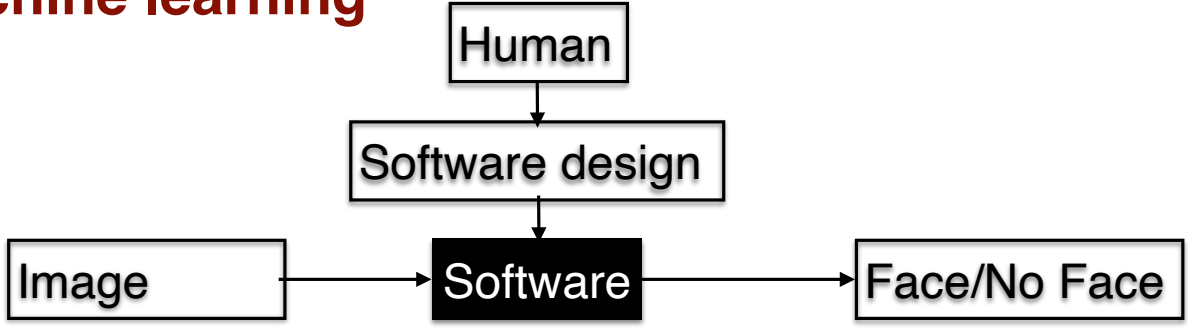
Task Analysis : Face Detection

Manual labour

Programming



Machine learning



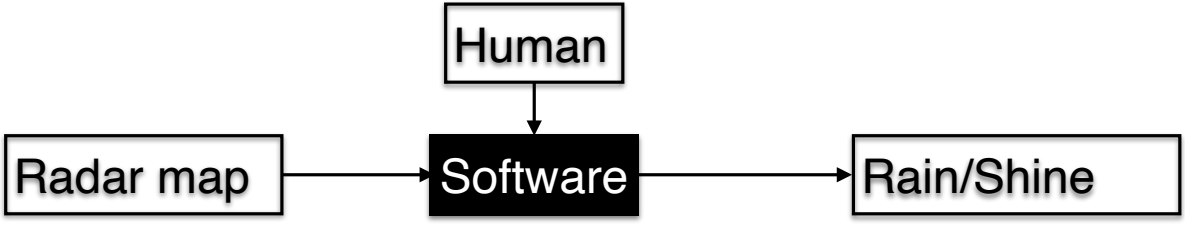
Task Analysis : Face Tagging

- Problems with Manual Labour
    - Scale: Having humans check all faces of every image is impractical.
- Problems with programming
    - Expressing face/not face in code is impossible.
- Case for Machine Learning
    - Lots of images available.

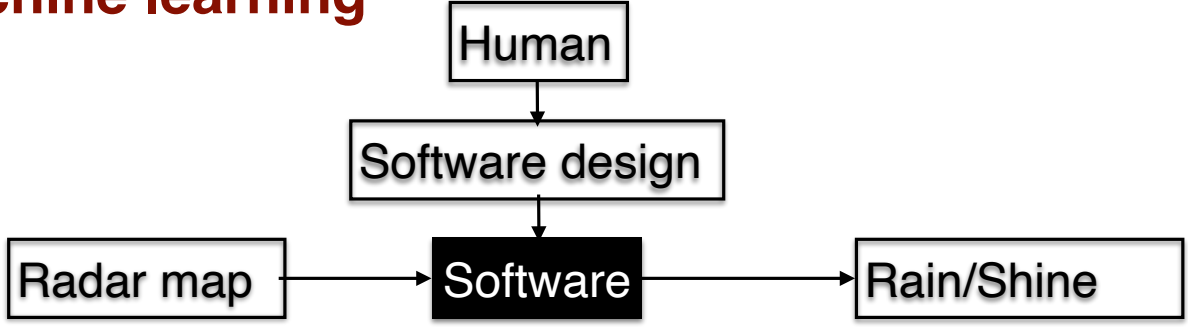
Task Analysis : Weather Prediction

Manual labour

Programming



Machine learning



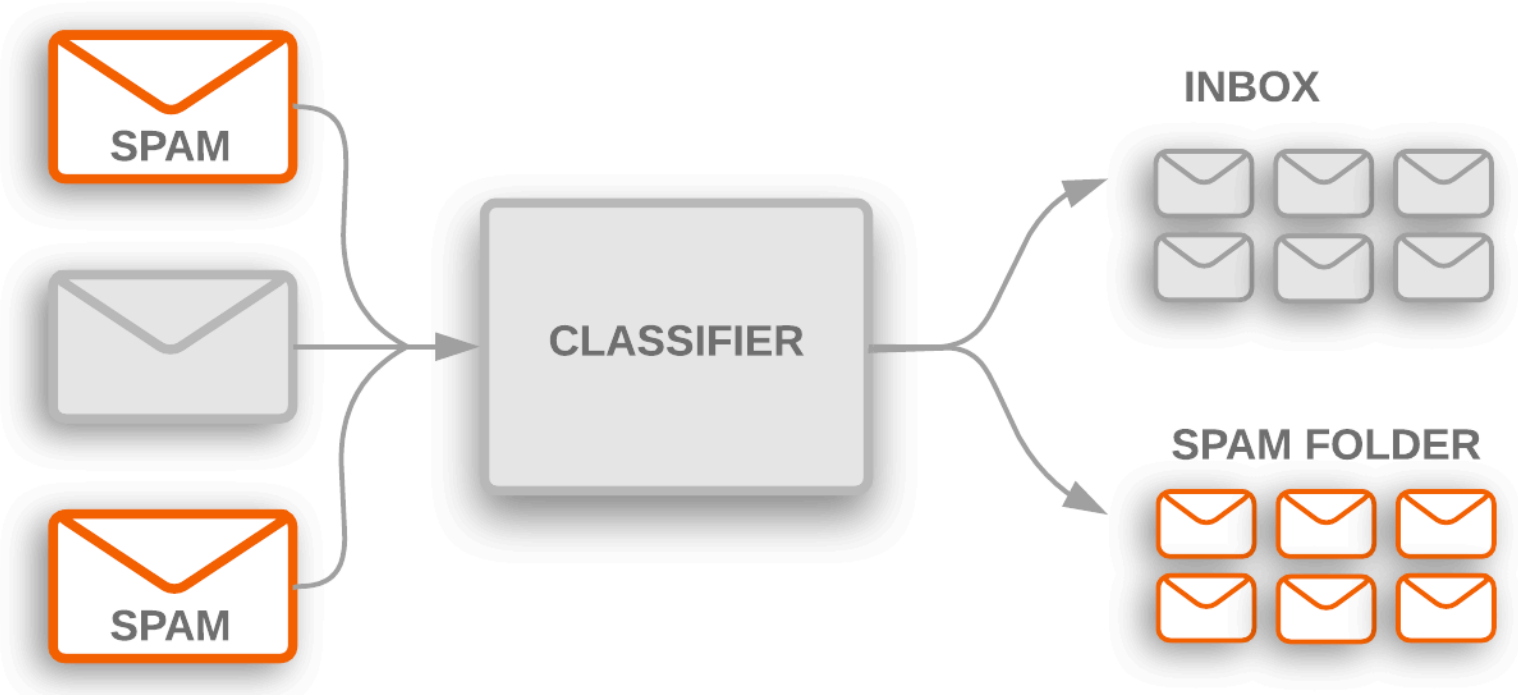
Task Analysis : Weather Prediction

- Problems with Manual Labour
    - Humans just do not know the full rules and can't process that much information.
- Problems with programming
    - Cannot code unknown rules.
- Case for Machine Learning
    - Lots of weather data available.

Outline

1. What is Machine Learning??
2. **The Wonders of Machine Learning**
3. Data, Models and ML Tasks
4. Supervised Learning
    1. Regression
    2. Classification
5. Unsupervised Learning
    1. Dimensionality Reduction
    2. Density Estimation

Machine Learning in your Inbox



Machine Learning in your Shopping Cart

Frequently Bought Together

Price For All Three: $258.02

Add all three to Cart

- **This item:** The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics) by Trevor Hastie
- Pattern Recognition and Machine Learning (Information Science and Statistics) by Christopher M. Bishop
- Pattern Classification (2nd Edition) by Richard O. Duda

Customers Who Bought This Item Also Bought

All of Statistics: A Concise Course in Statist... by Larry Wasserman

(8) $60.00

Pattern Classification (2nd Edition) by Richard O. Duda

(27) $117.25

Data Mining: Practical Machine Learning Tools an... by Ian H. Witten

(29) $41.55

Bayesian Data Analysis, Second Edition (Texts in... by Andrew Gelman

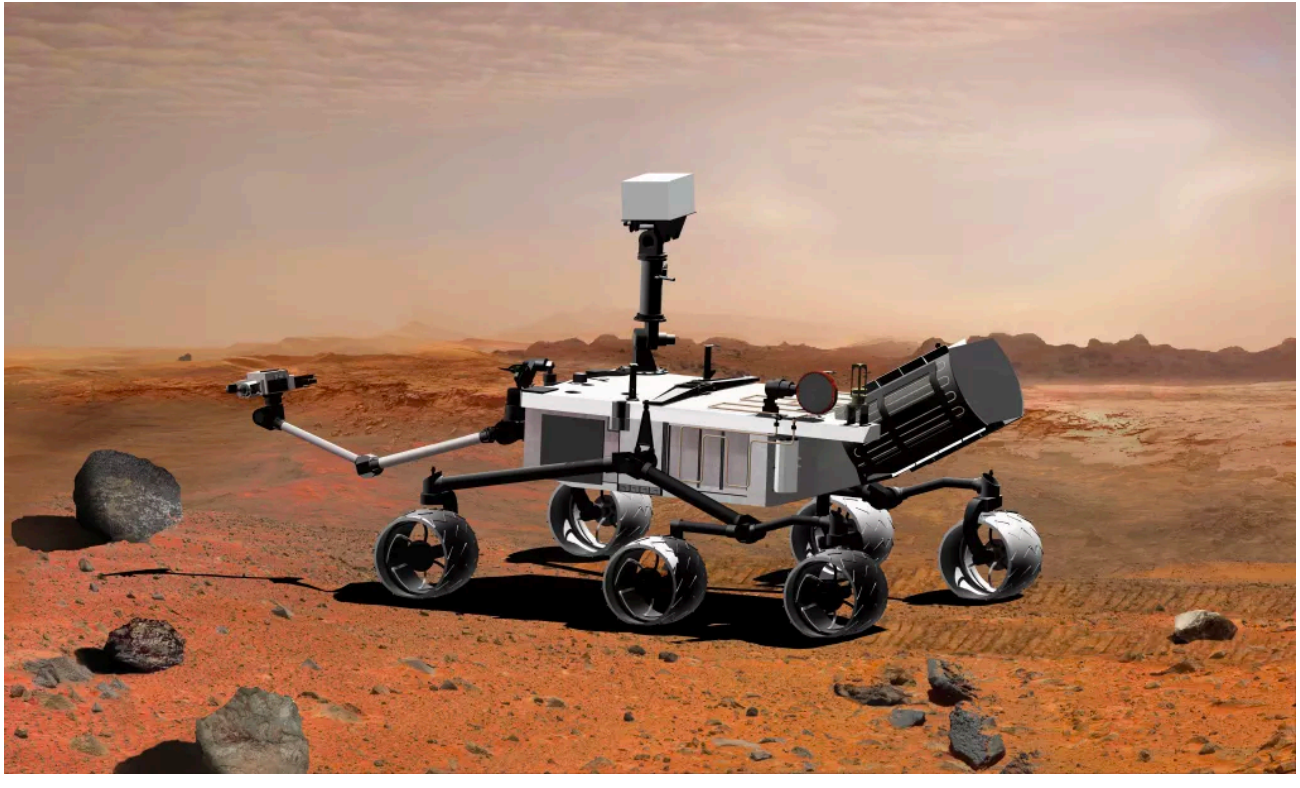
(10) $56.20

Data Analysis Using Regression and Multilevel /... by Andrew Gelman

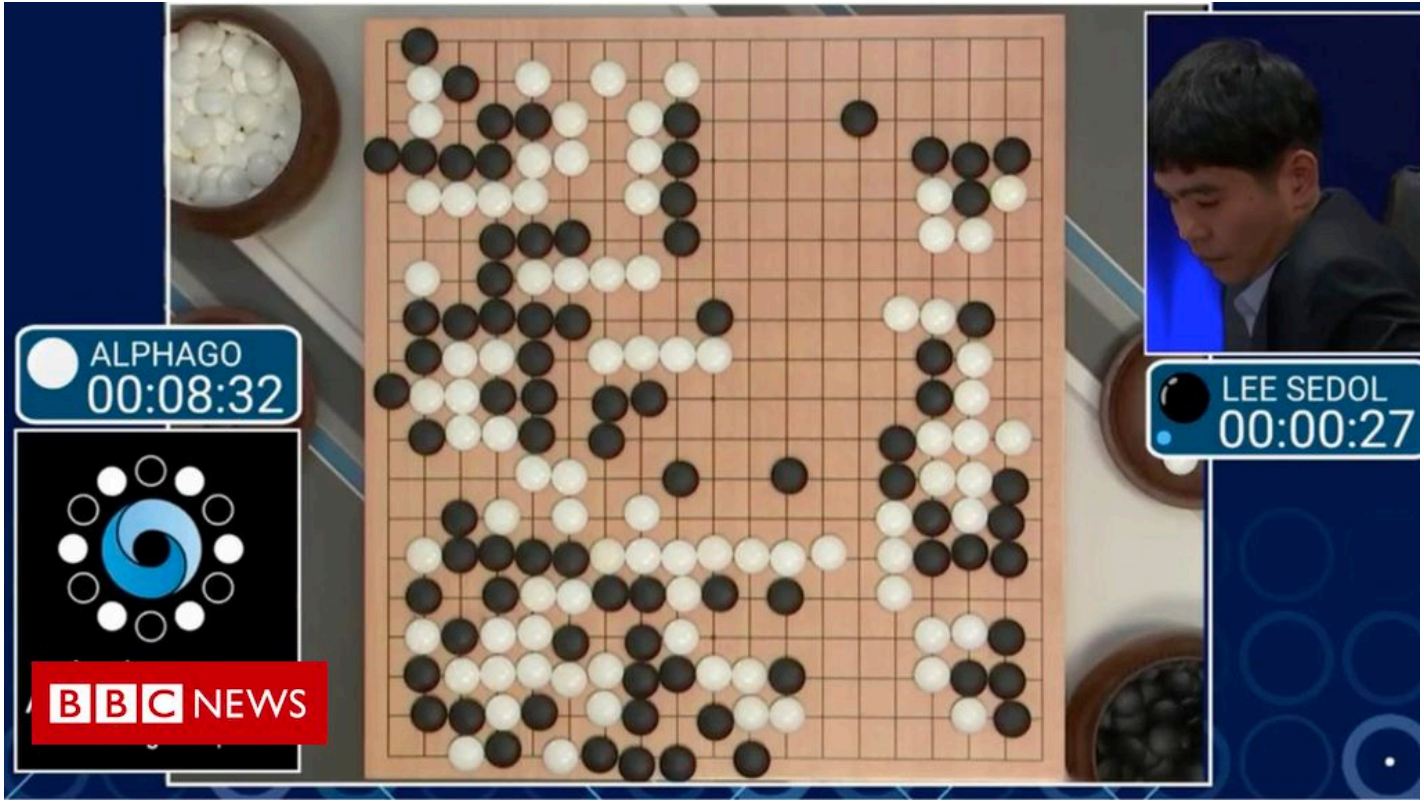
(13) $39.59

Machine Learning in your Smart Assistant

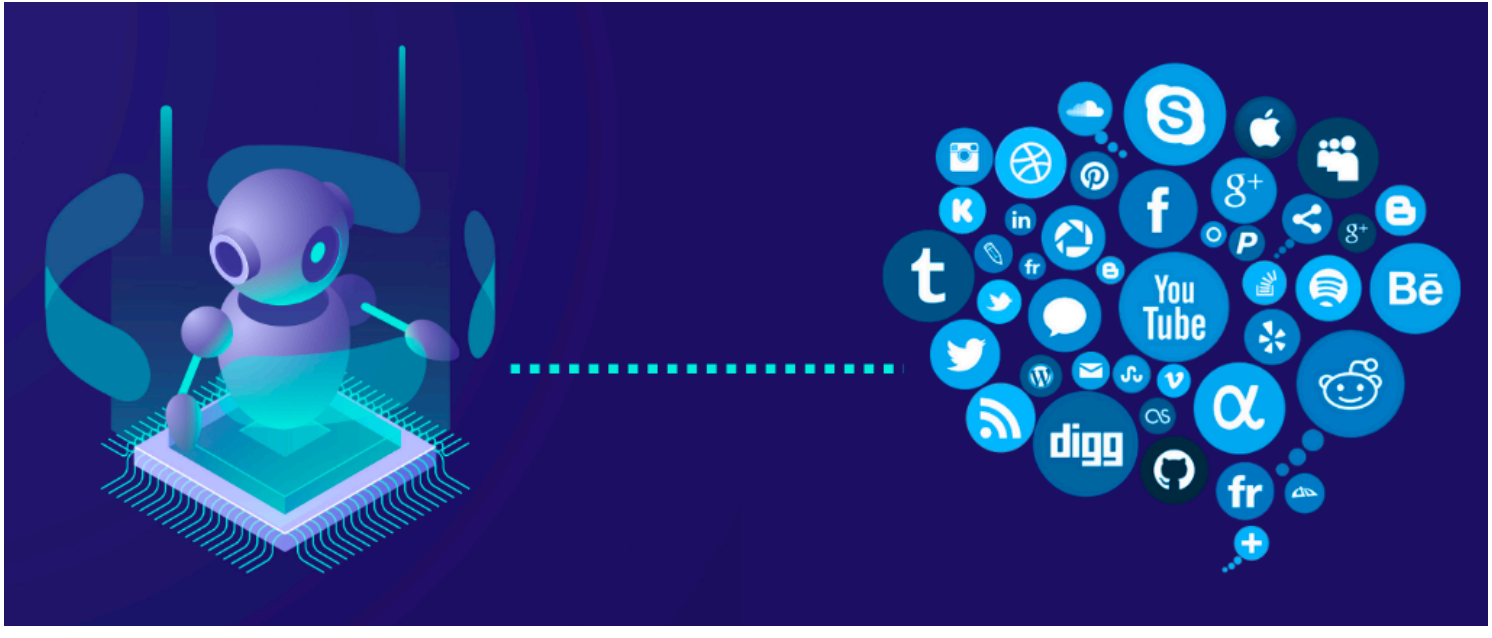
Machine Learning in Robot AIs



Machine Learning in Games



Machine Learning in Marketing



Outline

1. What is Machine Learning??
2. The Wonders of Machine Learning
3. **Data, Models and ML Tasks**
4. Supervised Learning
    1. Regression
    2. Classification
5. Unsupervised Learning
    1. Dimensionality Reduction
    2. Density Estimation

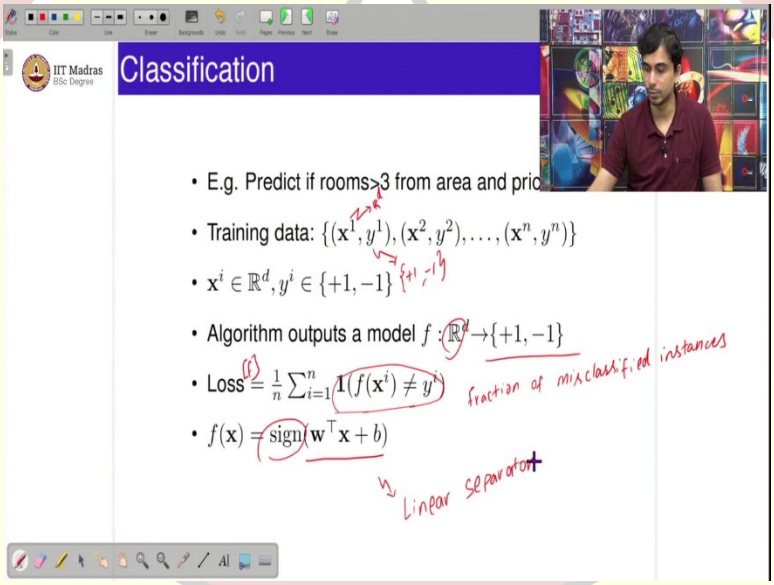
IIT Madras

ONLINE DEGREE

Machine Learning Foundations
Professor. Harish Guruprasad Ramaswamy
Department of Computer Sciences & Engineering,
Indian Institute of Technology, Madras
Supervised Learning - Classification

Hello everyone and welcome to another lecture on machine learning foundations. In the last lecture, we introduced the supervised learning paradigm and the important regression learning problem. In today's lecture, we will take a look at the classification learning problem and some examples.

(Refer Slide Time: 0:36)



A simple classification example, which we had already alluded to, is something like this, where you want to predict if you are given the area and the price of a house and you want to predict if the number of rooms in that house is greater than 3 or less than or equal to 3. So, that is the example of a classification problem, because the prediction that you are making is just two possible values, say that greater than 3 or less than or equal to 3.

The training data for such a thing would once again look similar to the regression case, you would have xi, yi tuples, where xis belong to Rd. But the difference is that yis are no longer real valued, but just, yi would just belong to either + 1 or - 1. That is the main difference in the training data from the regression case where the instances xis are still d dimensional vectors. But the labels are just either + 1 or - 1.

