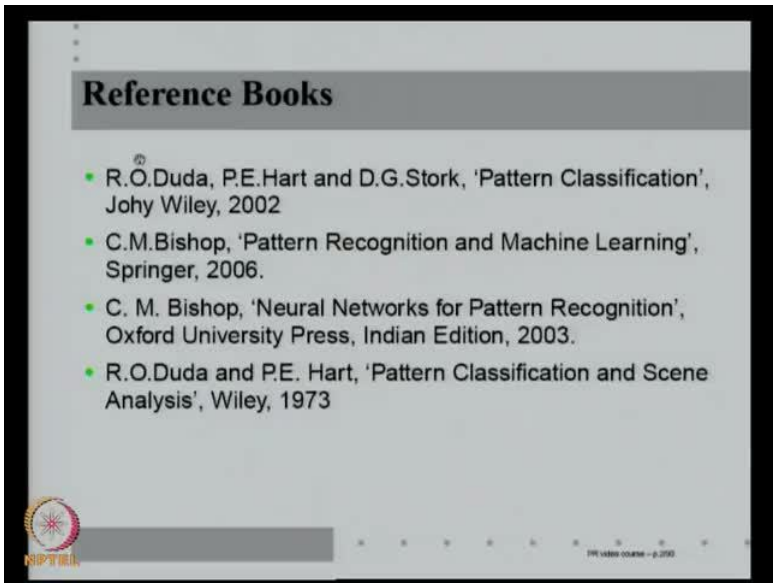
Pattern Recognition
Prof. P. S. Sastry
Department of Electronics and Communication Engineering
Indian Institute of Science, Bangalore

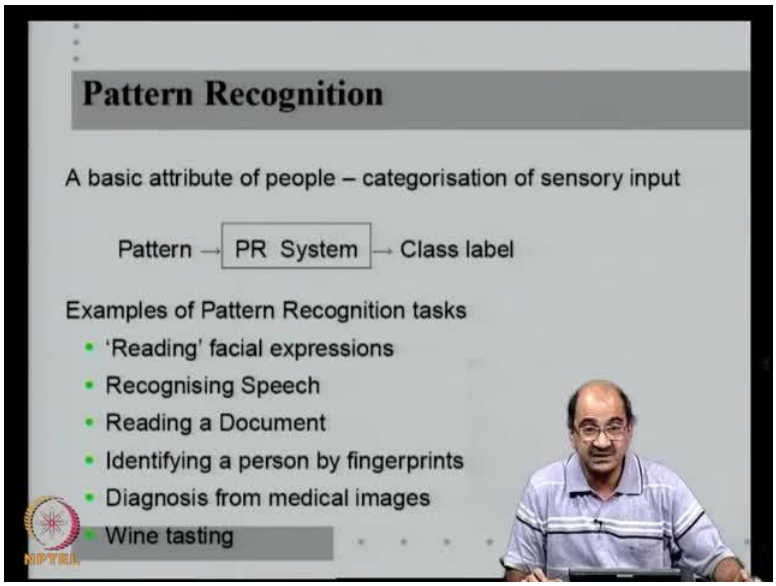
Lecture - 1
Introduction to Statistical Pattern Recognition

(Refer Slide Time: 00:23)



Welcome to this video course on pattern recognition. Today's lecture will be a general introduction to pattern recognition. These are the reference books for the course. The first two books that is, P.E.Hart and D.G.Stork and C.E.Bishop 2006 book are very good self content graduate level text books on pattern recognition.

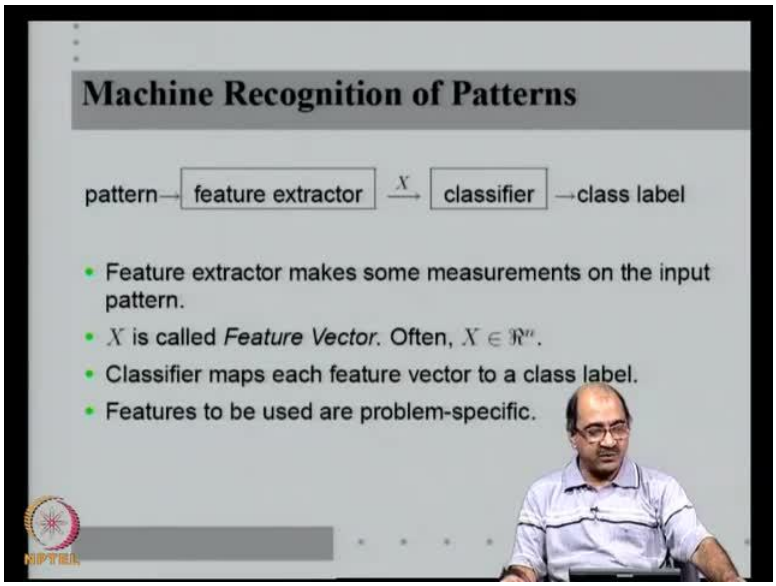
(Refer Slide Time: 00:58)



The book of Bishop is such a good book especially for neural networks. I have also included a very old reference of Duda and Hart 73 book, which actually defines the field and hence historically a very good book on pattern recognition. Even today, it is worth looking at. Now, let us start pattern recognition. Pattern recognition is recognising patterns. It is something that all of us do almost effortlessly. All human beings are capable of categorizing sensory input.

So, we can think of pattern recognition system as something that takes a pattern as an input and output is a class label. We want to categorise some standard pattern recognition tasks that all of us do. I have listed it here. Human babies when they are quite young can start reading their mother's facial expressions. It is a very complicated pattern recognition task. You need to know whether you are good or bad and whether somebody is smiling or angry.

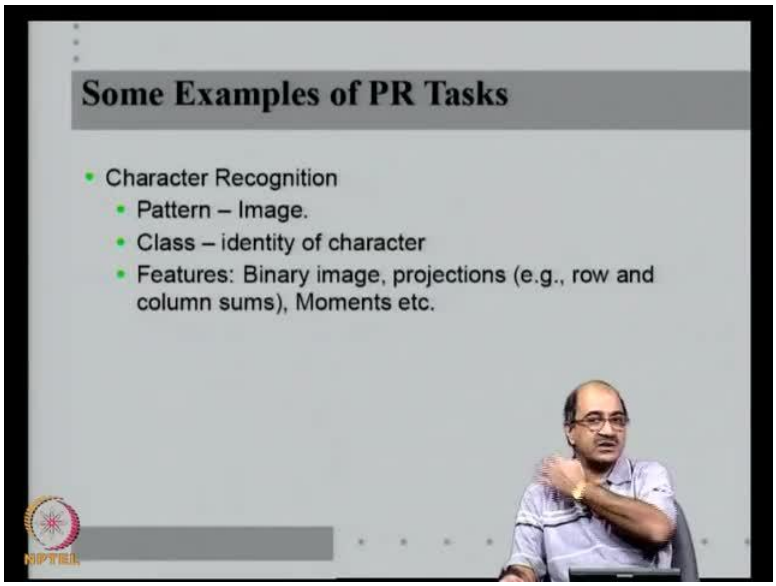
(Refer Slide Time: 02:40)



Another thing that all of us effortlessly do is speech. We are just born with the ability to learn to recognise speech. The next three things are what can be taught to us we do not effortlessly learn them, each with a little more difficulty. Reading is not like speaking. We have to be taught alphabets. The other two are professional tasks. They are finger print recognition and medical diagnosis. These are things that in each of the cases, we look at a pattern and diagnose it or categorise it into a category. I have also put the last 1. 1 does not know whether this is really true or not but, wine tasting is a highly priced pattern recognition ability that some humans can train themselves too.

Now, this course is about machine recognition of patterns. How do we program computers or some computer systems to do pattern recognition? The standard procedure followed for machine recognition of patterns is this two block, block diagram. You take a pattern input and then it goes through a feature extractor. The feature extractor makes some measurements on the input pattern. So the input pattern is converted to a vector of measurements, which we will denote by X. X is called the feature vector. For this course, most of the measurements yield real values; so X will be a vector in the n dimensional space, X belongs to R n.

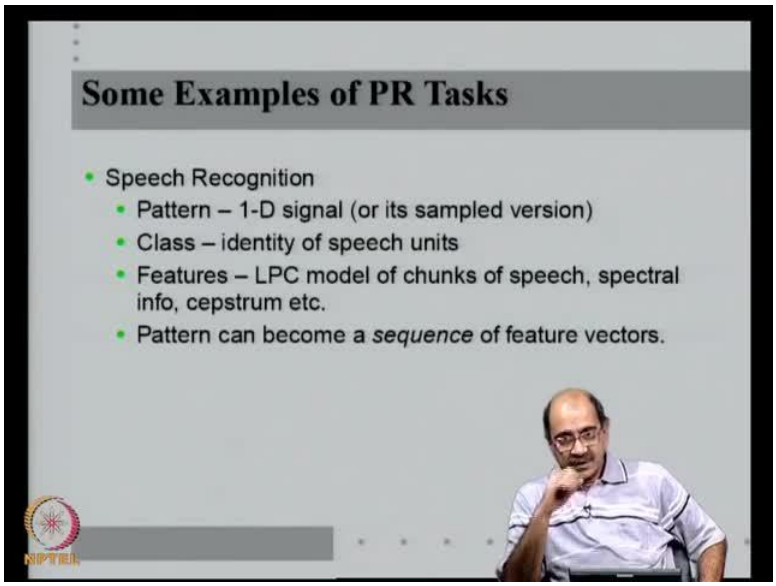
(Refer Slide Time: 03:38)



Then, the classifier block looks at such vectors and maps them to a class label. So, a classifier is a simpler function that maps each point in R n to 1 of finitely many values. In the general system, features are of course, very much problem specific. We will look at a few examples. Here are a few examples of pattern recognition task. The simplest one is character recognition. All of you would have seen OCR's. Almost every scanner today comes with an OCR system. The OCR has to take the image of a page, separate outlines, separates out words and characters and recognises which are the characters. So, just look how I do a character recognition.

So, the pattern is an image, somewhat isolated image of the character. The class is the identity of the character. So, if I am doing printed English recognition, then I have got 26 classes or 26 plus 26 classes if we count the capital and small letters. So, given the image of a character, the output is an identity of the character. There are many features people use. You can actually use a binary image itself as a feature vector. If you resize character say to 32 by 32, we can use a 1000 vector as a feature vector. Other thing that are often used in these tasks are row sums and column sums, projections of the image along various directions or expand some features in some basis functions. Those are the kind of features that 1 uses for character recognition.

(Refer Slide Time: 04:53)



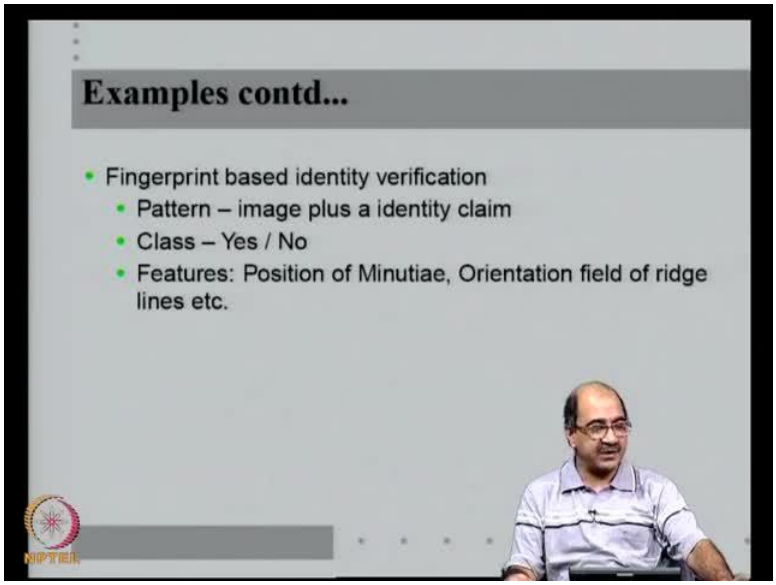
Let us see another example, speech recognition. Here the input is a 1dimensional signal of speech. The output of a microphone may be the continuous signal, but if you want to do it in a computer it is always this sample version. The class identity is the identity of the speech unit. Say for example, I want a voice activated phone. So I will just speak into the phone 1, 2, 5, 6, 7 and it should recognise that the first bit is 1 which is the first chunk of speech, the second chunk of speech is 5 and so on. The features used here are obviously much different from the character recognition problem. You have to somehow grab the spectral components.

So, you take the signals of the various kinds of features that people use and do a linear prediction coding. If you do not know what it is, it does not matter. Else, I take the spectrum of this speech signal and look at magnitudes of various frequency components or there is a special kind of spectrum that speech people calculate called the cestrum; so many such features are used.