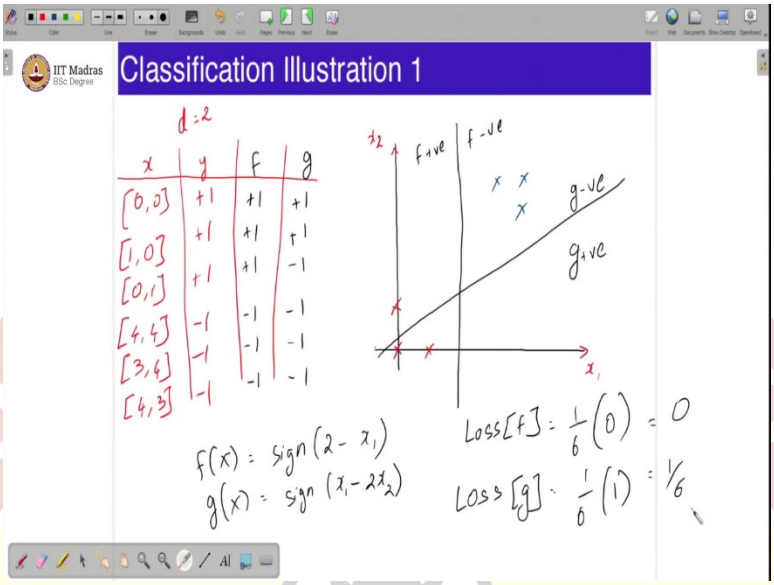
And the algorithm outputs of model f, which in this case is a function from Rd to + 1 or - 1. That is an example classification model. And similar to the case of regression, you want, you need a way to evaluate how good is this model. The ideal case would be that your f of xi would be equal to yi for all i, f of x1 should be equal to y1, f of x2 would be equal to y2 and f of xn would be equal to yn. So, that is the best case scenario. But in general, it might not be.

So, what you have here is the loss of a model f is simply the fraction of misclassified instances. So, that is why you have the indicator, predicators, indicator of f of xi not equal to yi. So, this particular term is going to be equal to 1 if f of xi is not equal to yi. And it is going to be equal to 0 if f of xi is indeed equal to yi. So, f of xi =yi means that you have made the right prediction. The f has made the right prediction and it has no loss. So, you add up all the losses and divide/the number of training points.

So, this essentially corresponds to the fraction of instances that are classified wrongly. So, that is the fraction of misclassified instances. For the regression model, the most common parameterization was you wanted to represent a real valued function and we just did a linear parameterization. That is, we said f of x =w transpose x + b, but for a classification model that is not possible, because you want a model which outputs just either + 1 or - 1.

And the most straightforward way to do that would be just use the same w transpose x + b, but have a sign function on top of that. So, f of x =sign of w transpose x + b and such a model is called a linear separator. This is a, I mean, this is not the only model/any means, but this is one of the most common models and to just, to instantiate an example class of models in your head, so this is the example.

(Refer Slide Time: 3:56)



So, we will have a very simple illustration of classification the same way that we had for regression. And so now let us say we have a two dimensional data set, let us say, d =2. So, let us say this is your data has, let us say, we have 6 data points, x, y, and let us say we have, (())(4:25). Let us say we have 6 data points. Let us say the first data point is say 0 comma 0, the second data point is 1 comma 0, the third data point is say 0 comma 1, the fourth data point is say 4 comma 4, the fifth data point is 3 comma 4, the sixth data point is 4 comma 3.

So, you have 6 data points, each of them being two dimensional vectors. And that is the instances, 6 instances. You I have to give the labels for all of them. Let us say the training data is very simple. So, it is + 1, + 1, + 1, - 1, - 1, - 1. The first three data points are positive, and the next three data points are negative. That is the, let us illustrate this in the 2D case. So, the 2D case, let us see, this is x1 axis, this is the x2 axis.

Note that this is the subscript which indicates the first coordinate and the second coordinate. So, the first data point is 0 comma 0, which would be here, the second data point is 1 comma 0, which would be here, the third data point is 0 comma 1, which would be here. So, these three points are all labeled + 1. So, let us say the red color indicates + 1. Let us use some other color. Let us say use blue to indicate the negative data points.

So, the fourth data point is 4 comma 4 which should be like this and 3 comma 4 and 4 comma 3. So, these three points are labeled - 1 and we will call them as negative data points. So, this together

forms visualization of the data. Now, the classification learning algorithm would essentially take this data and come up with the best model. It is the best model which maps from R2 to either + 1 or - 1. But in this illustration, we are going to do a much simpler task, where we are just going to give two models and ask which of these two models would the learning algorithm pick?

Let us say the models, let us call the models f and g. Let us say, f of x is let us say sign of 2 - x1 and another model g of x is let us say sign of x1 - 2x2. So, these are the two models that we would want the learning algorithm to evaluate. How would we do that? Well, simply, we can just compute the loss of these two models f and g. And to do that, you just need to evaluate the function f on the data points.

Well, if you do that, well, you would get sign of 2 - 0 is + 1, sign of 2 - 1 is also + 1, sign of 2 - 0 is also + 1, sign of 2 - 4 is - 1, sign of 2 - 3 is - 1, sign of 2 - 4 is also - 1. That is your f. And what would your g look like, sign of x1 - 2 x2, sign of 0 is arbitrary.

Typically, you can define a convention where sign of 0 is + 1 and sign of 1 - 2, which is - 1, sign of - 2 which is, sign of 1 - 0 is + 1, sign of 0 - 2 is - 1, sign of 4 - 8 is - 1, sign of 3 - 8 is also - 1, sign of 4 - 6 is also - 1. So this is, if you evaluate g on the six data points, you will get this.

Now, you can compute the loss of f. Loss of f is simply 1/6 × the number of instances that your f got wrong. And you can see clearly see that your f got all the 6 data points right. So, + 1 + 1 + 1 + 1 + 1 + 1 - 1 - 1, and so on. So, this particular loss will actually be 0. What about loss of g? Loss of g is 1/6 × how many of the 6 data points did your g get right, did your g get wrong?

Well, out of all the 6x data points, there is only one data point which it got wrong, which is this. The truth was + 1, but you got - 1 instead. So, that is 1/6× 1 which is 1/6. So, the loss of f is 0, loss of g is 1/6. So, your learning algorithm would prefer f over g. So, in this very simple case, what your learning algorithm could do is just return f, because it is just choosing among the two functions.

In general, however, your learning algorithm would just take this training data and return a function from scratch that is essentially it would return the best function among all possible functions are something similar to that. In this case, once again, it is very easy to visualize f, because a common way to visualize a classifier is as separate regions, because a classifier can take f, a classifier model

can take only two possible values. It is either + 1 or - 1. So, every region of the input is either + 1 or - 1.

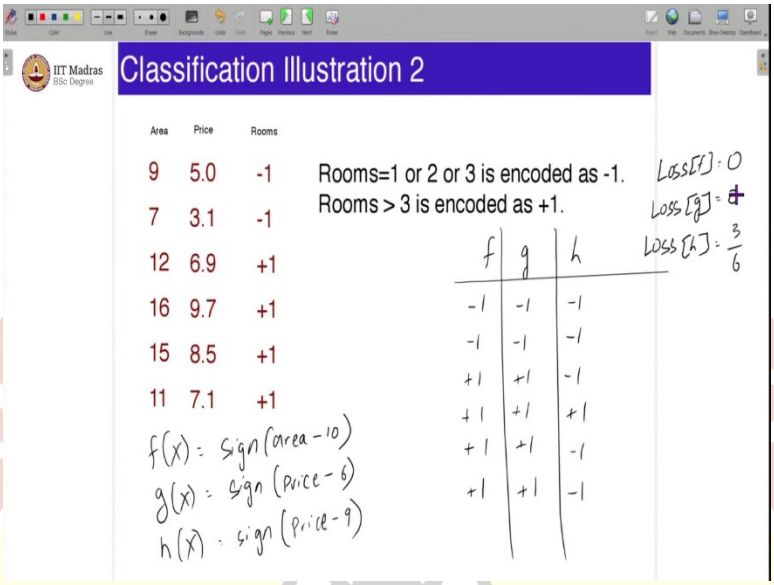
So, if you do that for f, what is the set of inputs which f would classify as positive? So, if some future input comes which, when would it be classified as positive? When it will be classified as positive, if sign of 2 - x1 is greater than 0 which means that these would classify f as positive versus f as negative.

And you can clearly see that based on this, this is the reason why this particular model has 0 loss, because you can see that in the negative region of f you have all the negative data points and in the positive region of f you have all the positive data points. And because of this, this particular model f has 0 loss on the training data.

Let us do the same thing for g. If you do that, here I cannot draw a straight line, something like this. I am purposefully offsetting it/0 because it should ideally pass through 0 and because we have defined sign of 0 to be equal to + 1, I am just moving in a little bit. So, this is the g thing. So, this everything to the right of the bottom right would be, would classified/g as positive and everything above would be classified/g as negative. We can see that this function has a loss of 1, because it classifies this particular data point as negative, even though it is actually positive. So, that is the reason why loss of g is 1/6.

You can argue whether are these the only functions, no, obviously not. There are several possible classifiers. And real classification algorithm would actually find the best classification model out of all such functions. But in this particular simple case, we are just interested in finding out which of the two is a better classifier, f or g and in this case, f is the better classifier.

(Refer Slide Time: 12:07)



So, now let us do one slightly more complicated example. I mean, it is pretty much the same complexity just that it is more real in nature that is corresponds to a data which we can understand. Let us do that. So, let us consider the, let us consider three models in this case. Let us consider the first model f as f of x is simply sign of area - 10. So, this is the same dataset that we saw earlier, where you want to classify the rooms as number of rooms, is it 3 or lesser or is it greater than 3.

So, we are doing an encoding here where we are converting all the rooms with either 1 or 2 or 3, we will call that as - 1, the label for that is - 1 and all rooms where the number of rooms is greater than 3 we will call that as + 1, because we are only interested in predicting whether the number of rooms is greater than 3 or less than or equal to 3. We are not interested in anything else. So, we can just convert the 1 or 2 or 3 to - 1 and 3 as + 1. You could do it the other way around also. It is equivalent.

So, we want to predict whether, from the area on price, whether the number of rooms is greater than 3 or less than or equal to 3. So, one possibility is, well, we can say f of x is sign of area - 10. So, what does this say? This says that if area is greater than 10, then your house has more than 3 rooms. If it is less than 10, then it has 3 or lesser rooms. That is what your model f is saying.

And another example would be, another example model would be let us say g of x =sign of price - 6, another model would be h of x would be sign of price - 9. Let us give f, g and h for the 6 data points, sign of area - 10 which all of these 6, the 6 houses with all houses have area greater than

10. Well, the last four have area greater than 10. So, you can clearly see that sign of f, the f would simply be - 1, - 1, + 1, + 1, + 1, + 1. This is the predictions made/f on the training data.

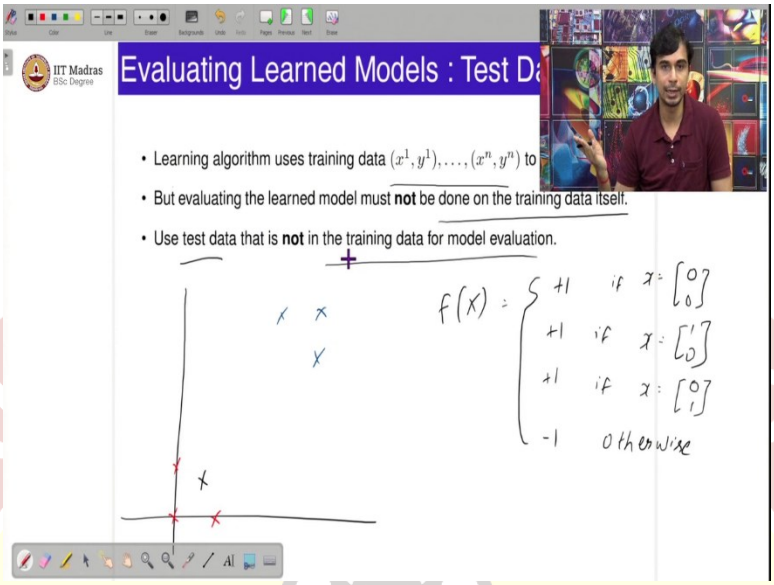
Similarly, what predictions would g make? Sign of price - 6, which all houses have price greater than 6, well, once again, it is the last four houses. So, once again, how sign of price 6 would be - 1, - 1, + 1, + 1, + 1, + 1. Similarly, you can do the same thing for h which is sign of price - 9, which would be - 1, - 1, - 1, + 1, - 1, - 1, only the fourth house has price greater than 9, everything else has price less than 9. So, everything else will be - 1.

Now, we can easily compute what is the loss of f. We can see loss of f =0, because it exactly the true pattern is - 1, - 1, + 1, + 1, + 1, + 1 and it is captured exactly/f. Similarly, loss of g is also equal to 0. But loss of h is not 0. Loss of h is in fact =1, 2, 3, it is 3/6. Out of the 6 data points, it got three of them wrong. So, it is loss of h is 3/6.

So, in this particular case, the learning algorithm is slightly stumped, because you have given two models which, both of which have the same loss which is 0. So, in principle, the learning algorithm can pick either. It can either pick f or it can pick g. As far as the training data is concerned, it is, they are both equally good. In fact, they are equally perfect. They get all the six points correct. It would just not pick h because h is a bad one. Both f and g are good classification models for this particular dataset.

Of course, once again, I have been emphasizing again and again the real, any real classification algorithm would not be choosing among some finite number of choices, but rather it would be choosing the best classification model from an infinite variety, infinite choice. How to do that is something which we will not be addressing right now. So this, with this, we can wrap up the example of classification.

(Refer Slide Time: 17:05)



So, now, we will move to another part which I have slightly glossed over with this. You have these models that say you have picked model f based on the training data or picked model g based on training data. How do you evaluate this model? You, here is the main thing. You are not supposed to use the training data itself to evaluate, the learning model here. Let us say, you have learned, you have decided to use model f based on a training data x1, y1, x2 y2, xn, yn, how would you evaluate this model? I am saying that you should not do that based on the training that itself. We should use test data which is not in the training data for evaluating the model.

Why is that? Well, here is a simple counter example I will give you. Let us take the classification example, first example that we had. So, we had three positive and three negative examples. So, these are the three positive and you had three negative examples. Now, consider this new model I am going to give you which is let us say f of x =+ 1 if x =0, 0, + 1 if x =1, 0, + 1 if x =0, 1 and - 1, otherwise.

So, what does this model exactly do? It essentially, this is what it does. It for almost all of the inputs, it is going to, for almost all of the inputs, it is going to be - 1. Just for the three special cases, it is going to predict + 1. On the training data, this particular model f has 100 percent, it has 0 loss, because it gets all the training points. It has 0 loss. But that is not the main reason why we even do machine learning. The main reason we do machine learning is how to use the training data to get a model which will perform well on future data.

Let us say, for example, you are trying to learn to predict the price of a house from its area and number of rooms and distance to metro and so on, you have a data set. Your goal is to not do well on the already existing data set. Your goal is to learn the principles. And when someone comes to the new house which is not in the data set, you should use the model that you have learned to make a good prediction and you should do well on that house. That is how you should evaluate the model, whether you have learned a good model or not./evaluating the model and the training data itself, you are going to get an overoptimistic estimate of how good the model is.

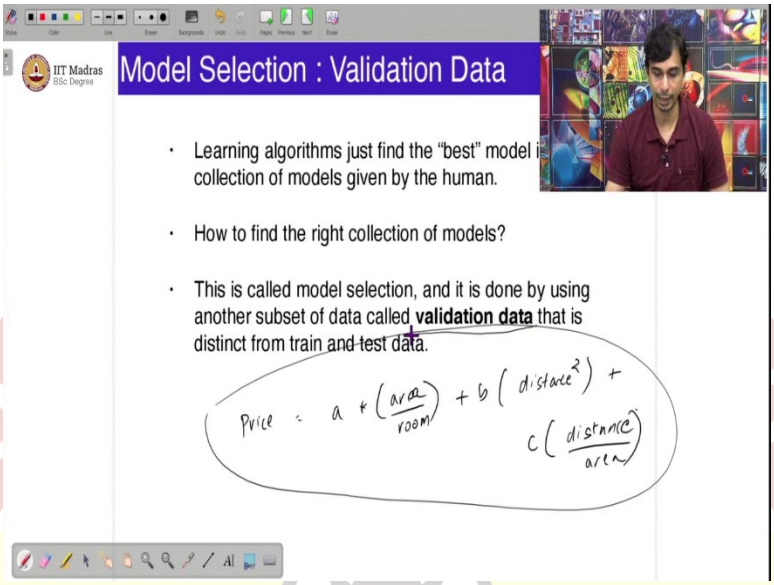
Here is an example. Here this particular example, the model has zero loss on the training set. But on the test, I mean, because this is a very unlikely model, so we can have a new data point. Let us say you have a new data point, let us say like this. We have a new data point. This based on logic you would guess that this new data point is likely positive. But according to your model, it is going to be negative.

According to your learned model, it is going to be negative, because it goes against intuition and because of that your model f, even though it has zero loss on the training set, it is going to perform quite badly on unseen data. So, this is not a good model. But if you used your training data set, you would say it is a good model. That is the reason why you should not use the training data for evaluating the model. You should use some held out data which is a test data, which is called test data that is not in the training data for evaluating the model.

In practice, for example, let us say you are using past hoses, let us say you are using the price of hoses from 1990 to 2020 to learn a new price model, how well you are going to be doing, how well your model does is evaluated based on future houses. So, some new house comes on the market on 2021, if your model does a good job of predicting the price of the house, then it is a good model.

Predicting the price of a house in 1990 accurately is not useful, because you have already seen that. That is the main idea we will have to keep in mind that is learn models or evaluate it on data which is distinct from the training data set, which is called a test data set.

(Refer Slide Time: 22:10)



Very often, you also have one more part of the data that is distinct from the training data and test data, which is called the validation data. And where is used, because this comes from one of the things that I mentioned earlier that is the learning algorithm just finds the best model in the collection of models given/the human. For example, you might say the class of models that you are looking at is linear models. So, let us say a × price, a × rooms + b × area + c × distance, that is your model. Your learning algorithm will find the best a, b and c, but why is that. So, why not do something else?

For example, you might just as well say price is a × let us say area/rooms + b × distance square + c × distance/area. So, you could have something like this also. Why did we not do this and did a × area + b × rooms + c × distance, why did we do that and not this. In fact, there is no reason for us to not do this. This is also perfectly valid you could argue.

So, this particular part of choosing the parameterization is not part of the learning algorithm and that is part of a process called model selection. And how do, this is generally done/humans. So, this is done based on common sense. You decide the price can linearly depend on area, distance and the number of rooms. So, this is something which is set intuitively/humans. But there are several multiple intuitive choices that can be made here.

And typically what is done is you choose among these collections separately and you decide which of these collections the best using a held out data set called the validation data. We will not really

go into great detail on this, but just keep in mind that the finding the right collection of models is done using validation data, finding the best model within the collection is done using train data and evaluating a learned model is done using test data.

With that, we can wrap up supervised learning, and in the next class we can start unsupervised learning.

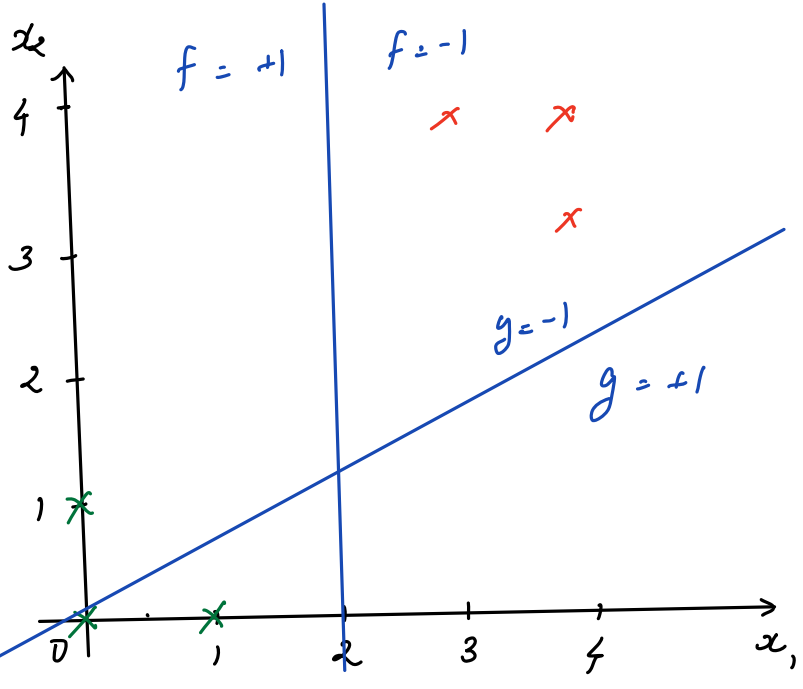
Classification

- E.g. Predict if rooms>3 from area and price.
- Training data: $\{(\mathbf{x}^1, y^1), (\mathbf{x}^2, y^2), \ldots, (\mathbf{x}^n, y^n)\}$
- $\mathbf{x}^i \in \mathbb{R}^d, y^i \in \{+1, -1\}$
- Algorithm outputs a model $f : \mathbb{R}^d \rightarrow \{+1, -1\}$
- Loss[f] $= \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(f(\mathbf{x}^i) \neq y^i)$ = Fraction of training data classified wrongly by f
- $f(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x} + b)$

Linear separator

Classification Illustration 1

x	y	f	g
[0, 0]	+1	+1	+1
[0, 1]	+1	+1	-1
[1, 0]	+1	+1	+1
[4, 4]	-1	-1	-1
[3, 4]	-1	-1	-1
[4, 3]	-1	-1	-1



$f(x) = \text{sign}(2 - x_1)$

$g(x) = \text{sign}(x_1 - 2x_2)$

$\text{Loss}[f] = \frac{1}{6}(0) = 0$

$\text{Loss}[g] = \frac{1}{6}(1) = 1/6$

Classification Illustration 2

Area	Price	Rooms
9	5.0	-1
7	3.1	-1
12	6.9	+1
16	9.7	+1
15	8.5	+1
11	7.1	+1

Rooms=1 or 2 or 3 is encoded as -1.
Rooms > 3 is encoded as +1.

f	g	h
-1	-1	-1
-1	-1	-1
+1	+1	-1
+1	+1	+1
+1	+1	-1
+1	+1	-1

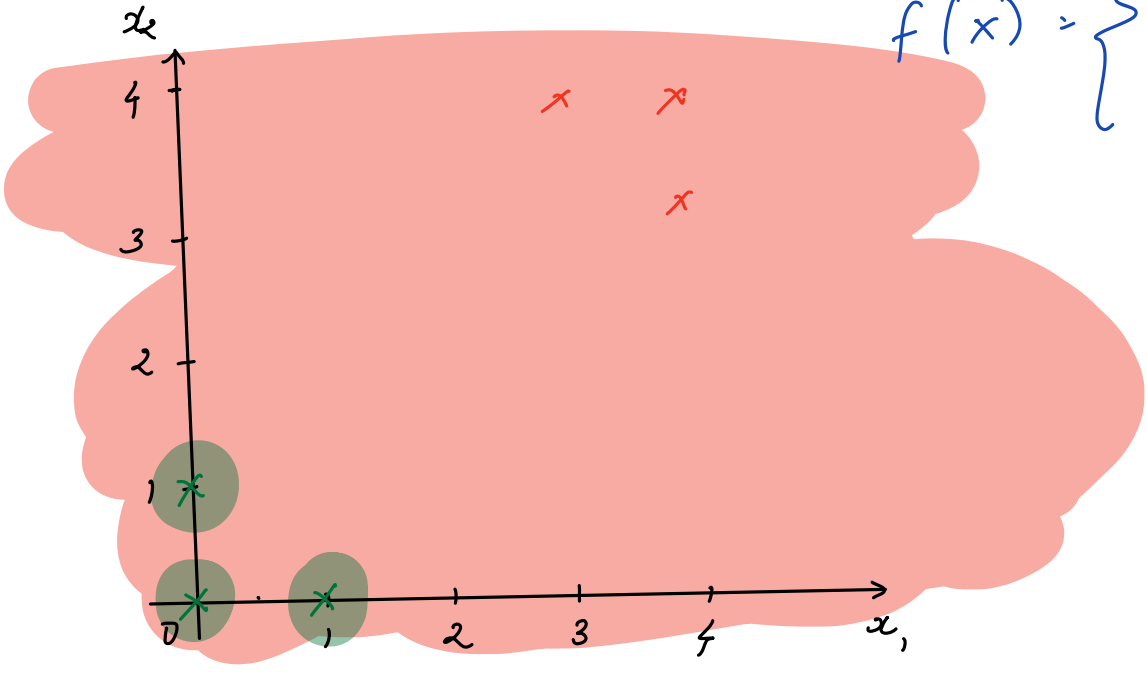
$f(x)$: sign (area - 10)

$g(x)$: sign (Price - 6)

$h(x)$: sign (Price - 9)

Evaluating Learned Models : Test Data

- Learning algorithm uses training data $(x^1, y^1), \ldots, (x^n, y^n)$ to get model f.
- But evaluating the learned model must **not** be done on the training data itself.
- Use test data that is **not** in the training data for model evaluation.



$$f(x) = \begin{cases} +1 & \text{if } x = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\ +1 & \text{if } x = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ +1 & \text{if } x = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ -1 & \text{otherwise} \end{cases}$$

Model Selection : Validation Data

- Learning algorithms just find the “best” model in the collection of models given by the human.
- How to find the right collection of models?
- This is called model selection, and it is done by using another subset of data called **validation data** that is distinct from train and test data.

$$Price = w_1 * (\#rooms) + w_2 (area) + w_3 (distance) + b$$

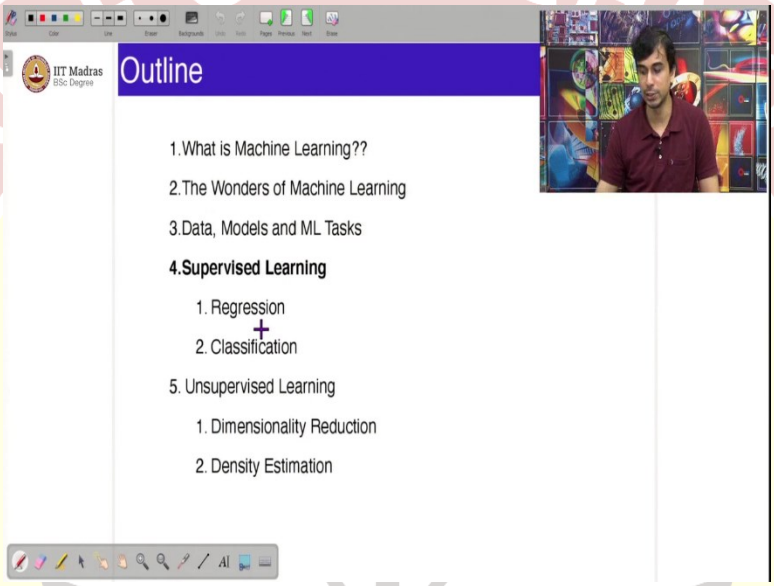
IIT Madras

ONLINE DEGREE

Machine Learning Foundations
Professor Harish Guruprasad Ramaswamy
Department of Computer Sciences & Engineering
Indian Institute of Technology, Madras
Supervised Learning - Regression

Hello, everyone, and welcome to another lecture on Machine Learning Foundations.

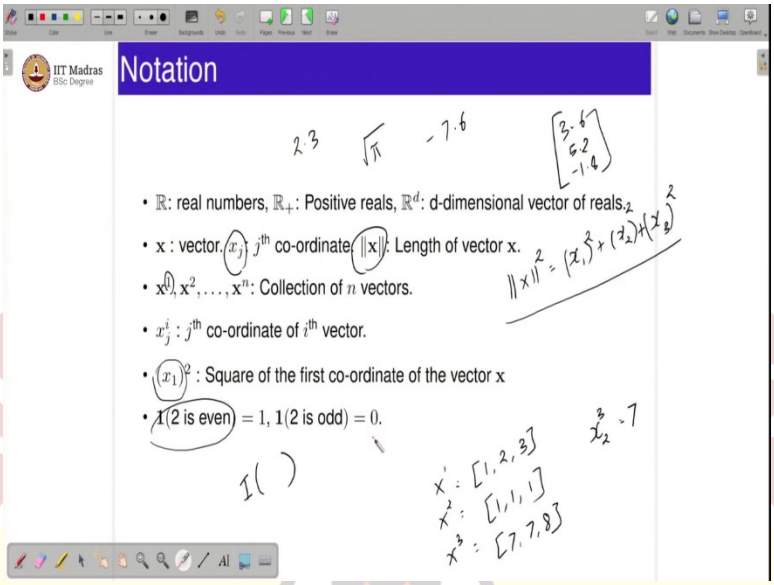
(Refer Slide Time: 00:26)



In this lecture, we will take a deeper dive into one main paradigm of machine learning, which is supervised learning. There are several supervised learning tasks. But the most, two most important tasks that are encountered in machine learning are regression and classification. And in fact, supervised learning is the most common paradigm of machine learning.

And when someone says, machine learning, they typically mean supervised learning unless they qualify it with extra things like unsupervised learning or reinforcement learning and so on. In this lecture, we will take a deeper dive into regression and classification, which are types of supervised learning problems.

(Refer Slide Time: 01:03)



So, before we do that, we will set up some notation, which we will need to handle the rest of the lecture. We will use $\mathbb{R}$ to denote the set of real numbers. Pretty much anything that you can, any scalar is a real number. So, for example, you have 2.3 is a real number, $\sqrt{\pi}$ is a real number, - 7.6 is a real number. All of these are examples of real numbers. The positive reals are simply real numbers that are positive.

And another set that you will encounter often is $\mathbb{R}^d$, which is the set of d dimensional vector of reals. For example, an element in $\mathbb{R}^3$ might be something like $\begin{bmatrix} 3.6 \\ 5.2 \\ -1.8 \end{bmatrix}$ is an example of an element in $\mathbb{R}^3$. We will often denote a vector by an element like x and x subscript j will denote the jth coordinate of the vector and norm of the vector or the length of the vector is denoted by this particular symbol, which the length will always mean the euclidean length.

For example, if we have norm of a three dimensional vector x is given by, the norm square is easier to say, the norm square is simply $x_1^2 + x_2^2 + x_3^2$, norm of this, norm or the length of this vector x is simply the $||x|| = \sqrt{x_1^2 + x_2^2 + x_3^2}$.

In a lot of situations, we will be working with a collection of vectors and we will denote different elements of the collection by a superscript. So, note that we are using superscript for denoting different vectors and subscripts for denoting elements in the vector. So, we can put both of these

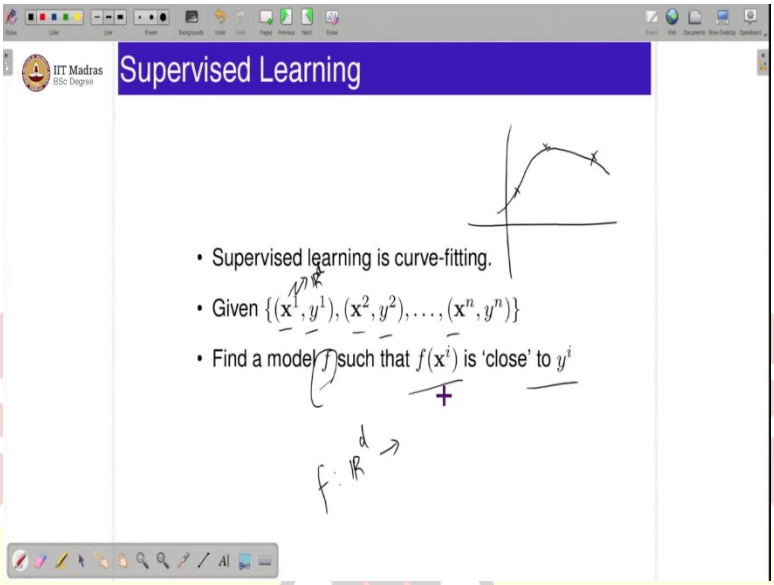
together to have, for example, something like x_j^i. x_j^i corresponds to the jth coordinate of the ith vector. So, for example, let us say you have a collection of three vectors in let us say 3D space. So, let us say you have $x^1 = [1,2,3], x^2 = [1,1,1], x^3 = [7,7,8]$.

You can have x_2^3 is the second element of the third vector which would be 7. Note that this is different from the $(x_2)^3$. We will denote that differently by, we will make a distinction between that by trying to use parenthesis around x1. So, $(x_1)^2$ corresponds to the square of the first coordinate of the vector x. For example, here, you can see that to denote norm x square, I purposefully created $(x_1)^2$. I might forget to do that some×, but from the context, it should be clear whether I am talking about the square of the x1 or the second vector.

From context it should be clear whether we are talking about superscripts or powers. This confusion is inevitable, because we just do not have enough space for denoting everything that we will have. But it should be clear from context whether we are talking about superscripts or powers. And one other notation that we will be using often is indicator variables, which is denoted by 1 of something. So, you can have a predicate class which is either true or false and based on that the output of this is going to be either 1 or 0. For example, you can have 2 is even.

2 is even is a class which is known to be true. So, $1(2\ is\ even) = 1$, but indicator of 2 is odd is known to be false. So, $1(\ 2\ is\ odd) = 0$. So, this is another notation which will be very useful for converting from math, from English to math in a lot of places. So, this is the most basic condition that you will need for handling the rest of the lecture. We will use other notation which we will introduce as and when required.

(Refer Slide Time: 05:55)

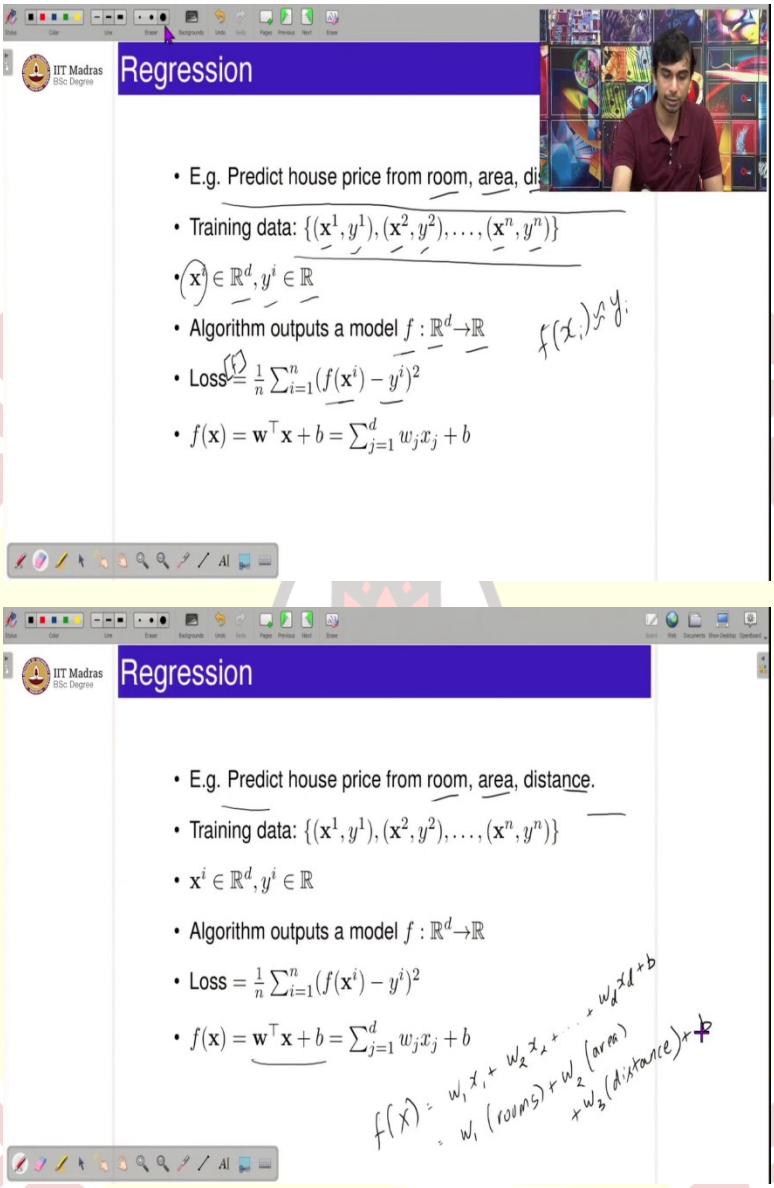


So, now, we enter the supervised learning paradigm. What is supervised learning? You can simplify, you can extreme, there is an extreme simplification that you can do and say that supervised learning is simply curve-fitting. Curve-fitting is something which you all would have done in your school. For example, you might have done something like this. You might have points like this, and you might fit a curve through these points. This is an example of a curve that fit, you fit a curve through these points.

Supervised learning at its core, at its essential is just curve-fitting. There are several aspects to this which make the problem much more complicated. But at its core, it is just curve-fitting. Which means that you have, what you have is a bunch of points, which are $\{(x^1, y^1), (x^2, y^2), (x^3, y^3), \ldots (x^n, y^n)\}$. Each of these x^i are a vector, let us say in $\mathbb{R}^d$. It is a d dimensional vector. And y^is are in some space. Typically, they will be either reals or + 1 or - 1 or some small set. We will come to that later.

The goal is to find a model. A model in this case simply is a function f from $\mathbb{R}^d$ to the space of labels. So, each data point x^i, y^i has an instance x^i and a label y^i. Your goal is to find a model f such that f of xi is close to y^i. And this is what a function really represents the curve here and you want the code to be as close as possible to the points. And that is represented by saying that you want to find the model f such that $f(x^i)$ is close to y^i. This is being intentionally vague because you will see this instantiated for regression and classification soon.

(Refer Slide Time: 07:50)



For the case of, what is an example regression problem, we saw that we want to predict let us say house price from room area and distance. So, that is an example of a place where a regression model is used. And how would you build such a model? Well, you will need training data, you will need a data set, which will consists of house one and the price of the house one, house two and price of house two, house three and price of house three.

When we say house one, we mean simply a three dimensional vector giving its number of rooms, its area and the distance to metro. So, each house is represented by a three dimensional vector and

you have a label, which corresponds to the price of the house. So, this would be an example of training data.

You have your instances x^i in the general setting, it would be in a d dimensional space and your labels y^i would be real valued. Your learning algorithm would essentially output a model f, which is essentially a function from $\mathbb{R}^d$ to $\mathbb{R}$. And how would you evaluate this model. So, ideally, you want your model to be such that $f(x^i)$to be approximately equal to y^i. So, if $f(x^i) = y^i$ for all i, then that is the best model that you can think of right now at least.

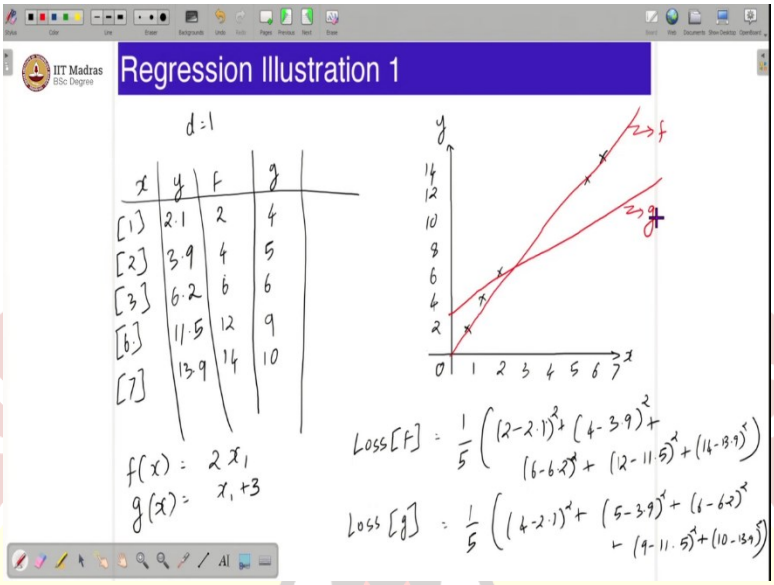
So, but that might not always be possible. You might be forced to choose a model which is not perfect. And a typical loss that is used for evaluating this model, evaluating a given model f, let us call that loss of f is simply measures the deviation of f of x^i from y^i by a simple square thing. So, if $(f(x^i) - y^i)$the whole square essentially captures how far is f of xi from y^i. You do not want your f of x^i to be too much above y^i or it should not be too much below y^i also. The ideal situation is f of xi is exactly equal to y^i, but you are willing to tolerate a little bit of error.

And $f(x^i) - y^i$ the whole square essentially ensures that it is always positive. This loss can never be negative. And immediately, you can see that when loss of f is equal to 0, then f of xi is equal to yi for all i. That is the best, that is the smallest loss possible. Typically, what you were, you like your learning algorithm to do is to find a model f which has low loss. So, that is the goal of a learning algorithm.

I have been saying f as a function from Rd to R, but how do you represent such a function? The most common representation for a function for such a function is a linear parameterization. That is, you are going to say my function f is given by $w^T x + b$. What do I mean by that? Well, f of x is going to be equal to $w^1 \times x^1 + w^2 \times x^2 + \ldots, w^d \times x^d + b$. In this case, your w^1, w^2. These are your parameters. w^1, w^2, w^d and b are your parameters and these, the dependence of the price or the output model is linear in terms of its input, actually, it is called a linear model.

For example, in the case of the house prediction price, it might say something like this. It is $w^1 \times$number of rooms + $w^2 \times$area + $w^3 \times$the distance + b. This is an example of a linear model. The w^1, w^2, w^3 and b form the parameters. This represents a linear function. So, every possible value of w^1, w^2, w^3 and b corresponds to a model. What your learning algorithm does is essentially find the best possible w^1, w^2, w^3, b for this particular training data.

(Refer Slide Time: 12:27)



So, here is a simple illustration of how that might be done. So, let us have a very simple data where your input dimension d is simply just one dimension, let us say d is equal to 1. Let us have a very simple data here, let us say x, y. So, your x is just a one dimensional vector. So, the instances are one dimensional vector.

So, effectively they are scalars. But I am going to write them as a vector format so that you understand that x can be a vector. Let us say the first data point x is x1 is, let us say just 1. The second data point, let us say it is equal to 2, let us say the third data point is 3, the fourth data point is say 6. The fifth data point is 7.

Now, five data points, x_1, x_2, x_3, x_4, x_5, they are all d dimensional vectors, but d happens to be equal to 1. So, this is a very simple case. And your y let us say it is 2.1 for y_1 is let us say 2.1, say 3.9, y_3 is let us say 6.2, y_4 is let us say 11.5 and let us say y_5 is 13.9. In this particular case, you can actually easily visualize the data because x happens to be just one dimensional. So, you can just plot the data and see what we mean by that. So, x which is 0, 1, 2, 3, 4, 5, 6, 7, let us say 2, 4, 6, 8, 10, 12, 14.

So, you have the first data point is 1,2.1 that approximately would be here, 2,3.9 would be approximately here, 3,6.2 would be approximately here, 6,11.5 would approximately be here, and 7,13.9 would approximately be here. So, now you have your, this is your x-axis, this is your y-

axis. Because the data happens to be just d equal to one dimensional, you can do the simple plotting to visualize the data.

What would a learning algorithm do? Learning algorithm would take in this data and output a model f which has small loss. We will not actually do the full business here. But we will just try to evaluate two distinct models. Let us have two distinct models. Let us evaluate the model given by f. Let us say f of x is let us say it is just $2x_1$. So, even though x is just a scalar, I am just going to write $2x_1$ here for completeness. And let us say another model, g of x is, let us say, $x_1 + 3$.

So, now, in general, the learning algorithm would have to find the best model. I am going to make the job easier for the learning algorithm and say that you just choose the best out of these two models. How would the learning algorithm do that? It is a very simple thing where it can just compute f and compute g, and decide which of these models have a, has a smaller loss. So, for example, if you compute f, it is $2x_1$. So, this is 2, this is 4, this is 6, this is 12 and this is 14. g would be x1 + 3 it would be 4, 5, 6, 9 and 10.

What is the loss of f? Well, that is simply equal to $\frac{1}{5} \times (2 - 2.1)^2 + (4 - 3.9)^2 + (6 - 6.2)^2 + (12 - 11.5)^2 + (14 - 13.9)^2$. So, this is the loss of the model f. You can compute this loss. And similarly, you can compute the loss of g also.