There is another interesting thing about this pattern recognition task. Suppose, all of us know that the frequency component plays a major role of which speech unit it is, that speech signal is non-stationery. So, what is normally done is you take a chunk of speech and cut it into small pieces, may be each of about twenty milliseconds duration. Then, on each such piece, you grow a feature extraction so that within that duration this speech signal is stationery. Hence, we can find the spectrum given. What it means is that,

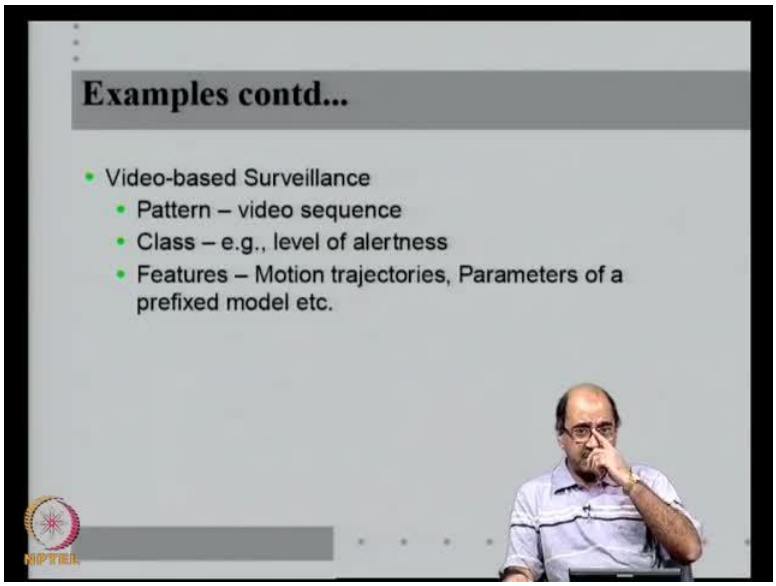
sometimes a pattern can become not one feature vector, but it is sequence of feature vectors. So, given a sequence of feature vectors you have to categorise them into a class.

(Refer Slide Time: 06:43)



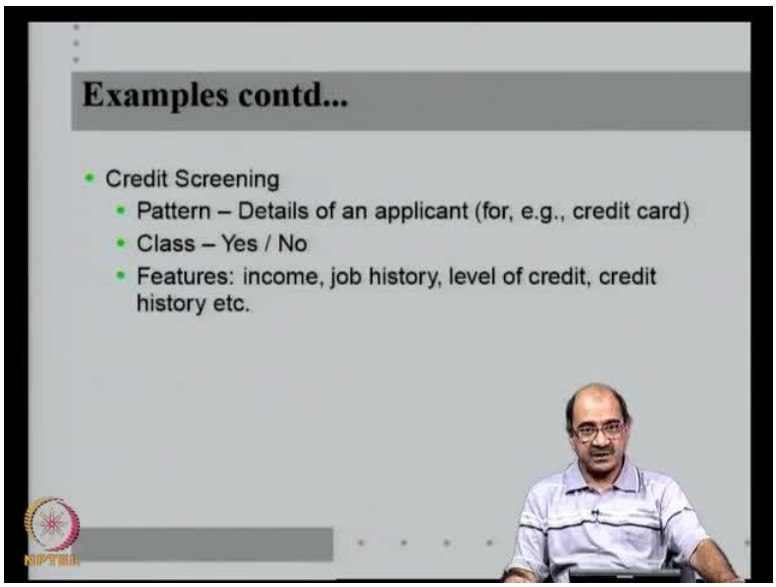
Another example, now a days there are bio-masses where, after typing in your login id instead of typing your password for authentication, you may just click on a mouse which has a finger print sensor on it. So, the computer will match your finger print and then decide whether you are authorized, that is whether you are the same user or not. So, here the pattern is an image, that is the finger print plus an identity claim by login id and the output is binary.

(Refer Slide Time: 07:28)



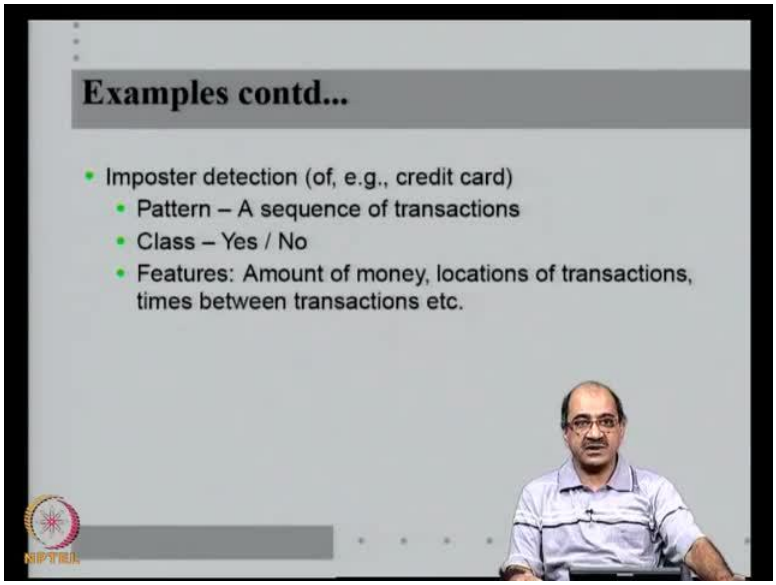
The output is a yes or no, the two class problem. There are various kinds of feature extracts of an image. These features are obviously different from what you extract from the image of a character for character recognition. They are called minutiae. The feature extractors are the position and orientation fields of minutiae. Another example video surveillance system; this is just what we see in many airports. So, what is the pattern here? It is a video sequence. A video camera scans the scene and the system alerts you on some security breach.

(Refer Slide Time: 08:29)



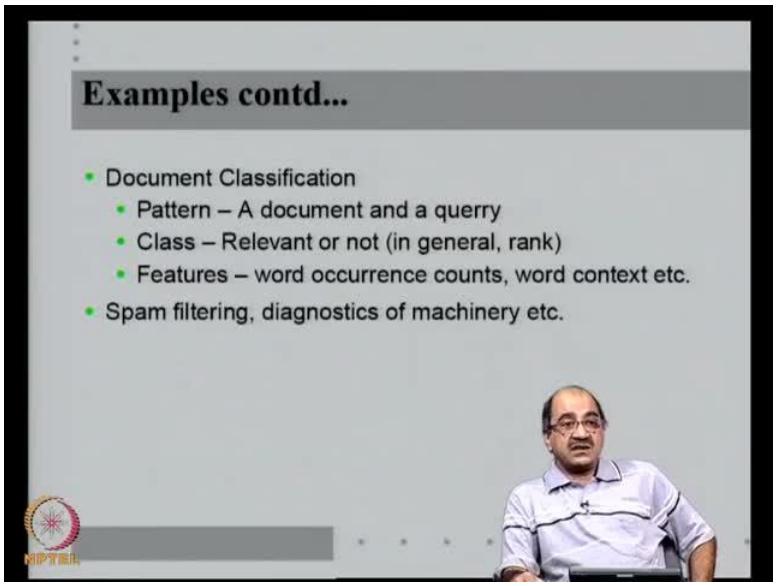
So, we can have different classes based on the level of alertness. Normally in the simplest of systems, you are looking for suspicious motions. So, you want to process the image, follow the motion trajectories, try and characterise object motion and then use those features to decide whether or not there is some suspicious motion. So, here the pattern could be a video sequence. We have seen different example the one example where pattern is an image. Another example pattern, is a 1 d signal. Here the pattern is a sequence of frames, that is actually a video sequence and accordingly the feature measurers are also different.