Or would that be it is the average of these terms which is $(4 - 2.1)^2 + (5 - 3.9)^2 + (6 - 6.2)^2 + (9 - 11.5)^2 + (10 - 13.9)^2$. You can compute the loss of the models f and g. And clearly, you can see that the loss of f is going to be much lesser than the loss of g. And your learning algorithm would prefer model f over model g.

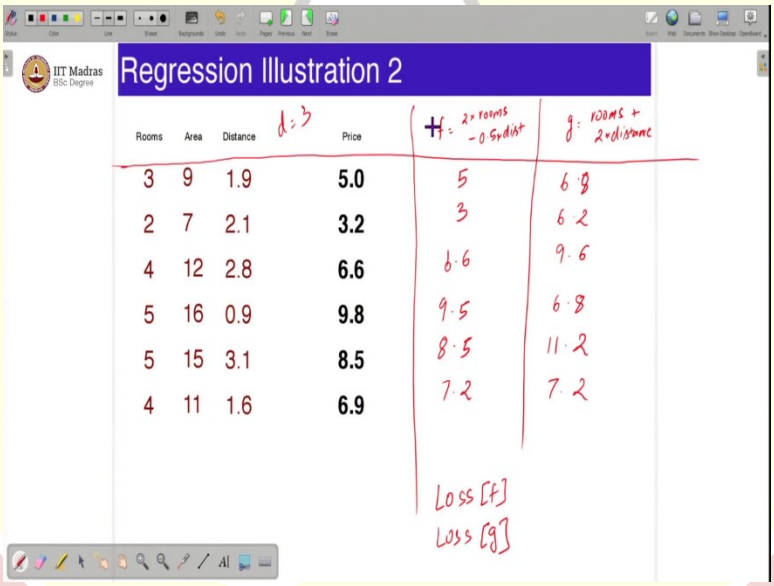
Is that the best model? We do not know yet. I mean, the learning algorithm in the real world, in practice, would not just have to choose between two models f and g. I have done here as just an illustration to show you that f, I mean, in principle, the learning algorithm just chooses the best model from an entire library instead of just the best out of two models.

And so, in this particular case, we can actually do a better job and actually illustrate the function f. And what is the function f here, the function f is simply $2x_1$ and that you can actually plot that function here. And if you do that, that is an example. This is f. It is not particularly straight, but I cannot do a much better job than this. So, this is f.

And g would essentially be $x_1 + 3$, which would be something like this. This is, you can clearly see that f is, f fits the data points much better than g. So, f is a much better curve that fits the data points than g and that is the reason why you would have loss of f is less than loss of g and the learning algorithm would prefer f over g. So, this is a very simple illustration of regression.

The learning algorithm, of course, we have, I have not given you the learning algorithm. The learning algorithm in principle does this job of finding the best model over an infinite set. So, here we have just done the, make the, made a simplified learning algorithm where it just chooses the best among two models. But in the real world, it has to choose the best among an entire library of models.

(Refer Slide Time: 19:57)



Here is another simple illustration that to make things more concrete for the case of price illustration. So, you can have, let us say, two models. The first model f can be something like let us say, 2 × rooms - 0.5×distance. That is one possible model. You can have another model, let us say, g, which is rooms + 2×distance. This is another simple example here. Now, d is equal to 3. So, you have your data lives in three dimensional space, and of course, the labels are always going to be real. They are going to be scalar. So, that is your 5, 3.2, 6.6, and so on.

And now, let us say the learning algorithm, once again, we are making the job of the learning algorithm easy. And we are just asking the learning algorithm to choose the better among these

two models, f and g. How would we do that? Well, once again, you can repeat the same thing that is you can compute the f and g for the training points and see which has the smaller loss.

For example, if you do that for f 2×rooms - 0.5×distance, that would be 6 - about approximately, I am not going to write the exact numbers here, because it does not matter. I am just going to write an approximate number here, that is approximately 5. 2×rooms - 0.5×distance it is 2×2, 4 - 0.5×2.1 that is approximately 3.

Similarly, you have 2×rooms, which is 2×4 - 0.5×2.8, 8 - 1.4 that is approximately 6.6. Similarly, you can do that for the other three data points also. So, 10 - 0.45 that is approximately 9.5, that is 10 - 1.5 that is approximately 8.5, 8 - 0.8 that is approximately, let us say 7.2. So, this is the predictions made by model f.

And you can do the same thing for model g that would be, let us say, rooms + 2×distance, that is 3 + 3.8, let us say, 6.8, 2 + 4.2 that is 6.2, 4 + 5.6, 9.6, 5 + 1.8 that is 6.8, 5 + 6.2 that is 11.2, 4 + 3.2 that is 7.2. Now, this column gives the predictions of model f and this column gives the predictions of model g. You can use this to compute loss of f and loss of g.

How would you do that? Well, that is the same thing that you average the square difference between 5 and 5.0, 3 and 3.2, 6.6 and 6.6, 9.5 and 9.8, 8.5 and 8.5, 6.9 and 7.2, and so on, that would give you the loss of f. And similarly, you can do the same thing for loss of g. And you can clearly once again see that loss of f is going to be much smaller than loss of g. So, f is a much better model than g. And once again, you can see that it is saying a very meaningful thing. It is f is saying that 2×rooms - 0.5×distance is a good estimate of the price, which is a reasonable, once you see the answer it looks reasonable.

That is, well, it seems to depend on the number of rooms, it increases with the number of rooms, but it does decrease with the distance to the metro. That is what this model is saying. And it does seem to work well on this data. If the learning algorithm has just given these two models to choose from, it will choose f. But once again the learning algorithm is not given just these two models, it is given an entire library of models and it will choose the best out of this library of models. So, with that, I think we have wrapped up regression. We will move on to classification in our next video.

Outline

1. What is Machine Learning??
2. The Wonders of Machine Learning
3. Data, Models and ML Tasks
4. **Supervised Learning**
    1. Regression
    2. Classification
5. Unsupervised Learning
    1. Dimensionality Reduction
    2. Density Estimation

Notation

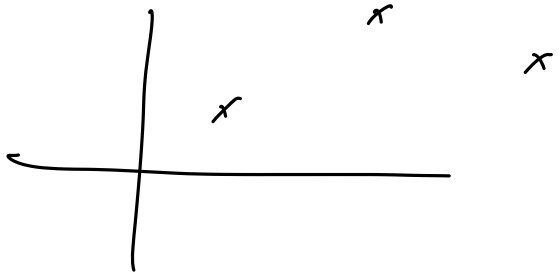
$x^1 = [1, 2, 3]$
$x^2 = [7, 8, 9]$
$x_2^2 = 8$

$\begin{pmatrix} 1.3 \\ -7.6 \\ 5.9 \end{pmatrix} \in \mathbb{R}^3$

- $\mathbb{R}$: real numbers, $\mathbb{R}_+$: Positive reals, $\mathbb{R}^d$: d-dimensional vector of reals.
- $\mathbf{x}$: vector. x_j: j^{th} co-ordinate. $\|\mathbf{x}\|$: Length of vector $\mathbf{x}$.
- $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n$: Collection of n vectors.
- x_j^i : j^{th} co-ordinate of i^{th} vector.
- $(x_1)^2$: Square of the first co-ordinate of the vector $\mathbf{x}$
- $\mathbf{1}(2 \text{ is even}) = 1$, $\mathbf{1}(2 \text{ is odd}) = 0$.

$x = \begin{bmatrix} 1 \\ 2 \\ 7 \end{bmatrix}$
$x_3 = 7$
$\|x\|^2 = x_1^2 + x_2^2 + \ldots + x_d^2$

Supervised Learning



- Supervised learning is curve-fitting.
- Given $\{(\mathbf{x}^1, y^1), (\mathbf{x}^2, y^2), \ldots, (\mathbf{x}^n, y^n)\}$ $\mathbf{x}^i \in \mathbb{R}^d$
- Find a model f such that $f(\mathbf{x}^i)$ is 'close' to y^i

f is a function

Regression

- E.g. Predict house price from room, area, distance.
- Training data: $\{(\mathbf{x}^1, y^1), (\mathbf{x}^2, y^2), \ldots, (\mathbf{x}^n, y^n)\}$ $\mathbf{x} \in \mathbb{R}^3$, $y \in \mathbb{R}$
- $\mathbf{x}^i \in \mathbb{R}^d, y^i \in \mathbb{R}$
- Algorithm outputs a model $f : \mathbb{R}^d \to \mathbb{R}$
- Loss[f] $= \frac{1}{n} \sum_{i=1}^{n} (f(\mathbf{x}^i) - y^i)^2$ = Squared loss
- $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b = \sum_{j=1}^{d} w_j x_j + b$

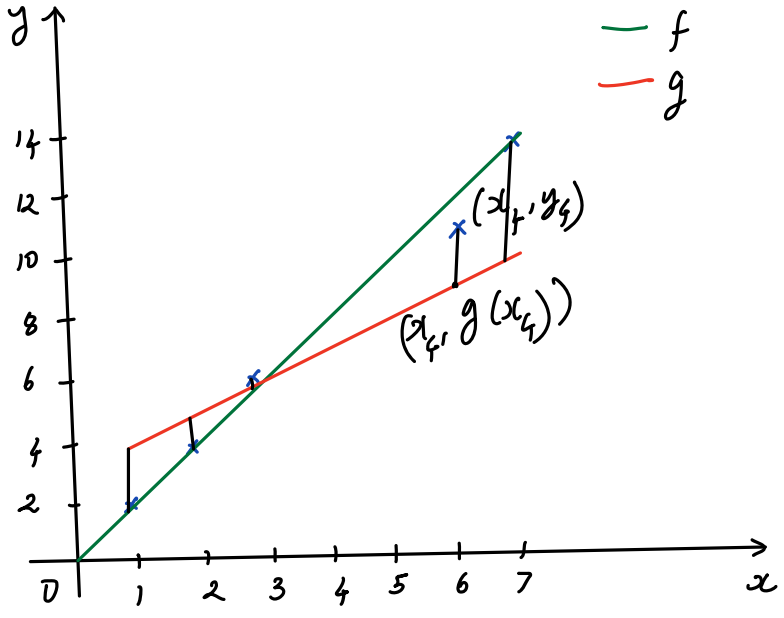
Linear Parameterisation

$= w_1$ (# rooms) + w_2 (area) + w_3 (distance) + b

Regression Illustration 1

$d = 1$

x	y	f	g
[1]	2.1	2	4
[2]	3.9	4	5
[3]	6.2	6	6
[6]	11.5	12	9
[7]	13.9	14	10



$f(x) = 2x_1$

$g(x) = x_1 + 3$

$$\text{Loss}[f] = \frac{1}{5}\left((2-2.1)^2 + (4-3.9)^2 + (6-6.2)^2 + (12-11.5)^2 + (14-13.9)^2\right)$$

$$= \frac{1}{5}(0.3)$$

$$\text{Loss}[g] = \frac{1}{5}\left((4-2.1)^2 + (5-3.9)^2 + (6-6.2)^2 + (9-11.5)^2 + (10-13.9)^2\right)$$

Regression Illustration 2

Rooms	Area	Distance	Price	f = 2 * Rooms − 0.5*dist	g = Rooms + 2*distan
3	9	1.9	**5.0**	5.05	6.8
2	7	2.1	**3.2**	2.95	6.2
4	12	2.8	**6.6**	6.6	9.6
5	16	0.9	**9.8**	9.5	6.8
5	15	3.1	**8.5**	8.5	11.2
4	11	1.6	**6.9**	7.2	7.2

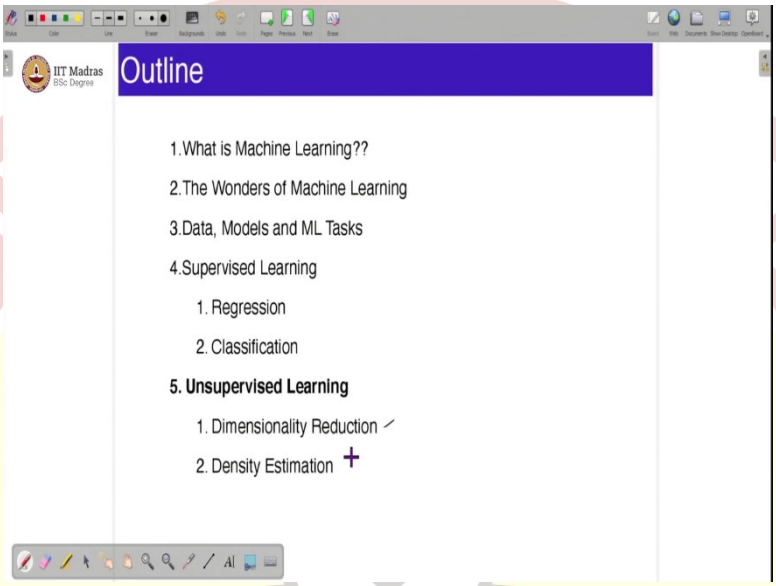
IIT Madras

ONLINE DEGREE

Machine Learning Foundations
Professor Harish Guruprasad Ramaswamy
Department of Computer Science and Engineering
Indian Institute of Technology, Madras
Unsupervised Learning Dimensionality Reduction

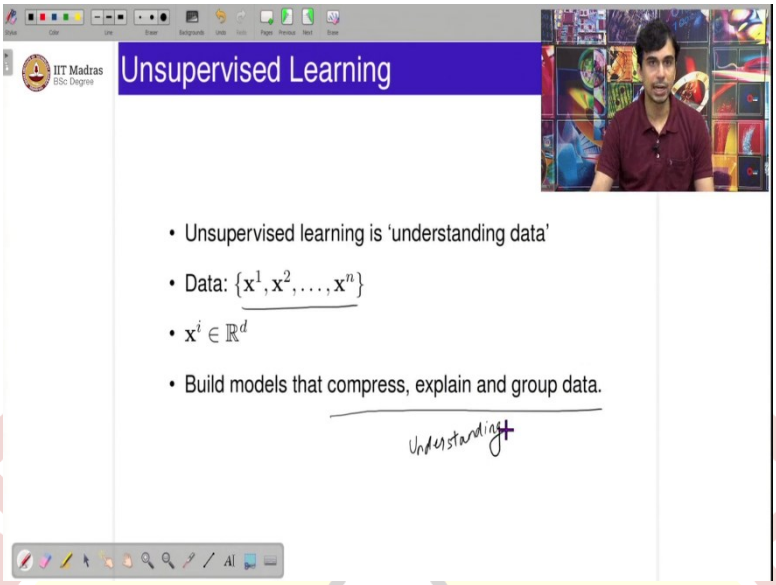
Hello, everyone, and welcome to another lecture on machine learning foundations.

(Refer Slide Time: 0:23)



In the previous lectures, we introduced the supervised learning paradigm and went into detail on the two main problems, two main types of supervised learning tasks which are regression and classification. In this lecture, we are going to start on unsupervised, on the unsupervised learning paradigm, and the two main tasks associated with it, which are dimensionality reduction and density estimation.

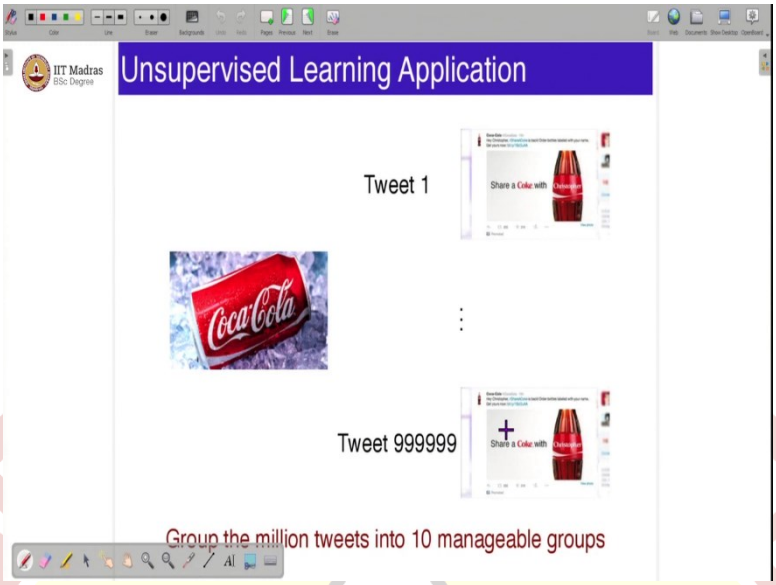
(Refer Slide Time: 0:47)



In contrast to supervised learning, which had very clear and marked goals and ways of quantifying them, unsupervised learning is typically much more vague. And unsupervised learning typically is used as a pre-processing step, and not as an end in itself. Vaguely, unsupervised learning can be viewed as base for understanding data. And data in our context here is simply going to be a collection of vectors.

Note that in contrast to supervised learning, which always had two pairs of xi yi, here you have just xi, which are just a collection of d dimensional vectors. The goal of unsupervised learning is to build models that compress, explain and group data, which is what I am broadly grouping as understanding. We will explain what all of these mean with two specific examples, which are dimensionality reduction, and density estimation.

(Refer Slide Time: 1:53)



Here is an example of how unsupervised learning can be useful. Unsupervised learning, as I mentioned, is typically not used as an end in itself, because the outputs of unsupervised learning algorithms by themselves are not useful. But after human interpretation, and after other machine learning tasks, they can become very valuable.

For example, let us say you are a marketing manager at Coca Cola. And your job is to collect the tweets about Coca Cola and summarize them to your boss. So, that is your job. And let us say in any given week, there are let us say 1 million, 10 million say, a million tweets about Coca Cola that happen in a given week. There is no way you can show all the million tweets to your boss and explain what each tweet is, it is just not possible.

A reasonable thing to do would be if you can group these, let us say a million tweets into 10 distinct groups. So, maybe you can have one group of people who are all just taking selfies with coke in a new place. Or maybe there is another group of tweets, which are from other brands, which has, which are doing some co-branding or we can have another group of feeds, which are all about people who are promoting Coke, and paid by Coke.

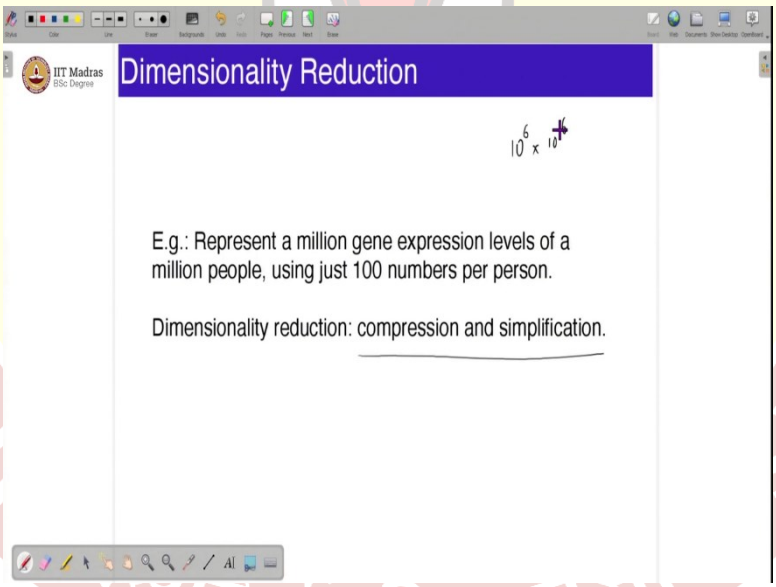
So, there are several groups which can potentially think of, and if you can, even without actually giving this information, if you could potentially group these million tweets into 10, manageable groups, and understand what these groups are by going through these groups. And then you can go to your manager and say, well, there are 10 types of tweets that happened this week.

And the type 1 tweets are called tweets by people who are buying coke for the first time and tweeting about it from a store near their house. And group 2 is all people who are all businesses which are collaborating with coke and so on. And you can easily summarize such a situation to your boss and the boss, you can do that in a reasonable amount of time, and you can get your job done.

So, this is an example of unsupervised and you can note that the what the unsupervised learning does is simply group that tweets into 10 groups. So, that is the, that is what it does anything beyond that interpreting these groups is typically the job of the human being, because groups by themselves are meaningless. Only when you assign coherent meaning to such a group, would, it, would it be actionable or useful.

So, that is done by the human which is in this case would be you. That is the reason why is it unsupervised learning is typically not an end in itself, but rather a pre-processing stage which is used by other processes.

(Refer Slide Time: 4:53)



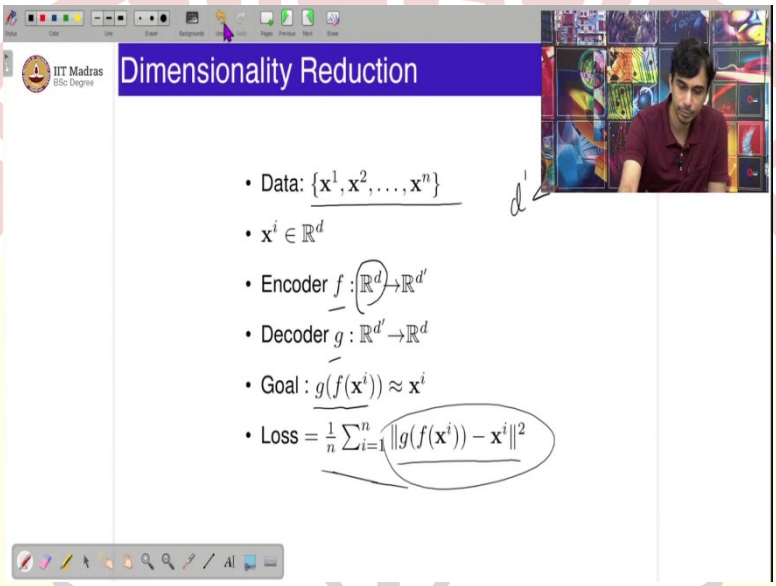
Here is one example of one concrete example of unsupervised learning which is dimensional reduction. So, the goal of dimensionality reduction is compression and simplification. Where would it be useful for? For example, let us say you have your genetic company and you want to export or comprehensively store the million gene expression levels of a million people.