(Refer Slide Time: 08:51)



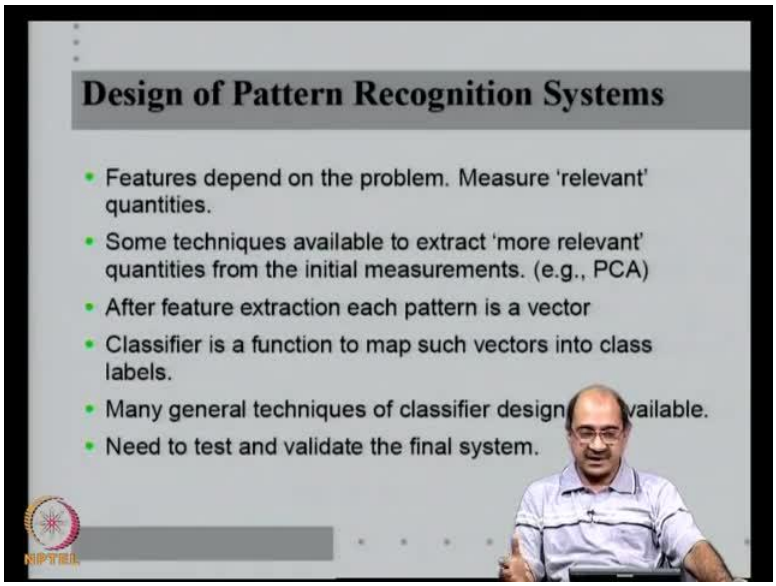
Many such examples, you can give. Many credit card companies these days use programs to do credit screening. So, here the input pattern is a detailed application. Then the features you measure are income, job history, level of credit and your old credit history. Once again, it is a binary classification whether you are credit worthy or not. For import detection, some credit card companies try to guess whether a card is being used illegally. So, once again it depends on features such as amount of money withdrawn, and location of transactions times between transactions. Given, the previous history of the card holder, anything that looks anomalous can be flagged and once again it is a binary classification task.

(Refer Slide Time: 09:18)



Another thing that all of us are familiar with, is searching on the Google. So, this can also be thought of as a pattern recognition task. So, it is slightly different. You can think of the pattern as a document and a query. For each given document and query, you want to also say whether the query is relevant or not because in general we have many documents and a query and we want to rank them. However, it is still a pattern recognition task where the input pattern is a document and a query and the output is relevant or not relevant. The standard features used are word occurrence counts, word context, which is a bigram of couple of phrases which indicate how often they occur in a document and so on and so forth.

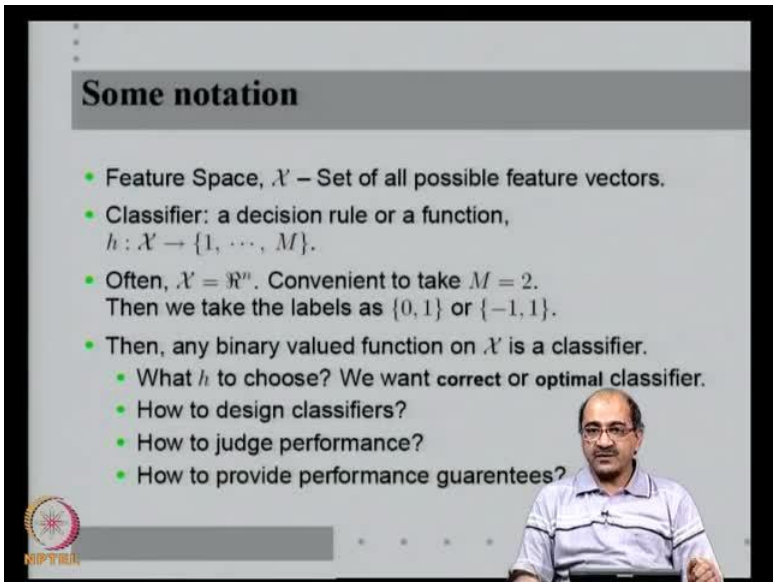
(Refer Slide Time: 10:02)



So, what we have seen is a whole set of examples of pattern recognition. So, one thing that comes out clear from these examples is that the features depend on the problem. So, depending on the problem, if it is a speech you may just use spectral components, if it is a video surveillance problem you have to characterise motion trajectories and so on. So, the features that you want to measure obviously depend on the problem. The only generic thing that we can say is sometimes having measured some features. We can transform them to so that they become more relevant for our classification task, other techniques such as PCA, which stands for Principle Complain Analysis, which we will see later on in the course.

So, there are some general techniques available to extract more relevant features, but in general a pattern recognition task feature extraction is very much problem dependant. It often depends on what all you can measure and what all you can think of measuring. After feature extraction every pattern is a vector and classifier is a function that maps such vectors to class labels. So, what we consider is that there are many general techniques for a classified design, where the classifier is used as mapping vectors into class labels and that is what this gist of this course is about. We will consider various methods of designing classifiers. From now on for us, a pattern is synonymous with a feature vector.

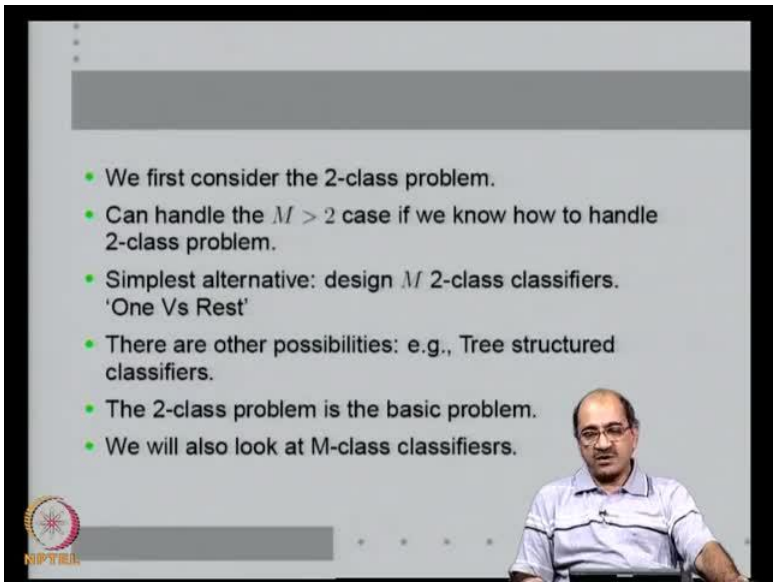
(Refer Slide Time: 11:27)



So, here is a notation. Let script X denote the feature space. It is the space of all possible feature vectors that that the classifier will ever encounter. A classifier is a function sometimes also called as a decision rule which maps the feature space to the set 1 to M, if we think there are M classes. So, what characterises pattern recognition is that either the output is finitely many or it is a categorization task. So, given some continuous input, I am categorizing the input into finitely many. I look at, when I am teaching my son the concept of a dog, I show him the images of so many real dogs which are also images for him. He has to somehow see what is common to all of them so that he can characterise a new image as a dog or not a dog.

So, a classifier maps this continuous feature space into 1 of finitely many class labels. For us X is n dimensional vector space. If you think there are n measurements, it is convenient often to consider the case M is equal to 2 which is called the 2 class problem. When we consider 2 class problem, we take the labels to be as a 0, 1 or minus 1 plus 1. If you do that, any binary valued function on the feature space is a classifier. It is by definition a classifier, but not all classifiers are equal for a given problem. We want that function which does well. So, what h to choose is the first question. Then, we say we want a correct or optimal classifier. Then I ask, do I define what is correct or optimal? What kinds of criteria can be used for optimality?

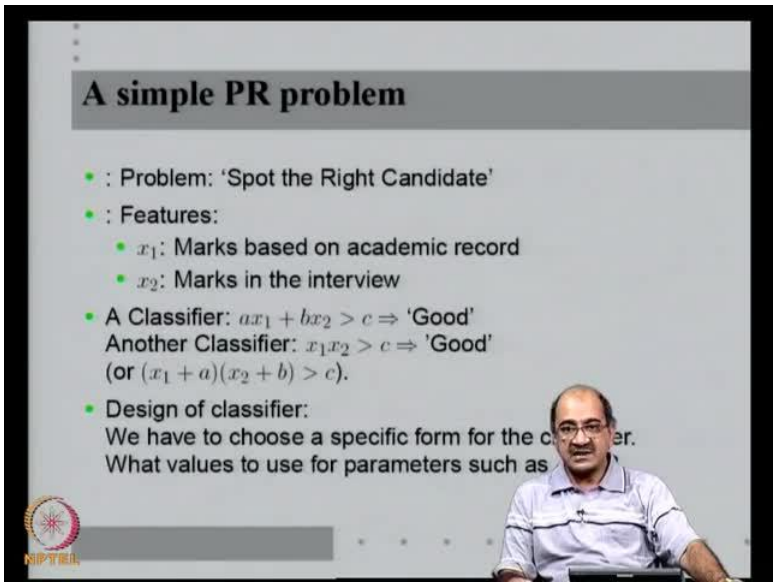
(Refer Slide Time: 13:24)



Given a particular criterion, how does one design a classifier? If I design a classifier, how do I judge its performance? How do I test its performance? Can I theoretically give any performance guarantees because of the particular method I follow for design? These are all the issues that we would be dealing within this course. For most of the course, whenever we consider a technique, we will always first consider the 2 class problem. 1 can handle the more M class problem, if we know how to handle the 2 class problem. For example, at this stage I can tell you 1 simple method. If we can design M 2 class classifiers, then we can solve an M class problem.

For example, I want to solve the 26 class problem of recognising English alphabets. I can design a classifier what says a or another classifier that says b or another classifier that says c or any other alphabet. Then, I give the pattern to all of them. When only 1 of them says yes, the other says no, then I know which character it is. Such, a method of designing M class classifiers using a number of 2 class classifiers is called 1 versus rest approach. This is very often used and in many cases it is successful. There is always a problem with this, suppose my a versus not a classifier also says yes, my b versus not b classifier also says yes, then I do not know whether to call it a or b.

(Refer Slide Time: 14:58)



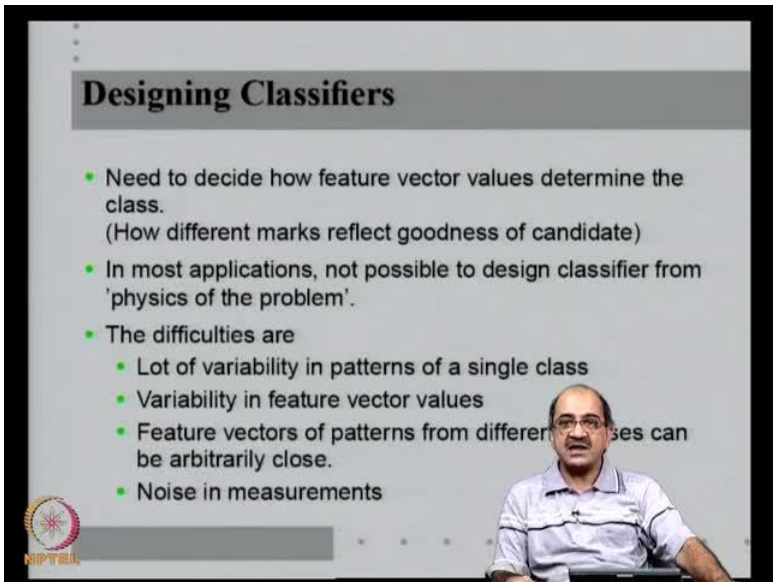
So, the 1 versus rest classifiers sometimes gives you ambiguous decisions. 1 needs to know how to handle it, but this is at least 1. In principle method of learning M class classifiers from 2 class 1's, there are other possibilities which are called as tree structured classifiers. In any case, the 2 class problem gives the most basic problem that is, what we will always first start with? We will of course, also look at the M class classifiers. So, let us consider a simple 2 class problem. I call the problem spot the right candidate. Many of you would have gone for some interviews or the other, and at the simplest level the interview panel has to make the decision whether candidate is good or not based on at least 2 features.

One feature sums of it is the academic record of the student. Let us say, for a student coming for a first job, the first feature is the marks obtained in his degree and the second feature is marks in an interview. So, the candidate is interviewed and the interview panel also give some marks. Given these 2 numbers, 1 has to make a decision whether the candidate is good or not good. So, for example, I can have a classifier which says for some a x 1 plus b x 2 greater than c. This means that the candidate is good. So, I am saying that if some linear combination of the marks is greater than some threshold, then it is good. So, the question is, what is the way I want to give different marks?

I know this is not the only classifier. For example, in principle based on the values of a, b and c, it is possible that even if x 1 is 0 and x 2 is sufficiently high, a candidate may still

be good. Whether we want that or not is up to us. If you do not want us, may be then we can take a hyperbolic classifier which says if product of x 1 and x 2 is greater than something, then the candidate is good. So, even a simple problem design of classifier involves what structure of classifier you want to choose. So, you choose a specific form for the classifier, we have to also choose what parameters to use and what values for a, b and c we will use.