So, you have you have gene results of a million people. And each, each person has, let us say, a million genes. So, what have is you have to we have computed the gene expression levels of these million genes, a million people. So, in principle, it is a $10^6 \times 10^6$ matrix. It will have

million people each having million genes. So, if you have the entire data, that is, that is a 10 power 6 + 10 power 6 numbers, and there is no way you can, let us say, transmit this data from one lab to another lab, it is just not possible.

What would be nice would be if you can compress this data into a simpler format, which can be used for transmitting. So, that is one reasonable goal to have. And dimensionality reduction is one of the main tools that you can use for such a task for such a goal.

(Refer Slide Time: 6:17)



Formally writing mathematic, mathematical terms, that is you might have data in D dimensional vectors x1 to xn. And the goal of dimensionality reduction is to come up with two models, unlike all the other previous cases like classification or regression, where the goal of the dimension, the learning algorithm was to come up with a single model, the goal of a dimensionality reduction algorithm is to come up with two models, an encoder and a decoder.

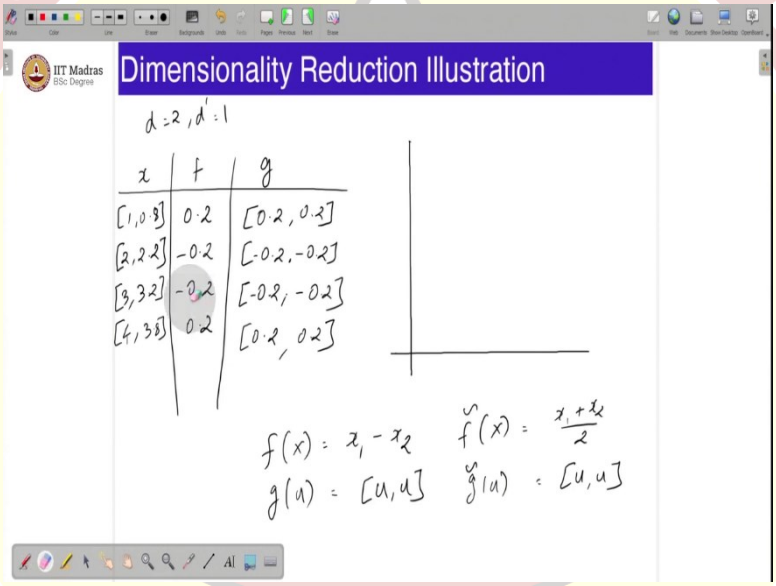
What does the encoder do? The encoder is a function which takes in a D dimensional vector and outputs a d' dimensional vector. Typically, the' is less than D, typically d' is much lesser than d. Effectively, the encoder compresses a D dimensional vector into a d' dimensional vector. And the decoder essentially hopes to undo the effect of the encoder, the decoder takes a d' dimensional vector and outputs a D dimensional vector.

What is the goal of the encoder and the decoder so that if you take any input in your data, and you encode it and you decode it again, you should get back the original data. The goal is that your g of f of xi should be equal to xi, but you are willing to relax it and say that, as long as it

is approximately equal to xi, you are happy. So, that is the goal now. The goal is g of f of xi is approximately equal to xi. And how do you measure approximation?

Well, a reasonable way of measuring approximation would be to simply measure take g of f of xi - xi and view that as a vector and compute the norm square or the length squared of that vector. I, if this was equal to 0, then the g of f, norm of g of f of xi - xi the whole square if that was 0, clearly, g f of xi =xi, but in general it need not be equal to 0, you are looking for a pair of encoder and decoder so that g of f of xi - xi, the norm squared of that is as small as possible. And once again, you are averaging this over all n possible input, all the n inputs as 1 by n, sigma i equal to 1 to n, norm of g of f of xi - xi the whole square.

(Refer Slide Time: 8:33)



Here, is a very simple illustration to show how the dimensionality reduction can work. So, let us take a very simple example here. Let us say you have d =2, and d' =1 in this case, it is a very simple example. You have 4, let us say how 4 training points, this is the 4 training points, this is your x, the 4 training points are let us say, 1 , 0.8. The second training point is let us say 2 , 2.2. Third is 3 , 3.2. And the fourth is 4 , 3.8.

This is your 2 dimensional data and you are looking for an encoder, which takes in a 2 dimensional vector and outputs a scalar. And a decoder which takes in a scalar or a one-dimensional vector and outputs a two-dimensional vector. We will consider 2 possibilities for this. So, let us take one, let us say f of x =one possibility is x1 - x2 and g of u is u , u.

Note that f is a mapping from R2 to R, which means that x is a 2 dimensional vector but x1 - x2 is clearly a scalar, x1 - x2 is clearly a scalar. f is a mapping which takes in a 2 dimensional

vector and outputs a scalar and g is a mapping which takes a decoder which takes in the scalar u and outputs a vector 2 dimensional vector u , u.

So, now, let us what f is, so f would be this case would be 0.2, - 0.2, - 0.2 and 0.2. So, why is that, because x1 - x2 is 1 - 0.8 is 0.2, 2 - 2.2 is - 0.2 and so on. And what is g, g would essentially take in the output of f, the encode, the f of x and g of f of x should ideally be equal to x for the training points x1, x2, x3 and x4.

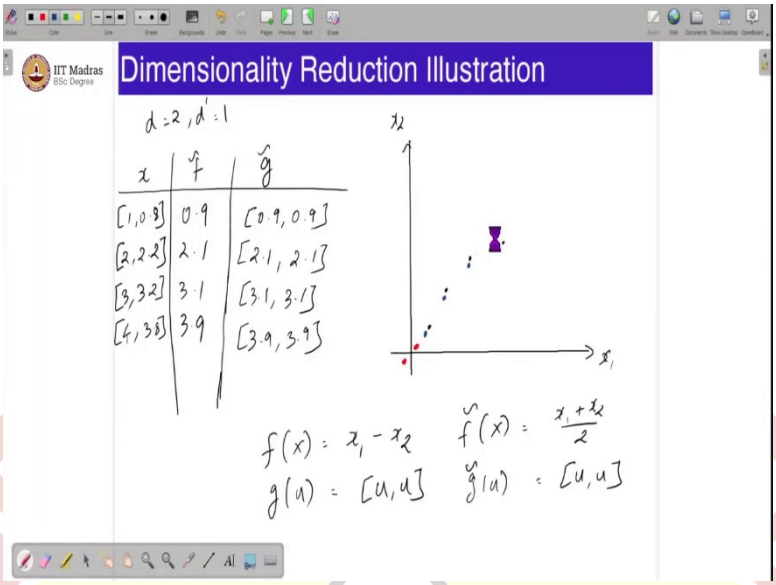
If you do that, for this the first training point g of 0.2 is the vector 0.2 , 0.2. You can immediately see that 0.2 , 0.2 is not equal to 1 , 0.8 and it is quite distance away. We can do the same thing for this the second data point will be - 0.2 , - 0.2 and the third - 0.2 , - 0.2 and 0.2 , 0.2 you have encoded this using the encoder and decoder given you have encoded the input and decoded it and this is what you get.

You can clearly see this is not a very good encoder and decoder because both 1 , 0.8 and 4 , 3.8 if you run it through the encoder and decode it again, you get the same 0.2 , 0.2 even though you have two very distinct inputs getting mapped to the same output. So, this is not doing a very good job of compressing losslessly.

So, it is doing some compression you are representing a two dimensional vector by a one dimensional scalar but, you are losing out, or I mean you are the goal is to get even though you lose, even though you store using lesser number of space or lesser number of numbers, but you want to get back the original input. You should, your goal is to not lose over the input, but on the other hand, but this particular example of encoder and decoder is not doing a good job of doing that.

Now, let us consider another encoder decoder path. Let us have f tilde and g tilde. f tilda of x is let us say, x1 + x2 by 2 and g tilde of u is u , u. If you do the same thing for f tilde and g tilde, let us see what that gives us. I will just erase this.

(Refer Slide Time: 13:23)



Tilda and g tilde, f tilde for the first data point will be x1 + x2 by 2 which is 0.9. 2 + 2.2 by 2 is 2.1. 3 + 3.2 by 2 is 3.1 and 4 + 3.8 by 2 is 3.9. This is the encoder output and the decoder if you apply it to the output of the encoder, it would be 0.9 , 0.9, 2.1 , 2.1. 3.1 , 3.1 and 3.9 , 3.9.

You can clearly see that this is a much better encoder decoder than the previous one. Why is that because 1 , 0.8 and 0.9 , 0.9 is pretty close and 4 , 3.8 and 3.9 , 3.9 is also pretty close and all the 2 , 2.2 and 2.1 , 2.1 is close. 3 , 3.1, 3.1 , 3.1 is also close.

Let us actually do a plotting for this. So, this is the x1 axis. This is the x2 axis you had 4 points which correspond to say 1 , 0.8, 2 , 2.2, 3 , 3.2 and 4 , 3.8. So, you had these 4 data points. If you had used the encoder and decoder given by f and g, the 4, f of g of x1, g of f of x1 would be, 0.2 , 0.2, g of f of x2 would be - 0.2 , - 0.2, and g of f of x3 would also be - 0.2 , 0.2, and g of f of x4 would also be equal to 0.2 , 0.2.

You would have the 4 outputs of the encoder decoder pair would essentially be these four numbers 0.2 , 0.2 and - 0.2 , - 0.2, which is clearly not a very good approximation. But on the other hand, if you do the same thing for f tilde and g tilda, you get really good numbers. So, you get for the first f tilde g tilde applied on x1, you would get 0.9 , 0.9, which is close to here.

And 2.1 , 2. 1, which is close here, I am not being very exact here to just get you the approximate idea of where the four points are. You can see that f of, f tilde of g, g tilde of f tilde of xi is very, very close to xi when compared to g of f of xi. So, we will say that g tilde , f tilde form a much form a better encoder decoder, or f tilde , g tilde form a better encoder decoder than f , g.

Once again, this is very simplified dimensionality reduction algorithm where we are already giving two pairs of encoder decoder f , g or f tilde , g tilde and asking the dimensionality reduction algorithm to choose among these two. In the reality, it would choose the best function from Rd to Rd' as for encoder and the best function Rd' to Rd for decoder, infinite number of functions.

That is what real dimensionality reduction learning algorithm would do. Here is a tie illustration where it just chooses between these two different encoder decoder pairs. So, with that, I think we can have the dimensionality reduction part of the thing.

Outline

1. What is Machine Learning??
2. The Wonders of Machine Learning
3. Data, Models and ML Tasks
4. Supervised Learning
    1. Regression
    2. Classification
5. **Unsupervised Learning**
    1. Dimensionality Reduction
    2. Density Estimation

Unsupervised Learning

- Unsupervised learning is ‘understanding data’
- Data: $\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n\}$
- $\mathbf{x}^i \in \mathbb{R}^d$
- Build models that compress, explain and group data.

Unsupervised Learning Application

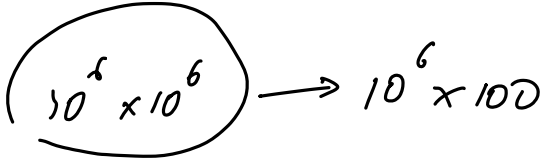
Tweet 1

⋮

Tweet 999999

Group the million tweets into 10 manageable groups

Dimensionality Reduction



E.g.: Represent a million gene expression levels of a million people, using just 100 numbers per person.

Dimensionality reduction: compression and simplification.

Dimensionality Reduction

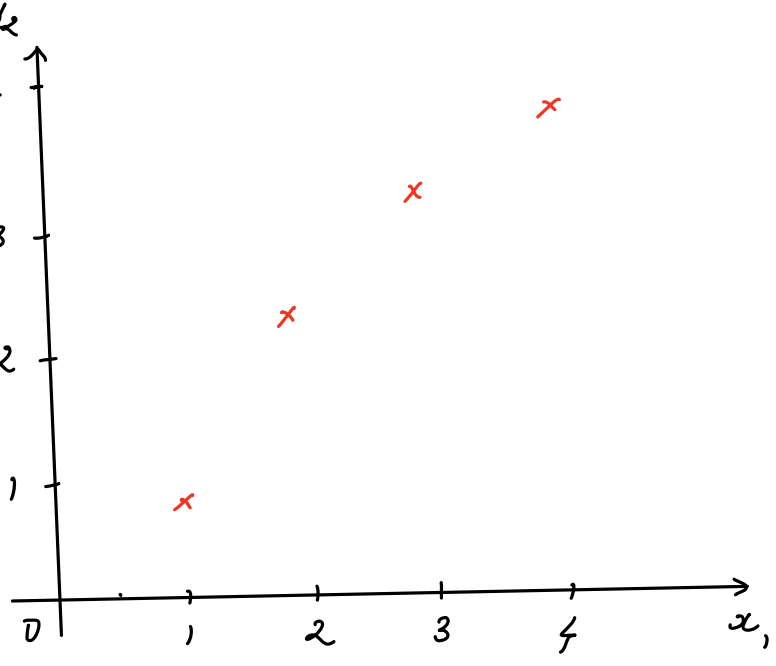
- Data: $\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n\}$
- $\mathbf{x}^i \in \mathbb{R}^d$
- Encoder $f : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$
- Decoder $g : \mathbb{R}^{d'} \rightarrow \mathbb{R}^d$
- Goal : $g(f(\mathbf{x}^i)) \approx \mathbf{x}^i$
- Loss $= \frac{1}{n} \sum_{i=1}^{n} \|g(f(\mathbf{x}^i)) - \mathbf{x}^i\|^2$

$d' << d$

Dimensionality Reduction Illustration

$d = 2$, $d' = 1$, $n = 4$

	f	g
$[1, 0.8]$	0.2	$[0.2, 0.2]$
$[2, 2.2]$	-0.2	$[-0.2, -0.2]$
$[3, 3.2]$	-0.2	$[-0.2, -0.2]$
$[4, 3.8]$	0.2	$[0.2, 0.2]$



$f(x) = x_1 - x_2$

$g(u) = [u, u]$

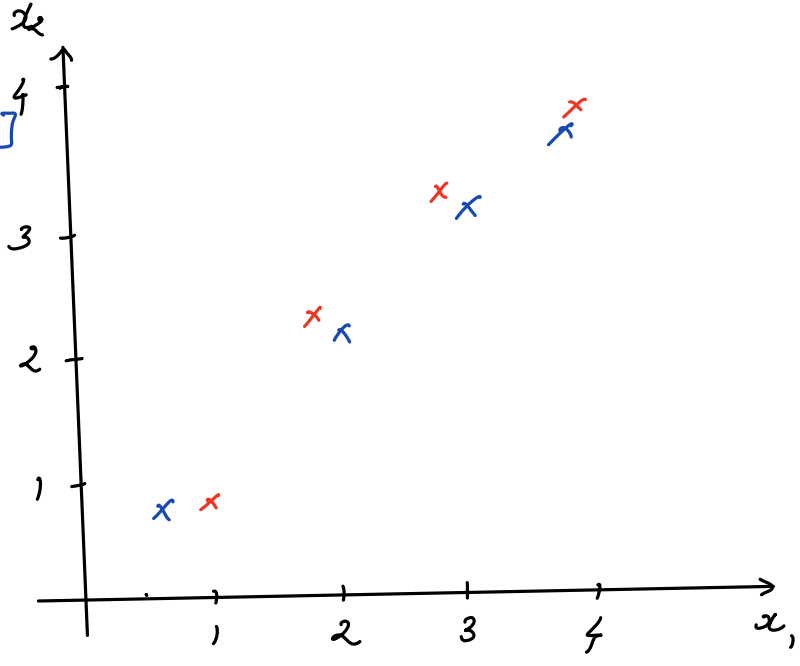
$\breve{f}(x) = \frac{x_1 + x_2}{2}$

$\breve{g}(u) = [u, u]$

Dimensionality Reduction Illustration

$d = 2$, $d' = 1$, $n = 4$

	$\tilde{f}$	$\tilde{g}$
$[1, 0.8]$	0.9	$[0.9, 0.9]$
$[2, 2.2]$	2.1	$[2.1, 2.1]$
$[3, 3.2]$	3.1	$[3.1, 3.1]$
$[4, 3.8]$	3.9	$[3.9, 3.9]$



$f(x) = x_1 - x_2$

$g(u) = [u, u]$

$\tilde{f}(x) = \frac{x_1 + x_2}{2}$

$\tilde{g}(u) = [u, u]$

$10^6 \times 100$

$\hat{f}$, $\tilde{g}$

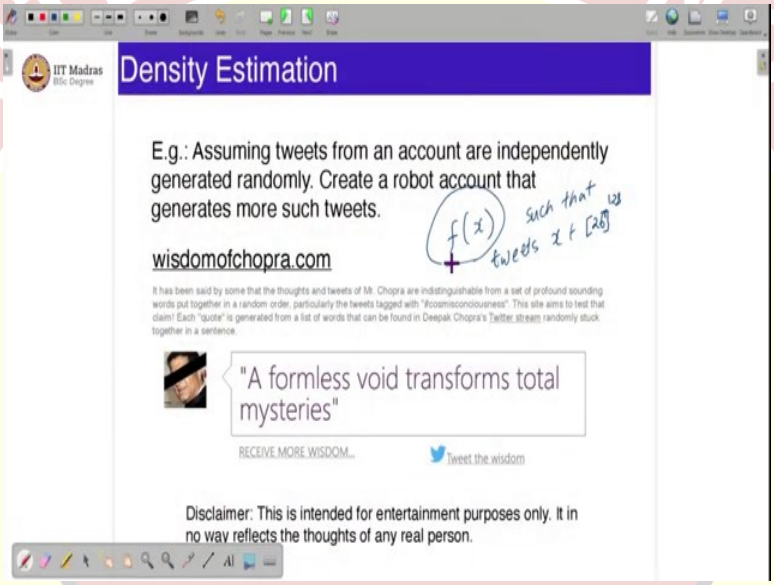
IIT Madras

ONLINE DEGREE

Machine Learning Foundations
Professor. Harish Guruprasad Ramaswamy
Department of Computer Science and Engineering
Indian Institute of Technology, Madras
Unsupervised Learning: Density Estimation

So, with that we wrap up the dimensionality reduction problem and move on to the next unsupervised learning problem, which is the density estimation.

(Refer Slide Time: 0:26)



Density estimation is, the output of a density estimation algorithm is a probabilistic model. Recall that I had mentioned earlier the probabilistic model is a model which essentially scores different configurations of reality. For example, let us say your goal is to build a model, which can generate more tweets, which look very similar to already generated tweets. So, assuming tweets from an account are independently generated randomly, your goal is to create a robot account that generates more such tweets.

Here is such a site built called wisdomofchopra.com, where it has been said by some that the thoughts and tweets of Mister. Chopra are indistinguishable from a set of profound sounding words put together in a random order. So, based on this, they built a probabilistic model, and which generates tweets that look similar to the tweets of Mister. Chopra.

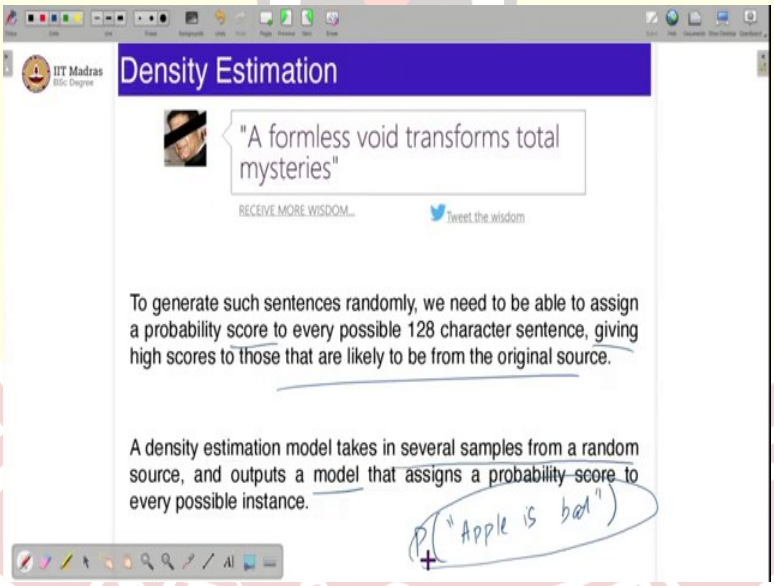
This is an example of how you can use a density estimation or a probabilistic model to generate robots. So, whenever someone wants to generate a robot Twitter account, effectively, a density

estimation is the main tool that is used here. So, what is, what does a density estimation algorithm give you? It gives you a probabilistic model, which scores all tweets, effectively, tweet is something, is a is 128 length array.

So, that is, let us say, for example, a tweet is 128 length character array, which in this case, what you need is a scoring function, which would score any x such that tweets x belonging to 26^{128}. So, I am assuming that you are writing always in lowercase. So, you take there are 26 possible outputs for each space 27 if we include a space and onetwo8 characters, so that is the total number of legal tweets possible.

So, your f of x would essentially take such a thing, and it would say how likely was this tweet, how likely is that this particular tweet can be generated by Mister. Chopra. So, that is how do you get such a, this is a probabilistic model. And constructing such a probabilistic model is the goal of density estimation.