(Refer Slide Time: 16:39)



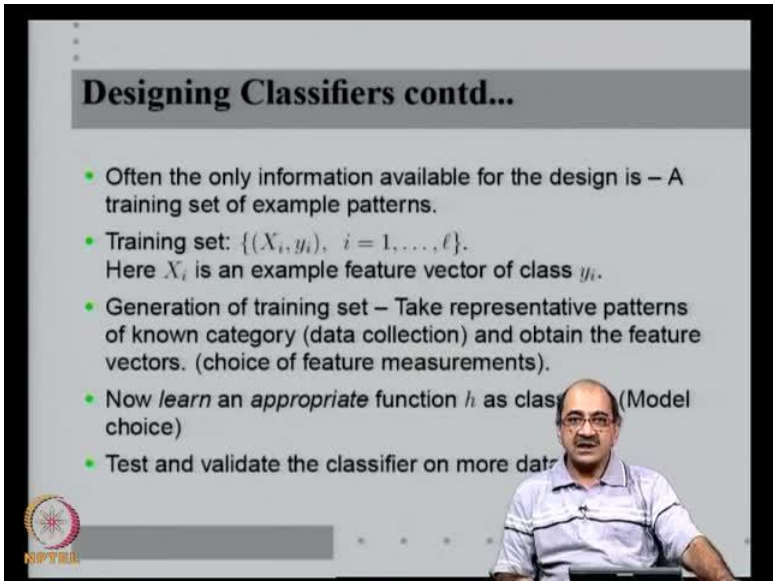
This is what constitutes the design for the classifier. Now, how do we design? We need to decide how the feature vector values determine the class. In our simple problem, we need to know how different marks reflect goodness of a candidate, at 60 percent in his interview and 80 percent in his university marks. Is it better that he gets 70 percent in interview and 70 percent in university marks? I do not know. So, those are the kind of issues that 1 has to determine. 1 has to determine the problem in most situations where pattern recognition techniques are used is not possible to derive this for many of the situation. For example, in this case there is no theory of how marks are given, so that using the physics of the problem I can derive what is the best linear combination.

Why is that so? It is because there are lot of variability in patterns of a single class. In my simple problem, good candidates who sometimes may do well in academic record, may not do well in interview. Some people are good at written exams. Some people are good at oral exams. So, there is a lot of variability in the patterns of a single class. If you think

of character recognition and think of all the ways different people write their a's and b's or all the different calligraphic a's and b's that you can recognise, there is a lot of variability in the shape of the characters. Still, all of us know what is a, what is a b. So, there is a lot of variability in patterns which of course, translate to variability in feature vector values.

So, if I measure the feature vector for any given set of features that I choose for many different a's and b's. For example, I will get different numbers. So, there will be a lot of variability in feature vector values of this of pattern belonging to the same class. Also, the pattern belonging to different class can have arbitrarily close feature vector values because my feature measurements are arbitrary. It is not determined based on a measurement that is known to be discriminatory. I just measure what I can. For example, in the character images we just measure row sums, column sums, sum projection and sum movements. So, none of them have anything to do with a's and b's. So, the feature vectors of patterns of different classes can be arbitrarily close and of course, there is always noise in measurement.

(Refer Slide Time: 19:04)



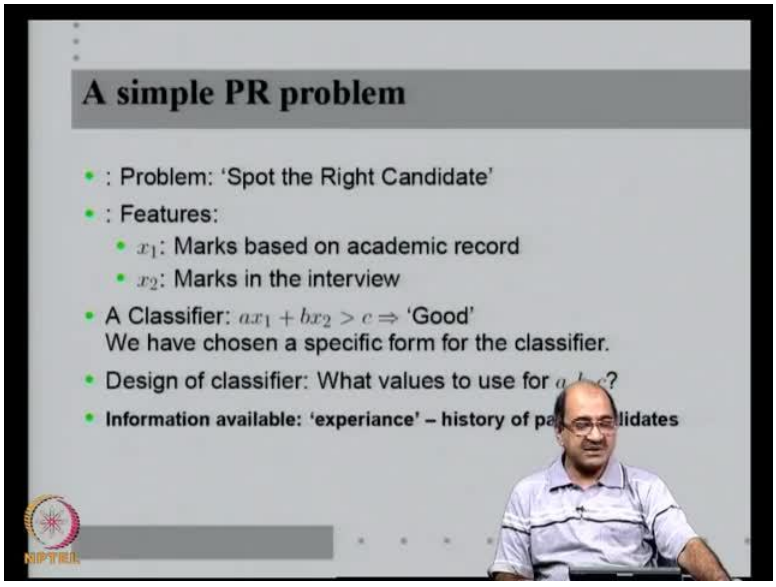
So, given this much variability it is not possible to design the classifier from any first principle. Then what do we have? So, the only information available for designing the classifier is what is called a training set of examples. I am given a training set X i, y i. I am going from 1 to l, where each X i is an example feature vector. I am told what is the

class of X i and y i. You can think of this as the way we learnt our first alphabet. The teacher writes on the board or may be shows a card of shape of a and say this is a. She writes a b and say this is b. So, if she writes a many times and b many times, then we learn to recognise those shapes.

So, if you are given a training set of examples and just using that you have to design a classifier, how do I generate the training set of course? Here, in this training course I do not have any classifier. If I have a god given classifier, which can classify feature vectors correctly, then I do not need to design any other feature pattern classifier. So, even if I measure features on some random patterns, I do not know how to get away.

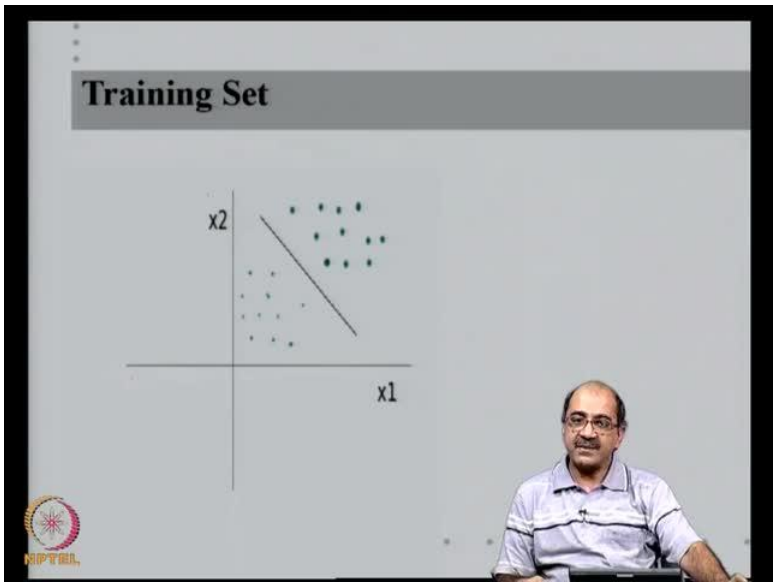
So, the way to generate the training set is to take representative patterns from the category. So, for example, if I want the training set for character recognition, I will go to 20 people and give them ruled sheets and ask them to write many a's and many b's. Then, I take images of each of them and do my feature measurement. So, I get feature vectors and the class of a feature vector is known because it has come from the person who wrote the shapes as a.