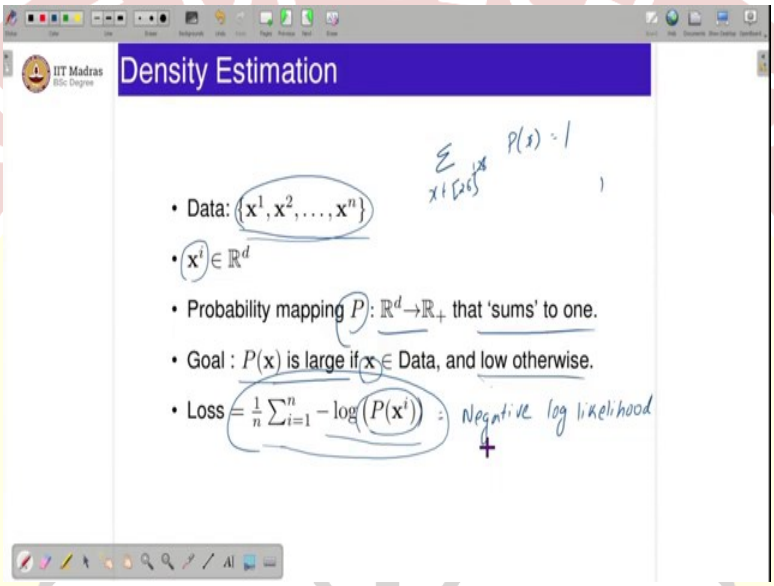
(Refer Slide Time: 3:15)



To generate such sentences randomly, we need to be able to assign a probability score to every possible onetwo8 character sentence giving high scores to those that are likely to be from the original source. How does it do that? Well, a density estimation model takes in several samples from a random source, but in this case, it would take several samples from the original Twitter stream and it will output a model probabilistic model that assigns a probability score to every possible instance. So, by every possible instance, we mean every tweet.

So, we can give, we can say, what is the probability that "Apple is mad"? It should be able to the output of the learning algorithm in the case of a density estimation would simply be a function which would be able to say what is the probability that Apple is bad? So, this is a sentence Apple is bad is one possible output. And you can ask the question, what is the probability that this particular tweet is generated by Mister. Chopra.

(Refer Slide Time: 4:18)



In mathematical terms, once again, your data is it should be a collection of tweets maybe, collection of n tweets, x1 to xn, where each data point is a d dimensional vector. The goal of a density estimation learning algorithm is a probability mapping P, which is a mapping from $\mathbb{R}^d$ to $\mathbb{R}^+$, so it is only positive numbers are allowed. And here is the key thing. It must sum to 1.

That is the reason why you, that would mean that sum over x in 26^{128} probability of x should be equal to 1. So, that you cannot give very high scores to all the tweets, that is just not possible. Because each tweet has to have a positive score or 0 score, but the total probability of, the total score has to be equal to 1. So, that is the goal, the goal of a density estimation algorithm is to come up with such a probability model.

And what property models are desirable? What is the last function for this? The goal is that if P of x is large, if your x belongs to this data, and it should be low otherwise, that is the goal here. And of course, it can be large. Otherwise, also, if the data is some x is very similar to an already existing

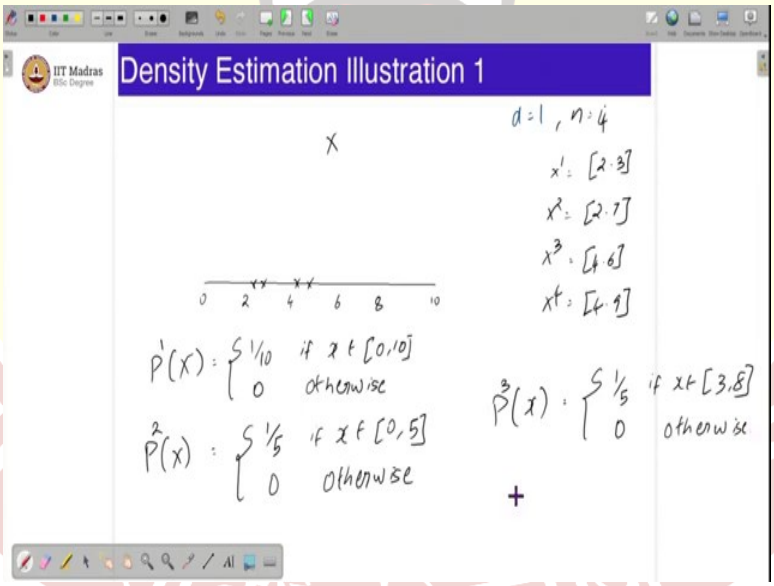
data. So, the last that is used is called the negative log likelihood, which is simply defined as follows.

For each data point, xi, you want your P of xi to be as large as possible. And in other words, you want your negative log of P of xi to be as small as possible. What is the ideal case scenario, the ideal case scenario would be you would have log of, P of xi would be equal to 1 by m, does not matter.

So, P of xi would be very high for x1 to, P of x1 would be high, P of x2, P of x3 would be high, P of xn would all of that would be high. So, that - log of P of xi would be small. And when you sum this over an average over all the n data points, this would be small. So, this is the average negative log likelihood. Your goal of a density estimation algorithm is to come up with a probability model or a probability mapping P, which has a small negative log likelihood as possible.

Here is have a couple of examples of how to do that. Here is a very simple illustration problem.

(Refer Slide Time: 7:05)



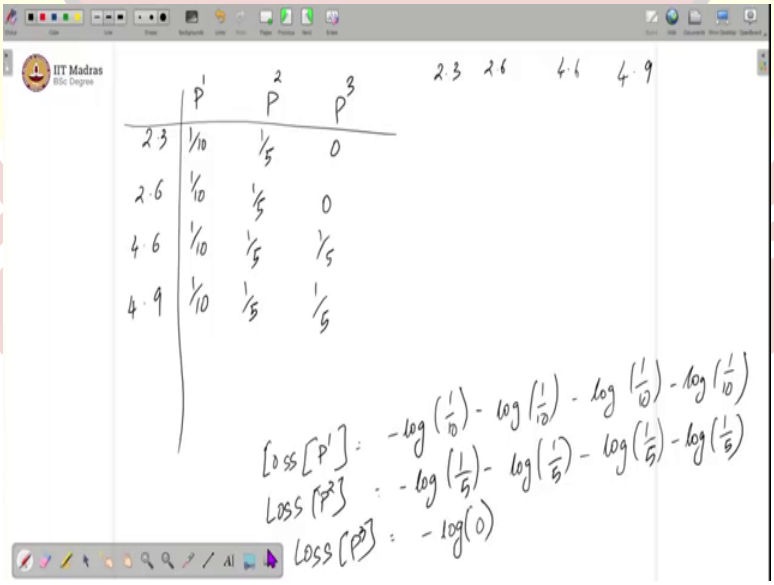
Let us say you have your, in this case, D = 1, you have just one-dimensional data. And let us say your data, you have 4 data points, this n equals 4 the data points are given by, let us say, x 1 = let us say 2.3, x, sorry about the superscript should be a superscript. Say x1 = 2.3, x2 = let us say 2.7, x3 = let us say 4.6, x4 is let us say equal to 4.9.

So, you have 4 data points, let us maybe plot that as on the number line, from 0 to 10, you have 2.3, and 2.7. And you also have 4.6, and 4.9, you have these 4 data points. Now, once again, in the spirit of things that we have done so far, we will give a bunch of models, and we will ask the density estimation algorithm to choose the best model out of these, so what are the models here?

So, the first model we can consider is that P of x = 1 by 10, if x belongs to 0 to 10, and 0 otherwise. So, I forgot to mention so in this case, you are, it is not a discrete thing like a tweet where you can take only 1 of 26 possible values. So, here your X can take a real valued thing. X is a real valued instance. So, summing to 1 is essentially replaced by integrating to 1, so that is why I have 1 by 10, 0 to 10. So, if you integrate this function over the rail line, you will get the value would be equal to 1. So, that is why I am using 1 by 10. So, this is one possible probability model.

Let us have another probability model P of x =, let us say, 1 by 5, if x belongs to 0 to 5, and 0 otherwise, this is also a valid probability model. So, then we will use, finally we will use another model called P1, P2 and let us call this P3. P3 of x = 1 by 5. If x belongs to let us say 3 , 8, x is between 3 and eight, it is 1 by 5 and 0 otherwise. All these three are valid probability models. Now, I just want the learning algorithm to choose between these three models. Which of these three has the lowest negative log likelihood? So, if you could compute that. Let us do that.

(Refer Slide Time: 10:26)



Let us recall the data once again, it is 2.3, 2.6, 4.6, 4.9. What would the scores of each of these components 2.3, 2.6, 4.6, 4.9. According to P1, P1 is just uniform between 0 to 10, so it would

give a score of 1 by 10, 1 by 10, 1 by 10, 1 by 10. For all of these 4 points. P2 is uniform within 0 to 5, it will give a score of 1 by 5, 1 by 5, 1 by 5, 1 by 5.

P3 is uniform in the range of 3 to 8. Which means that, according to P3, the score for 2.3 and 2.6 should actually be 0. And 1 by 5, 1 by 5, we can compute the negative log likelihood that is loss of the model P1 would be - log 1 by 10 - log 1 by 10 - log 1 by 10 - log 1 by 10. So, this would be the loss of the probabilistic model P1. What would be the loss of P2?

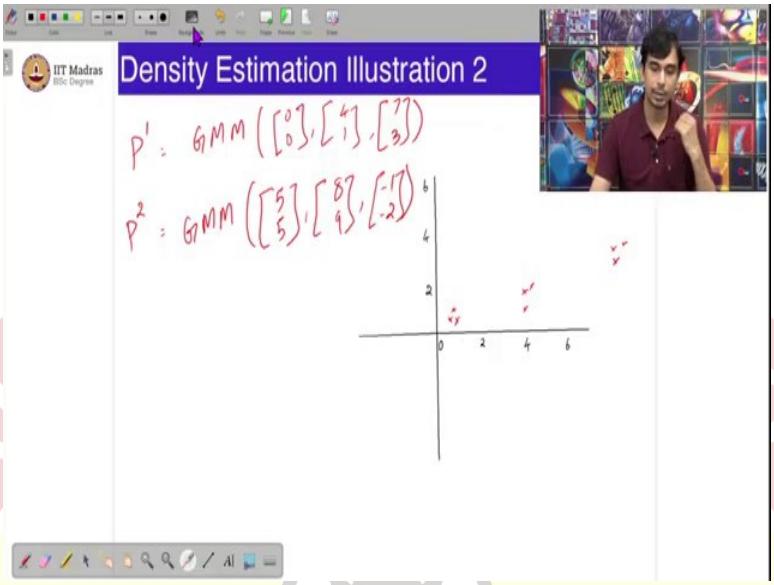
That would be - log 1 by 5, - log 1 by 5, - log 1 by 5, - log 1 by 5. What would be the loss of P3? Here is, I think here you would have - log of 0, it actually does not matter what you write for the rest of the things because - log of 0 is already infinity, log of 0 is - infinity and - log of 0 is infinity. So, the loss of the model P3 is going to be infinity, regardless of how the other things are going to be.

So, you can clearly see that loss of P1 and loss of P2 are going to be not infinity. So, which is better, we can clearly see once again that loss of P2 is going to be smaller than loss of P1. So, P2 is the better probabilistic model for explaining this data than P1. And it seems logical, P1 says that these 4 data points are coming from uniformly between 0 to 10.

P2 is saying that these data points are coming uniformly between 0 to 5. P2 seems to be a better model than P1 for this particular data. Of course, I am not saying that is the best model in the context of this particular type problem P2 is better than P1 because P2 has lower negative log likelihood than P1.

So, in a real density estimation model would not once again be given multiple choice questions like this, it would not be given P1, P2 and P3, you would actually be, it would actually find the best probabilistic model from an infinite set of models. So, that is one example. Let us move on to another example.

(Refer Slide Time: 14:16)



So, here is another example. Let us say you have 9 points. So, here is a slightly more complicated class of models, which you can estimate. Let us say you have 9 data points, I am just finding and I am not going to write down the 9 points, here these are the 9 points. I have put an X in the Cartesian plane for illustration these 9 points, so you can have two models now.

One model is a Gaussian mixture model which says that the data is centered around 3 P1 is a, I am not going to explain what the Gaussian mixture model is right now, but broadly wanted to saying is, the Gaussian mixture model centered at 0 , 0 and 4 , 1 and let us say 7 , 3. That is one Gaussian mixture model. And there is another Gaussian mixture model, which says that this is data centered around 3 possible centers, just let us say 5 , 5, 8 , 9, - 1 , - 2, which is a better model?

I am not going to go into the numerical details, but you can immediately see that out of these two models, you have, you can look at the data and you can see that it seems to have 3 centers, one center around here and one center around here and another center around here. So, based on this P1 is a much better explanation for this data than P2.

So, you are, if you had actually computed the negative log likelihood of these two models P1 and P2 and chosen the model with the lesser negative log likelihood, you would have chosen P1. Once again real density estimation models do not have multiple choice questions, but rather they are given entire parametric family of models and they choose the model among this infinite models, which has the least negative log likelihood or which has the least loss in this particular case.

This is once again there are several extensions possible to this based on here in the previous case, we had uniform models that are uniform, here in this case, we had the models were all Gaussian mixture models. You can have several other examples in the same way that you had for regression and classification and dimensionality reduction. You can have similar other examples for density estimation also.

And with that, I think you have a broad overview of the two main unsupervised learning problems, which are dimensionality reduction and density estimation. With that will wrap up the introductory week of the Machine Learning Foundations course. And later on in the course, we will dig deeper into all of these learning problems and actually come up with ways to formulate the learning algorithm in the full power that is to find the best model in the entire model, in the entire class of models, in the entire library of models.

Which is how do you write, how do you construct learning algorithms, which finds the best model among an infinite family of models and that we will cover later in the course when we do individual problems which are highlighted as a special cases of the tools that you will learn throughout the course. That will wrap up the first week. Thank you.

Density Estimation

E.g.: Assuming tweets from an account are independently generated randomly. Create a robot account that generates more such tweets.

$f(\text{Tweet})$ = Score of the tweet

wisdomofchopra.com

It has been said by some that the thoughts and tweets of Mr. Chopra are indistinguishable from a set of profound sounding words put together in a random order, particularly the tweets tagged with "#cosmisconciousness". This site aims to test that claim! Each "quote" is generated from a list of words that can be found in Deepak Chopra's Twitter stream randomly stuck together in a sentence.

"A formless void transforms total mysteries"

RECEIVE MORE WISDOM...

Tweet the wisdom

Disclaimer: This is intended for entertainment purposes only. It in no way reflects the thoughts of any real person.

Density Estimation

"A formless void transforms total mysteries"

RECEIVE MORE WISDOM...

Tweet the wisdom

To generate such sentences randomly, we need to be able to assign a probability score to every possible 128 character sentence, giving high scores to those that are likely to be from the original source.

A density estimation model takes in several samples from a random source, and outputs a model that assigns a probability score to every possible instance.

Density Estimation

- Data: $\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n\}$
- $\mathbf{x}^i \in \mathbb{R}^d$
- Probability mapping $P : \mathbb{R}^d \to \mathbb{R}_+$ that 'sums' to one.
- Goal : $P(\mathbf{x})$ is large if $\mathbf{x} \in$ Data, and low otherwise.
- Loss $= \frac{1}{n} \sum_{i=1}^{n} -\log\left(P(\mathbf{x}^i)\right)$

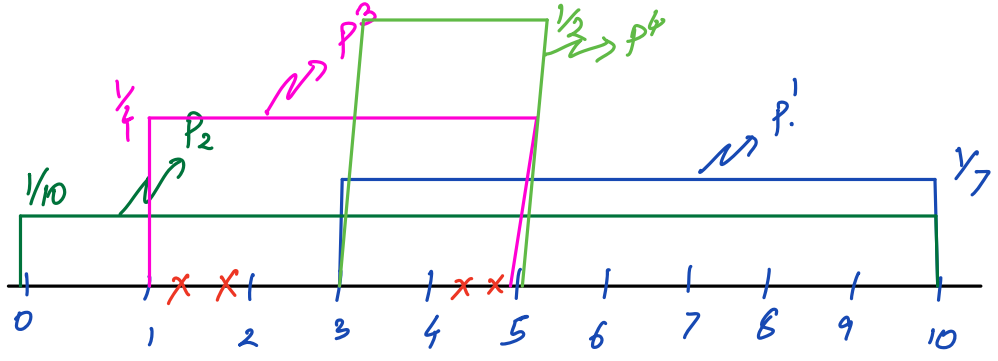
Σ sum over all tweets $\{a..z\}^{128}$

$P(\text{tweet}) = 1$

$P(x^i)$ is large.

$P(\text{anything}) = 10^{10}$

Density Estimation Illustration 1



$x^1 = [1.2]$

$x^2 = [1.9]$

$x^3 = [4.3]$

$x^4 = [4.8]$

p^1 = Uniform in $[3,10]$ $0, 0, \frac{1}{7}, \frac{1}{7}$

p^2 = Uniform in $[0,10]$ $\frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10}$

p^3 = Uniform in $[1,5]$ $\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}$

p^4 = Uniform in $[3,5]$ $0, 0, \frac{1}{2}, \frac{1}{2}$

$loss[p^4] = loss[p^1] = \infty > loss[p^2] > loss[p^3]$

Density Estimation Illustration 2

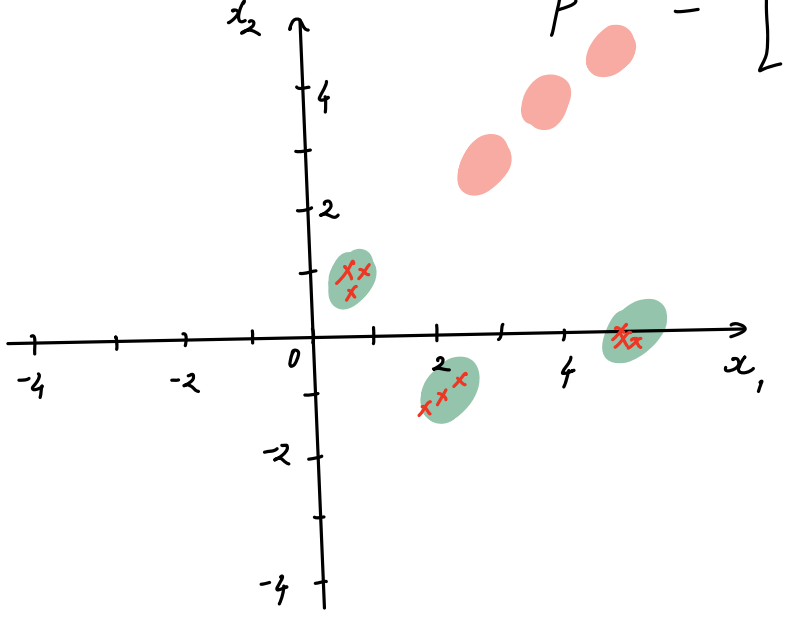
$d = 2$

(1.1, 1.3)
(0.9, 0.7)
(2.1, -1)
(5.1, 0.1)
(2.2, -0.9)
(5.1, 0.0)
(0.9, 1.2)
(1.9, -1.1)
(4.8, -0.1)

Gaussian Mixture Model

$p^1 - \begin{bmatrix} 3 \\ 3 \end{bmatrix}, \begin{bmatrix} 5 \\ 5 \end{bmatrix} \begin{bmatrix} 4 \\ 4 \end{bmatrix}$

$p^2 - \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ -1 \end{bmatrix} \begin{bmatrix} 5 \\ 0 \end{bmatrix}$



Clustering

Outline

1. What is Machine Learning??
2. The Wonders of Machine Learning
3. **Data, Models and ML Tasks**
4. Supervised Learning
    1. Regression
    2. Classification
5. Unsupervised Learning
    1. Dimensionality Reduction
    2. Density Estimation

What is Data?

Data is a collection of vectors.

E.g.

3	9	1.9	5.0	House 1
2	7	2.1	3.2	House 2
4	12	2.8	6.6	House 3
5	16	0.9	9.8	House 4
5	15	3.1	8.5	House 5
4	11	1.6	6.9	House 6

Metadata is information on the data.

E.g. : (# rooms, Area in 100 sq.ft, Distance to metro in km, Price in 10 lakhs)

What is a Model?

A model is a mathematical simplification of reality.

Some examples:

The Ideal Gas model

Inverse square law for gravitational attraction

Moore's Law for semiconductors

Cobb–Douglas model in Economics

"All models are wrong, but some are useful"

George Box

Types of Models in ML

- Predictive Model
  - Regression Model
  - Classification Model
  - …
- Probabilistic Model
  -

Predictive Models

Regression Model

Model the price of a house based on its area and distance to metro.

Example good model:

Price = 0.5*Area - Distance

Predictive Models

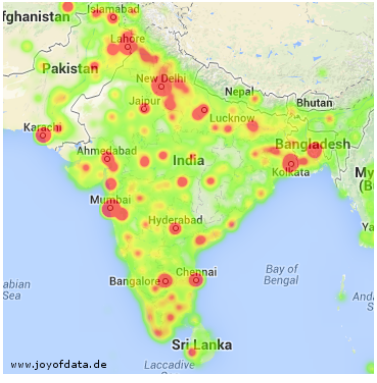
Classification Model

Model whether a house is closer than 2kms to a metro based on price and area

Example good model:

Answer = Close if 2*ROOMS-PRICE < 1
Far otherwise

Probabilistic Models



What is the probability that a randomly chosen person is in lat-long : (25N,30E) ?

"A formless void transforms total mysteries"

RECEIVE MORE WISDOM...

Tweet the wisdom

What is the probability that a given tweet was generated by Mr. Chopra?

Learning Algorithms

Learning Algorithms: Data → Models

Choose from a collection of models, with same structure but different **parameters**.