(Refer Slide Time: 20:51)



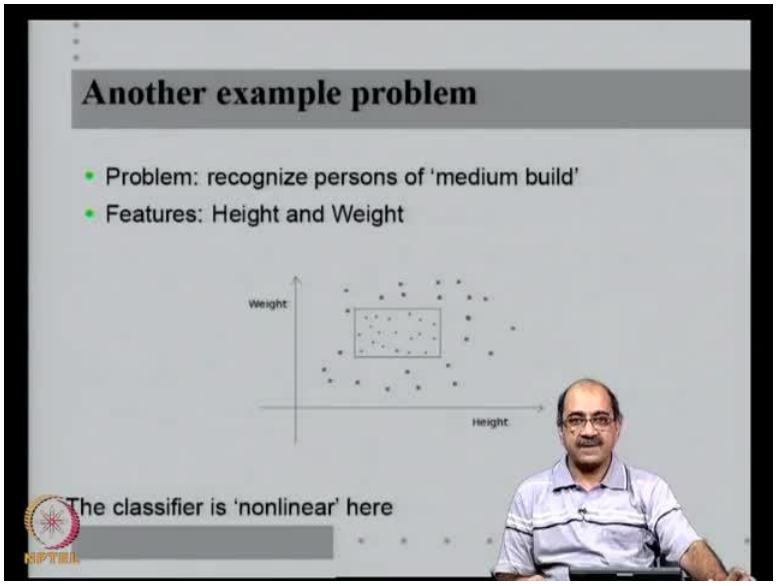
So, what you say a is a and what you say b is b, that is how I know the class labels. So, in general I generate the training set by taking representative samples from the respective categories. Now, we have to learn the appropriate classifier by using the training set. So, coming back to our simple problem, spot the right candidate.

(Refer Slide Time: 21:14)



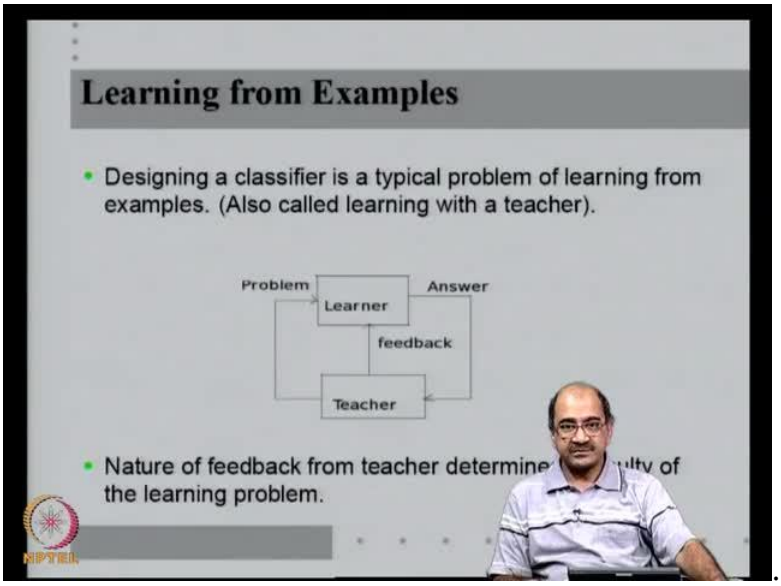
Let us say we chose an a a linear combination classifier. We chose the classifier to be a x 1 plus b x 2 is greater than c. We have chosen a specific form for the classifier. So, what is the information available for us? The experience of the people making the decision have seen history of candidates. So, for example, the information can be presented on the x 1, x 2 space, which is the feature space. If I put these points, then each person will become a point there.

(Refer Slide Time: 21:44)



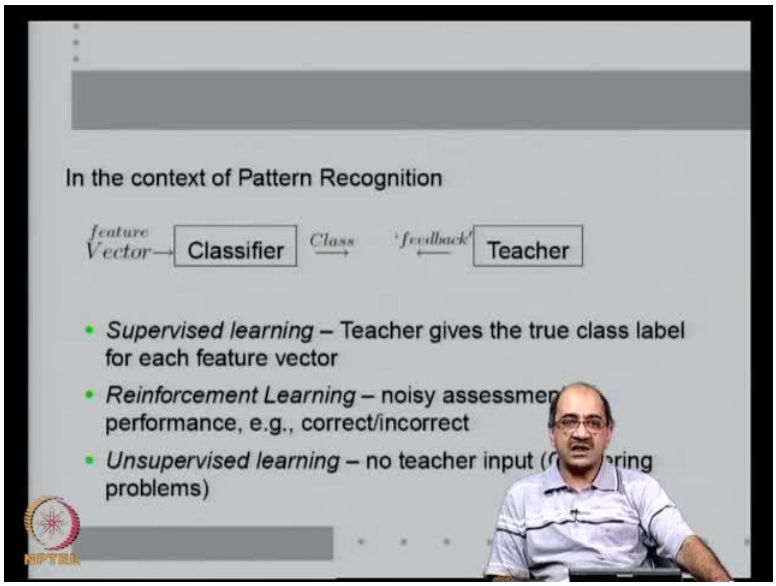
I know who performed well afterwards and who performed poorly afterwards. So, for example, let us say that those big are those of good candidates and the small dots are those of bad candidates. So, given this experience, I want to know which line is the best line to separate them? I want to know the 1 output line that separates them? So, in general that is the way I will learn a classifier from a training set. So, here is another example. Suppose, I want to teach somebody the concept of medium built, of course, we know that medium built means a range in height and weight, but the person learning it does not know that concept. So, we just show him lot of examples of people of medium built or non medium built.

(Refer Slide Time: 22:27)



So, for example, once again there are 2 sizes of dots. We want to separate 1 size from the other, but what separates now is not a line, but a rectangular enclosing box. So, the classifier is non-linear here. So, the boundary between classes can be captured by a line or a hyper plane in general or it may not be captured by a line, but may need more complicated surface like a rectangle. So, you can think of designing a classifier either by learning from an example sometimes also called learning with a teacher. So, you can think of it like this. There is a teacher, who is trying to teach much like the way, we learnt our alphabets.

(Refer Slide Time: 23:29)

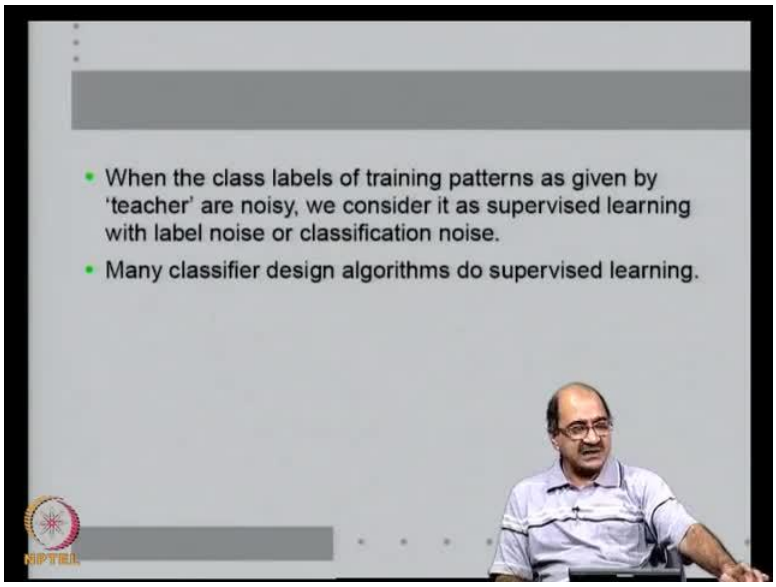


So, the teacher shows me a card on which something is written. Then, the learners give some answer. I shout out and then I get a feedback whether what I said is right or not. You should have said b or you should have said c. So, this setting about a problem and feedback is what the training examples are all about. Now in general, of course, pattern recognition is only 1 example of learning 1 instance of learning from examples. In general it can be used in many different situations. Learning to ride a bicycle is also something that we do now. All these can be captured into this block diagram by saying that the nature of the teacher's feedback can be different once again in the context of pattern recognition. So, what we are saying is that you set a feature vector at the problem. A classifier output is a class. Then, I get a feedback from the teacher which tells me the right class that is the X i, y i in the training set.