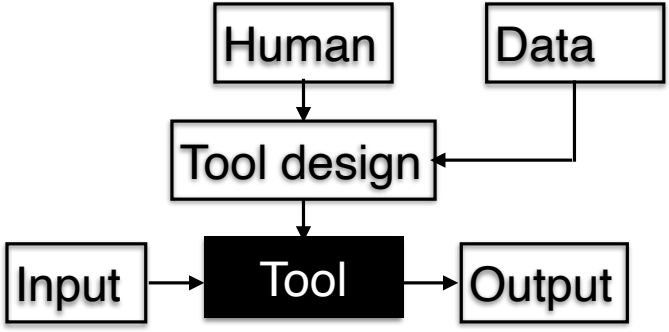
E.g.
Price = a*(area) + b*(# rooms) + c*(distance to metro)

Parameters: a,b,c

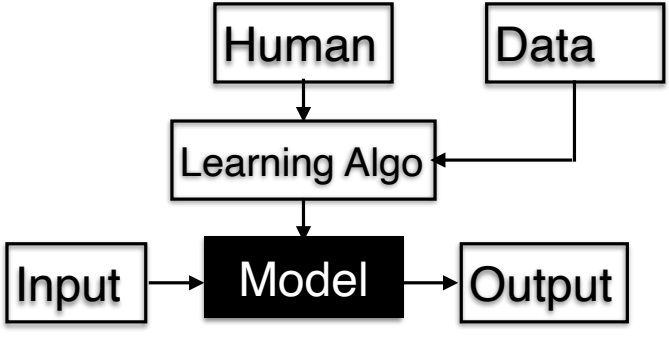
Use data to get the "**best**" parameters

Machine Learning Tasks Revisited

Machine Learning



Machine Learning

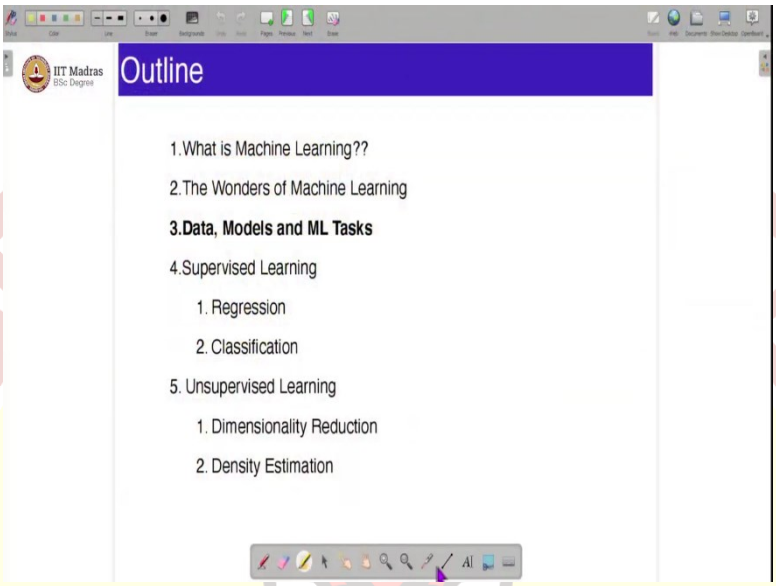


IIT Madras

ONLINE DEGREE

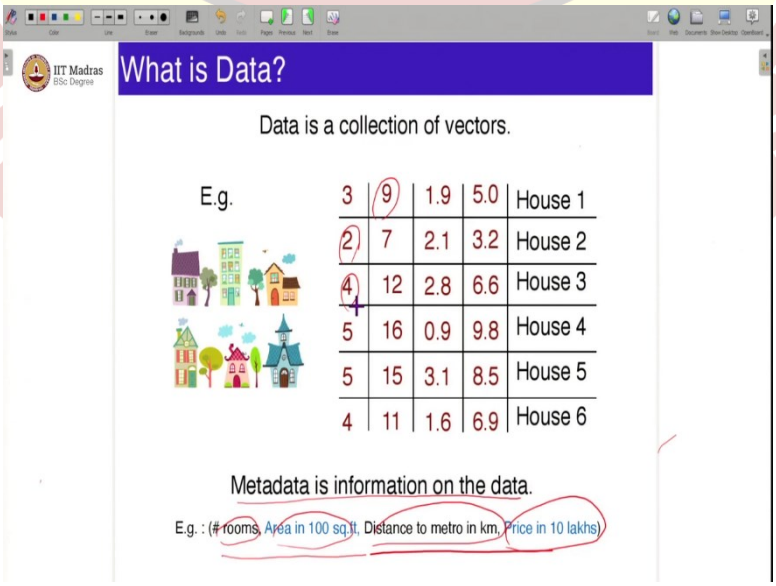
Machine Learning Foundations
Professor. Harish Guruprasad Ramaswamy
Department of Computer Sciences & Engineering
Indian Institute of Technology, Madras
Data, Model and ML Task

(Refer Slide Time: 0:26)



Hello, everyone, and welcome to another video of foundations of Machine Learning. In this part of the course, we will define some fundamental terms which have which are, what is data? What is models and what do ML tasks mean? These are the some of the most important terminologies that you will encounter when talking to people on machine learning.

(Refer Slide Time: 0:46)



The first question that must come to mind is, what is data? Because this is so prevalent in machine learning and data science that it has the risk of running out of meaning. Data in its most general sense can pretty much mean any collection of bits or bytes. But in the context of machine learning, data will almost always mean a collection of vectors.

For example, let us say you have a bunch of houses, you have let us say you have 6 houses, and each house is represented by a vector. The example house 1 is represented by a vector, a four-dimensional vector, just 3,9,1.9,5.0. This is an example vector. And similarly now house 2, which is represented by the vector 2,7,2.1,3.2.

Similarly, you have 6 houses, each house is represented by a vector. And you have all of these put together is a collection of vectors. And this is an example of a data or a data set. By itself, this is sufficient for a computer to work with, for most reasons, for most purposes, but humans require a little bit more understanding.

That is, for example, you are saying that house 1 is represented by this vector 3,9,1.9,,5.0, but what does it even mean? So, this is just a bunch of numbers, but what does it mean in reality? This information is typically contained in metadata. Metadata is information on the data. It is self-referential in nature, because data already is information. So, metadata is information on the information.

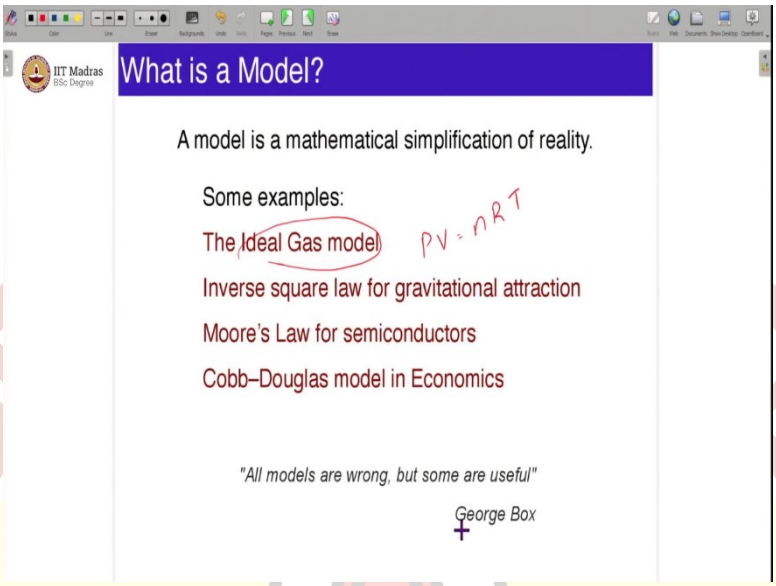
For example, in this particular case, it might contain information on what the numbers mean. For example, it might say the first number refers to the number of rooms in the house, the second number represents the number, the area in hundreds of square feet, the third number represents distance to the nearest metro in kilometers, and the fourth number represents price in tens of lakhs.

House, I mean, representing each house is a vector is abstract, but when you put that together with a meta data, it tells a story. What this says is house 1 is a three-bedroom house with 900 square feet, at a distance of 1.9 kilometers from the metro and its price is 50 lakhs. For this metadata makes essentially makes a data human interpretable, but as far as the computer is concerned, it does not matter.

Because these meaning of these words really is not useful to the computer in any way. The only thing that matters is that, it is consistent. That is, if you are using the number one and the first number for representing number of rooms for house 1, you consistently keep doing that for house 2, house 3 and house 4. As long as you do that the data is useful for the computer. It

does not need to know the information about these numbers, but this, the metadata is still useful for humans to understand the data.

(Refer Slide Time: 3:56)



Now, next, the next main player in this, in the field of machine learning is a model. What is a model? So, a model is, has lots of it is based in, it is in fact the core player in most of science. And in fact, models have been used for thousands of years. For example, the most, the simplest way to express a model to define a model is simply called a model, a mathematical simplification of reality. It is almost always representing reality, but it is simpler than reality, and it is more compact than really.

The most famous model perhaps, that all of you might have heard of is the is the Ideal Gas model, which is essentially something on the order of pressure into volume is equal to number of moles, × the gas constant × temperature. So, this is an equation that you might have seen in your school. This is an equation that is satisfied by an ideal gas. In reality, there exists no ideal gas, all gases have non-idealities, but it still captures an important aspect of a lot of gases to a reasonable approximation that this forms a major progress in science.

The other example of a model is, the inverse square law for gravitational attraction that is the Newton's formula, which says that the gravitational attraction force decays $\frac{1}{R^2}$ there R is the distance between the two bodies. This is also a model, it tries to model a very real physical thing, which is gravitational attraction or a gravitational force in terms of mathematical formula.

We know for a fact that it is not exact because it does not account for example, for the speed of the object, it does not account for the curvature of space time and so on. It does not do that, but despite its inaccuracies, it is still very useful that. The Newton's gravitational model is very useful because it serves a purpose and explains a whole lot of reality, even if not all of reality.

And similarly, there are other laws, for example, the Moore's law for semiconductors simply states that the number of semiconductors, number of transistors in a chip is going to double every few months and so on. This is clearly not a physical law, it does not, it is not exact, but the trend of that is ascribed to Moore's law is still useful, because it pretty concisely captures a trend which held on for several decades. And models are actually the bread and butter in most of economics.

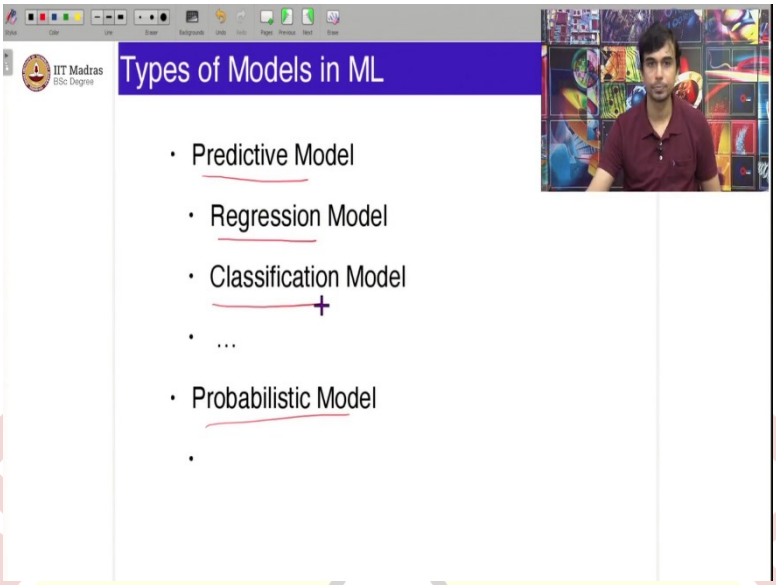
The basic idea of economics everyone is aware of, that is, if you increase the supply of a product, the price goes down, if you increase the demand of a product the price goes up, but how much. If you increase the supply by 10 percent how much is the price going to go up? If suddenly this product becomes popular among one group of people how much is the price going to go up?

These are questions which you do not know the exact answer. And to answer these questions exactly, economists regularly go to models, which are essentially mathematical simplification of reality. They are not reality. Reality is not very nicely expressible in a few equations, but equations can be studied, equations can be analyzed and you can plan with equations, but you can never plan with the full reality.

So, there is this very famous statement attributed to George Box a famous statistician who said that all models are wrong, but some are useful. So, no model is exact because models by definition are simplifications, by models by definition are approximations. But despite their approximations, some of them are useful. For example, ideal gas model or the gravitational, Newton's gravitational model are very useful despite their approximation despite their errors, because they give a sense of law, I mean, give a sense of order to that chaos around.

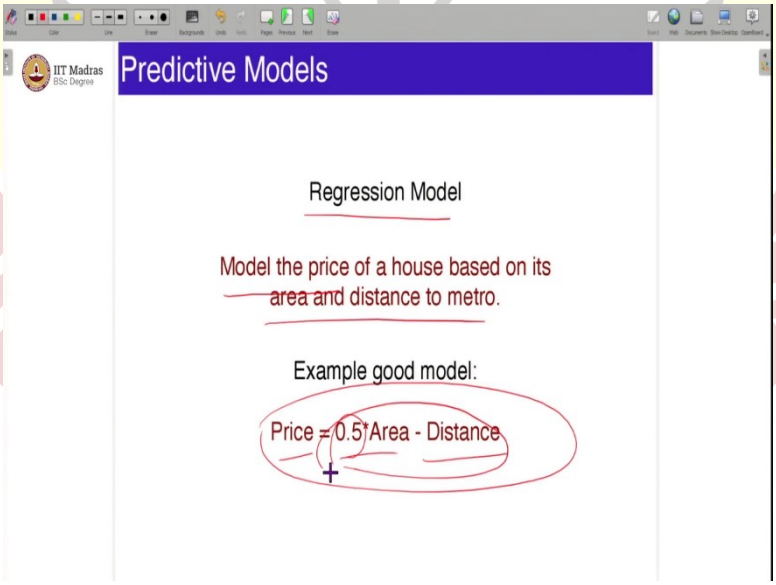
So, this is model in the method in the scientific or mathematical sense. In the machine learning world, when we mean a model, it is related to the scientific model sense, but we mean it in a different slightly different sense.

(Refer Slide Time: 8:11)



That there are two main types of models that are used in machine learning. They are the predictive model and the probabilistic model. There are several types of predictive models. The two main ones are the regression model and the classification model. There are several other types of models, which we will not look into. Now we will see the example, I mean, what do we mean by regression model and the classification model.

(Refer Slide Time: 8:41)



A regression model is a type of predictive model. What kind of thing does it do? Well, an example of Regression Model would model the price of a house based on its area and distance to metro. This is the goal of elevation model here. It models the price of a house based on its

area and distance to a metro. An example good model would be something like the price of a house is $0.5 \times$ area $-$ distance to metro.

Do not worry too much about the coefficients, they do not matter. What matters is the trend that is here, you have a situation where the price of the house increases with area and decreases with distance to metro, which is a reasonable approximation. So, the price of a house will increase as the horse gets bigger, but it will get cheaper as you move farther and farther away from the city, which is approximated by distance to metro.

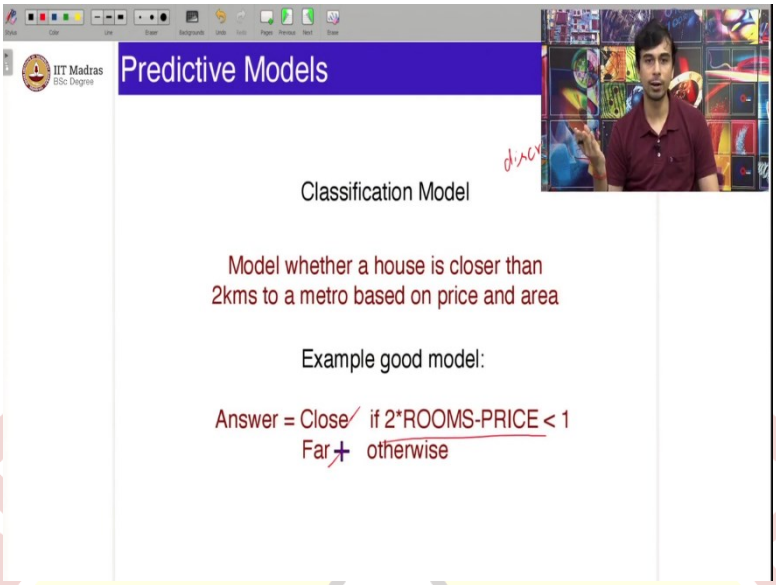
And clearly you can see this is not a law of physics or this is not always solved, means not always. There are going to be a lot of exceptions to this rule, but despite the exceptions, this is a general rule this is meaningful. You can, if you are looking for a cheap house, you have two options now. You can either look for a reasonably big house far away from the city or super small house in the middle of the city. This a kind of broad conclusions are possible with the model. With reality, you can never do that.

With reality if you are looking for a reality, you will just have to look at all prices of all the houses and then have to make a decision. But with a model you can have a good sense of what to look for. Let us say you have a budget and you can decide, okay, with this budget I can either look for a large house far away or a small house close by. That is the power of a model.

And a regression model. Why is it called a regression model? Well, regression models predict a real valued, price is a real value. It lives on a continuum. It can be 5 lakhs, it can be 5 lakh 50,000, it can be 5 lakh 50,223. So, a price it lives on a continuum, it is a, when you want to predict a real valued variable what you need is a regression model. And regression model is a type of a predictive model because it predicts the price.

For example, if someone says I have this house, which is 700 square feet at a distance of three kilometers from the metro, you can use this model to predict the price of this new house. Even though you have no, you do not have any information on its price, you do have information on its area and the distance of the house from the metro, you can use that to your advantage along with a good model to predict the price of this house. So, that is why it is called a predictive model. You can use the model to make predictions on unknown data on unseen data at least.

(Refer Slide Time: 11:33)

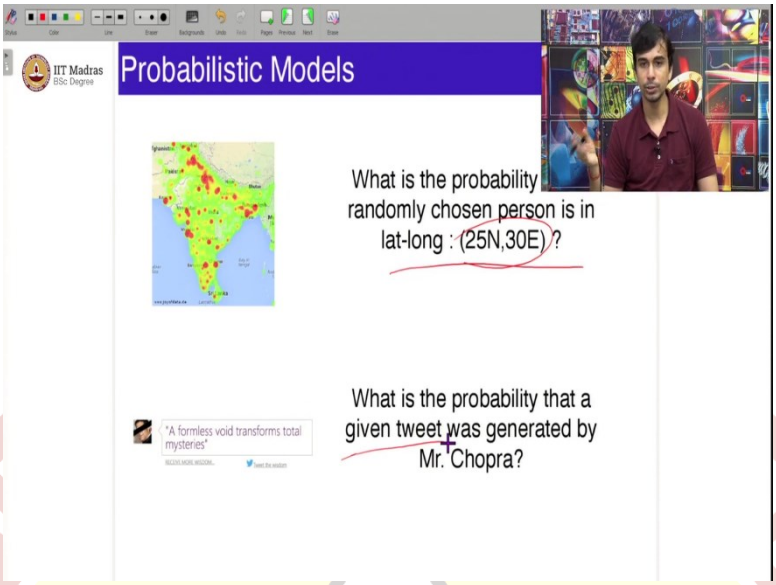


The other type of predictive model is a classification model. A classification model is very similar to a regression model. The only difference is that the variable that you are predicting is no longer real value, the variable that you are predicting is discrete in nature. For example, you might want to predict whether a house is closer than 2 kilometers to a metro based on price and area. So, that is your, you want, you are only interested in two possible outputs, that is, it is closer than two kilometers to a metro or farther than two kilometers to a metro. So, that is the only two things that you are interested in.

An example good model might be you might say, if $(2 \times \text{rooms} - \text{price}) < 1$, then it is close otherwise, it is far. So, what is it really telling? It says that if the number of rooms is small, and the price is high, so that is what this is saying. If you have a small, you have like a one-bedroom apartment costing 1 crore rupees, it is quite likely that it is close to the city. So, that is what this model is saying.

And if not, it is perhaps far away from the city. So, in this case, you can see that the variable that you are predicting has only two possible values, which is either close or far. Unlike the case of the price, which you are predicting a real valued number. A price is a real valued number just printing whether it is close or far is just there are only two possible options. That is why it is called, such models are called classification models.

(Refer Slide Time: 13:00)



Now we'll move to what are probabilistic models. Probabilistic models are slightly different from predictive models, that is, their goal is not to predict the future. You can use both classification models and regression models to predict on future data. For example, if someone gives a new house, the details of a new house, you can use their number of rooms and the price to predict whether it is close to a metro or not. Even though you have not used, not seeing that house before you can use the classification model to make a prediction.

Probabilistic models, on the other hand are not tuned towards making a prediction. Their goal is slightly different. Their goal is to evaluate how likely a certain event or a configuration is. For example, someone might pick a randomly chosen person from India, and they can ask the question, what is the probability that a randomly chosen person is in the latitude longitude given by 25 ° north and 30 ° east.

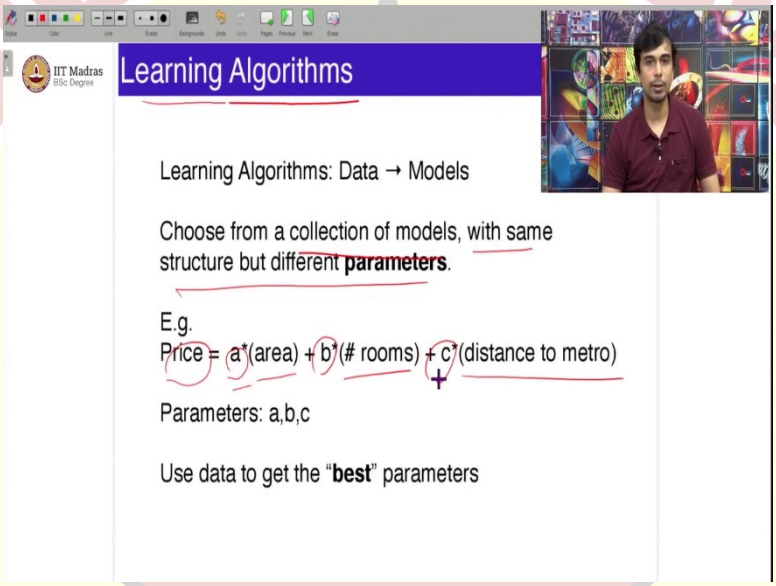
The probability of this 25 ° north and the degree is east let us say if it is in the middle of the Sahara Desert is going to be quite low, and if this is in the middle of Bombay, it is going to be quite high. Based on different configurations of the latitude longitude, some latitude, longitude are much more likely to have a randomly chosen person rather than other latitude longitudes.

So, this essentially scores reality. Probability models essentially score reality, and you can give any configuration of reality. For example, 25 ° north and 30 ° east, and it can say how likely is this particular reality? Another example might be, for example, you might give a tweet and ask the question, what is the probability that a given tweet was generated by Mister. Chopra? So, that is the question that they can ask.

And there is, the broad idea is that tweets that are similar to tweets that Mister. Chopra would tweet would get a high score, and tweets that are completely random collection of digits and numbers, digits and characters are likely to have a low score. So, a probability model simply scores different configurations and it has other constraints also, but that is the broad again, you should take.

That is, if you have a probabilistic model will effectively score different configurations of reality. So, now we come to the major player, another major player of machine learning, which is a learning algorithm.

(Refer Slide Time: 15:36)



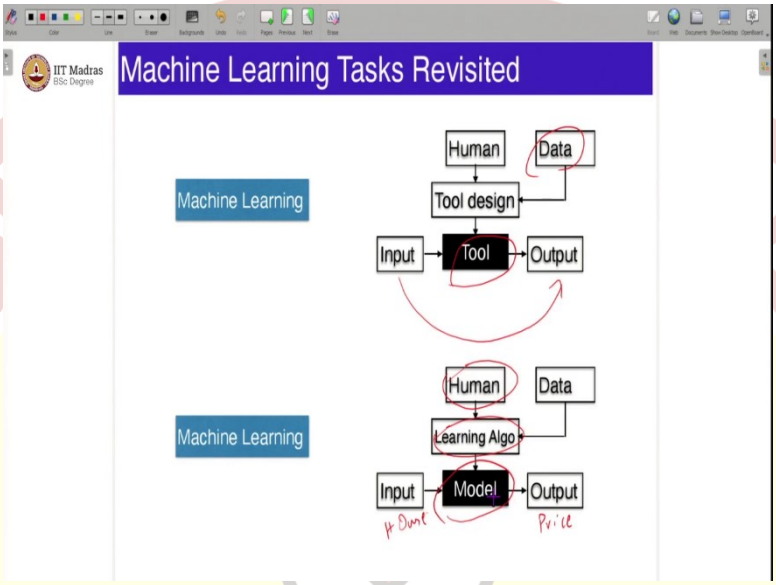
What is a learning algorithm? A learning algorithm is what converts data into models? In most machine learning tasks, what you are interested in is, is getting a model. And the learning algorithms are the tools that learning algorithms are the things that convert data into models.

And how does it do that? Well, typically, they choose from a collection of models. There is a whole bunch of models, and learning algorithms consider all these models. And among these models, they give the best model. The typical way it has done is, it chooses from a collection of models with the same structure, but different parameters.

For example, let us say, you are you are building a learning algorithm for predicting the house of, price of the house. You can, before you even start, before you even look at any data, you can say, well, I am going to decide that my price of the house is A × the area of the house, + B × the number of rooms in the house, + C × the distance to metro.

Before you even see any data, you can just decide this in advance. You do not know A, B and C. But before even looking at any single data you can decide this. And what the learning algorithm does is, it takes a look at all the data and then decides the correct values of A, B and C. And these A, B and C are called parameters that represent or that characterize the model, so that is a learning curve.

(Refer Slide Time: 17:12)

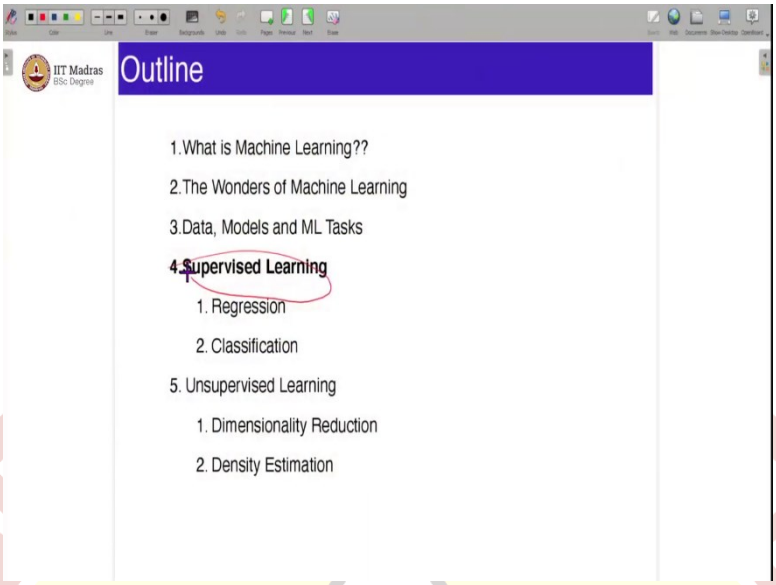


Now, we can actually put all of this in context with our task hierarchy. In the machine learning task hierarchy, we said that the tool which transforms input to output is not designed directly by the human but rather the human, rather only gives a broad outline to a tool design, which in turn uses data to create to tool.

Now, we have seen enough terminology of machine learning to actually give more precise names. In our case, the tool is simply the model. The model, for example, let us say the price of a house prediction model that will take a new house and it will predict a price. So, this is a house and price. But what machine learning does is you do not, the human does not directly write the model. It does not say the price of the house is 3×1 square feet $- 5 \times$ distance to metro, he does not directly write that down.

The human simply gives broad outlines, and the learning algorithm uses past data of several houses and prices and decides the right model. It builds the right model with a guideline from a human and data of several houses and it constructs the model from such data. So, now, this particular block diagram should give an idea and ground several ideas that several abstract ideas that we expressed when we add tasks into machine learning terminology.

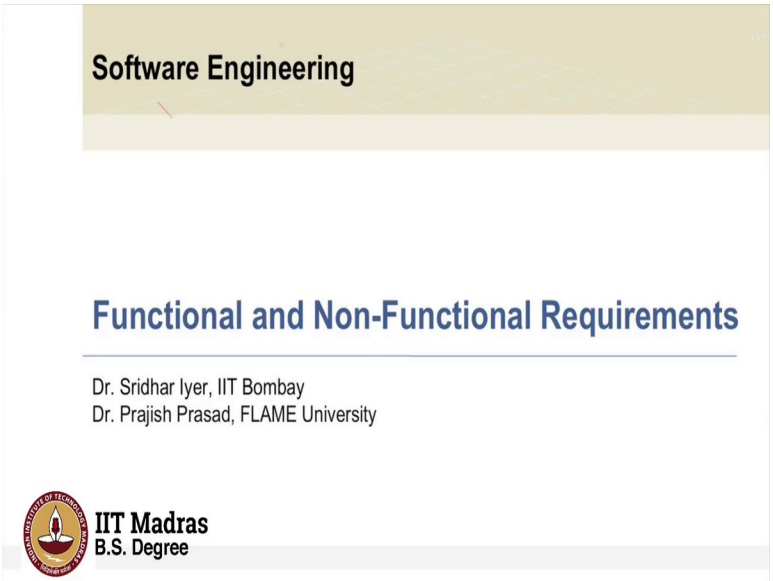
(Refer Slide Time: 19:04)

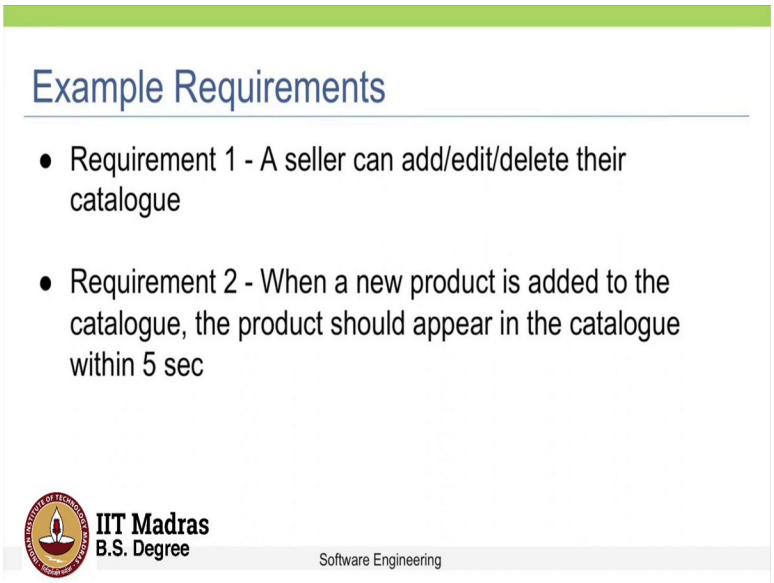


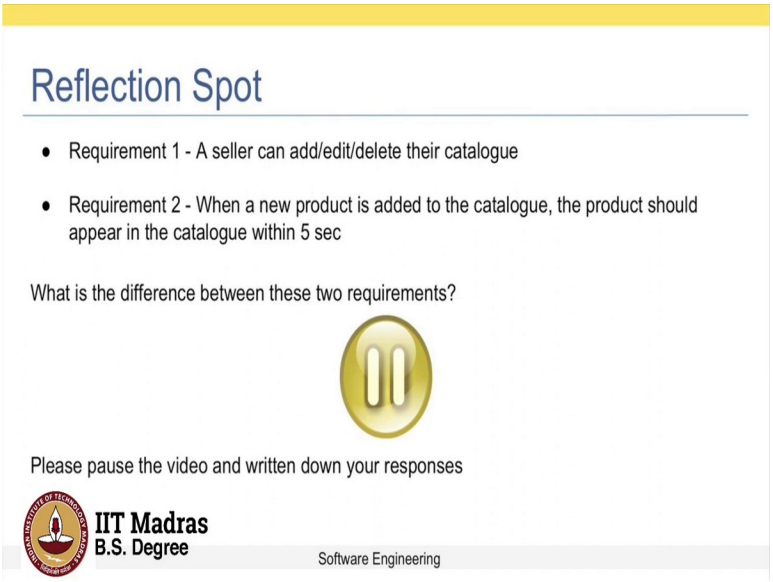
With that, I think we can wrap this part of the lecture, and the next part we will give concrete examples on more details of supervised learning tasks and unsupervised learning tasks.

Software Engineering
Professor Sridhar Iyer
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Professor Prajish Prasad
Department of Computer Science
FLAME University
Functional and Non-functional Requirements

(Refer Slide Time: 0:20)

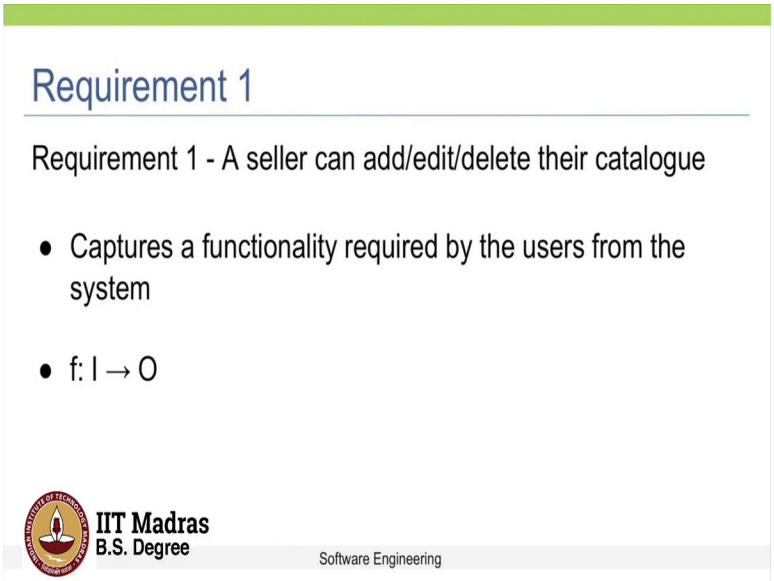


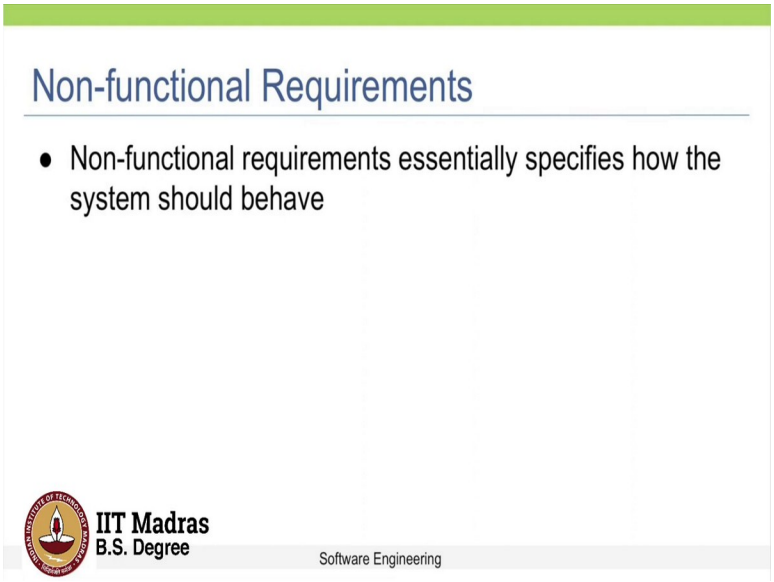


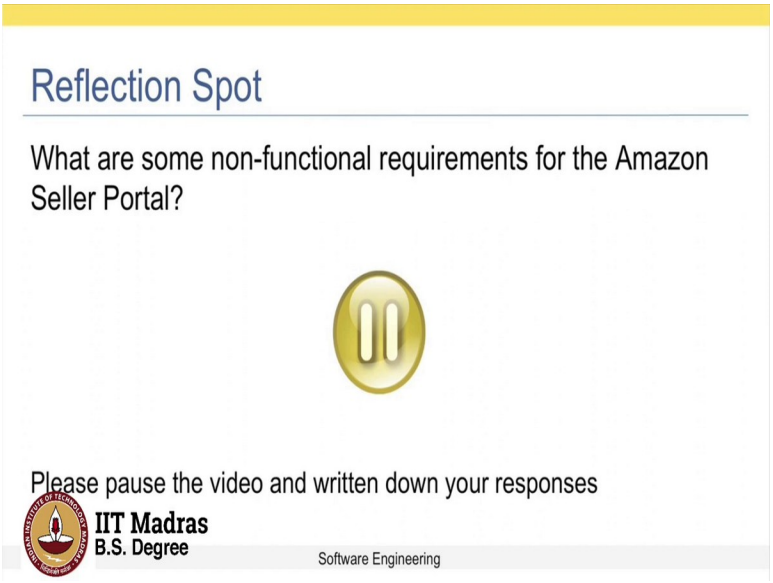


Professor Sridhar Iyer: In the previous video, we looked at how we can identify requirements using several techniques? Such as interviews, looking at the documentation, questionnaires, and so on. You might have noticed that not all requirements have the same characteristic, or similar type of characteristics. For example, consider these two requirements. A seller can add or delete items from their catalogue, this is one requirement. Second one is, when a new product is added to the catalogue, it must show up on the user's interface within 5 seconds. Can you say what is the difference between these two requirements? What is a nature of these two requirements? How was that different? Pause here for a moment, think about it, and then proceed.

(Refer Slide Time: 1:28)







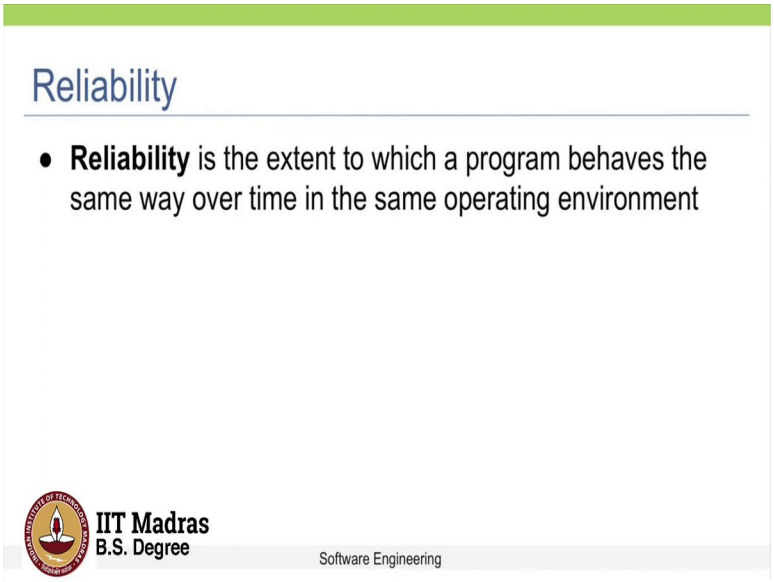
Professor Prajish Prasad: So, let us look at the first requirement. So, requirements 1 captures the functionality required by the users from the system. So, the user performs some actions, give some inputs, and based on that input, the system provides an appropriate output. So here, you can see that the user can maybe add, edit, or delete some things from their catalogue, and the output will be a result of these actions. So, these types of requirements are known as functional requirements. So, functional requirements can be considered similar to a mathematical function, where the function transforms an element in the input domain I to a value in the output O. So, the functional requirements of the system should clearly describe each functionality that the system would support along with the corresponding input and the output dataset.

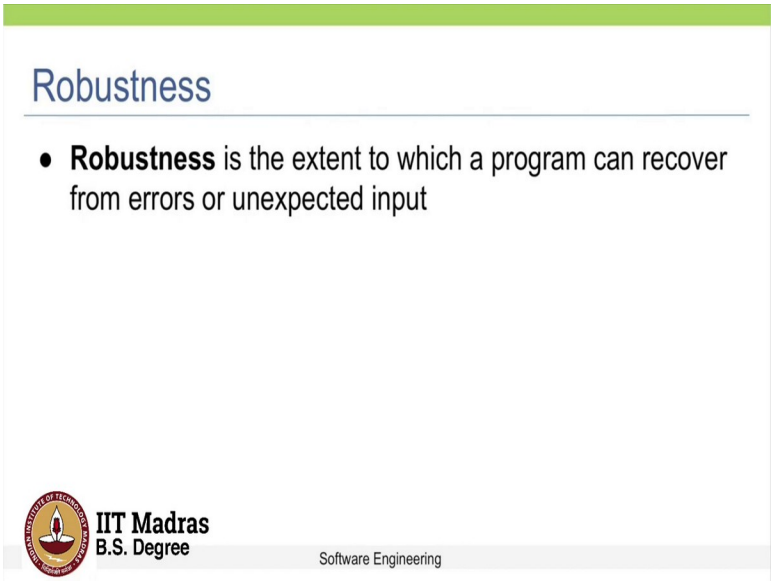
Professor Sridhar Iyer: So, coming back to the requirements, we have saw that requirement 1 of being able to add or delete a product from the catalogue is a functional requirement. The

requirement, second requirement that we talked about that is that the product should appear in the catalogue within 5 seconds. It is not a requirement that can be expressed as functions, that is as accepting some input and producing some output data. So, such requirements essentially specify how the system should behave, not what the system should do. So, such requirements are called non-functional requirements.

It is a little difficult to understand both these terms in the beginning, especially because we are used to the word non-functional interpreting the word non-functional in a different way. So again, let us take the example of the seller portal. We have talked about one non-functional requirement which was that the product update should appear within 5 seconds. What are some other non-functional requirements that you can think of for the system? Pause here, write down one or two non-functional requirements and proceed.

(Refer Slide Time: 4:11)



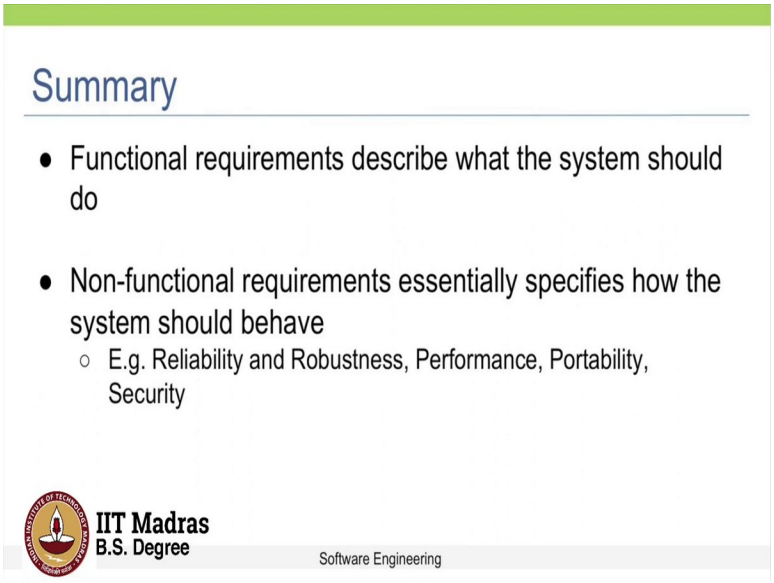


Professor Prajish Prasad: One type of non-functional requirement is reliability. So, a client would want the system to be reliable. So, reliability is the extent to which a program behaves the same way over time in the same operating environment. So, in the case of software, we need to ensure that the software does not crash often, otherwise users will get frustrated. For example, in the seller portal reliability is even more important. We must ensure that the portal handles all operations, especially when the inventory changes based on the buying and selling from customers.

Professor Sridhar Iyer: Now, in addition to reliability, we also want the system to be able to handle unexpected situations. What does it mean? For example, a user may just give some garbage as input, what is the system do now? So, such features, or non-functional requirements come under the category called robustness. So, robustness is the extent to which a program can recover from errors and unexpected input. For example, in the seller portal, we have to ensure that the service handles large numbers and high traffic.

When we specify requirements related to reliability and robustness, we are in a way ensuring that the system behaves the way we intend it to behave. And we are not exactly specifying what we want the system to do? Or we are not checking the functional requirements. So, that is the key between functional and non-functional requirements.

(Refer Slide Time: 6:10)



Professor Prajish Prasad: So, to summarize, in this video we looked at functional and non-functional requirements. We saw that functional requirements describe what the system should do? Whereas, we saw the non-functional requirements like reliability, robustness, they essentially specify how the system should behave? So, in addition to reliability and robustness, there are other non-functional requirements as well, such as performance, portability, security, interoperability, etc. So, we will not look into those in detail now, but it is important to consider the functional as well as non-functional requirements when we start building a software system.