Now, this kind of learning is called supervised learning. Here the teacher gives the true class label for each feature vector. So, this teacher is very nice, very informative like my teacher in teaching me the characters. This tells me tells me what I should have said for that character. There are other kinds of learning situations, which are also important. Those are not considered in this course. They are called reinforcement learning, where the teacher could only tell you whether you are doing poorly or well. So, that is called reinforcement learning.

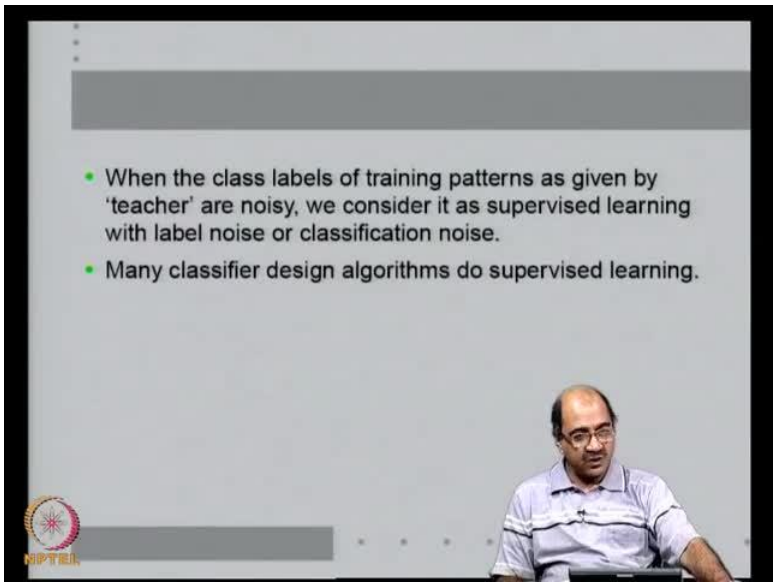
For example, learning to ride a bicycle is not like learning an alphabet. There also we need some instruction, but that is mostly done by day. The feedback I get, that is how long I have been able to balance, how soon I fell, when I fell and some more. Using this we learn to learn the right coordination.

(Refer Slide Time: 24:54)



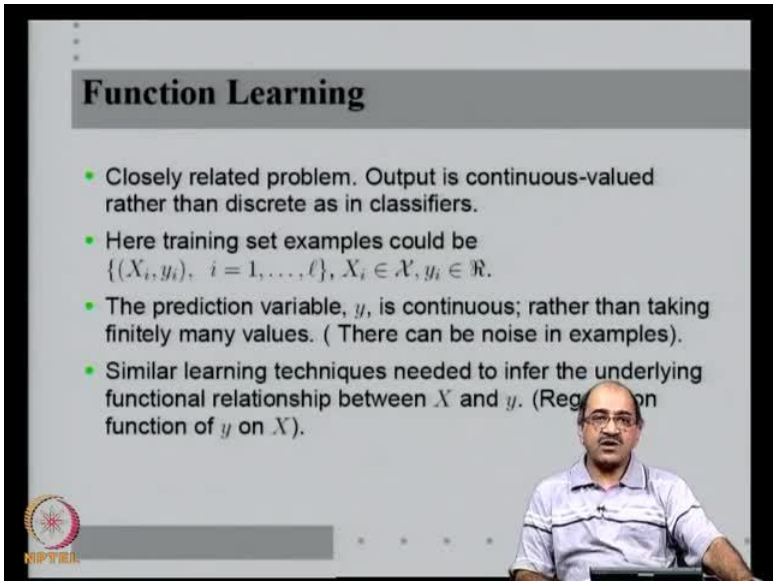
So, in reinforcement learning the teacher just provides a noisy assessment of performance. Also, there are also unsupervised learning problems where there is no teacher at all. Just by looking at all the examples and this structure inherent to them, I want to somehow categorize; sometimes this is called clustering problem. We will not consider neither reinforcement learning nor unsupervised learning in this course. So, we will essentially be considering supervised learning, where the teacher gives me the true class label. So, I have my X i, y i, but when I say 2 class labels, there is inevitable noise in any training set.

(Refer Slide Time: 24:54)



So, we will always budget in our design for some of the labels in the training patterns given by teacher which may not be correct. We will still consider it as supervised learning with noise. We will like classifiers that can withstand such noise, that is that are robust to such noise. Almost all classifier learning algorithms do supervised learning.

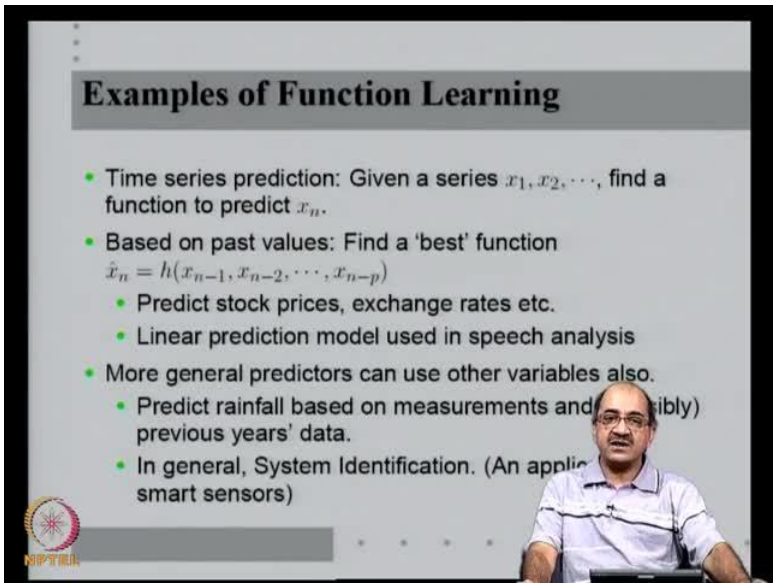
(Refer Slide Time: 25:38)



So, we will be considering various supervised learning algorithms for learning classifiers. Before we complete our generic introduction, there is another very closely related problem and this will also form a part of the course which is called function

learning. It is closely related to pattern recognition. Here also, you have training examples, X i and y i. The only difference is y i instead of taking finitely many values in the classifier case, it takes finitely many values and the values are in the class label.

(Refer Slide Time: 26:34)

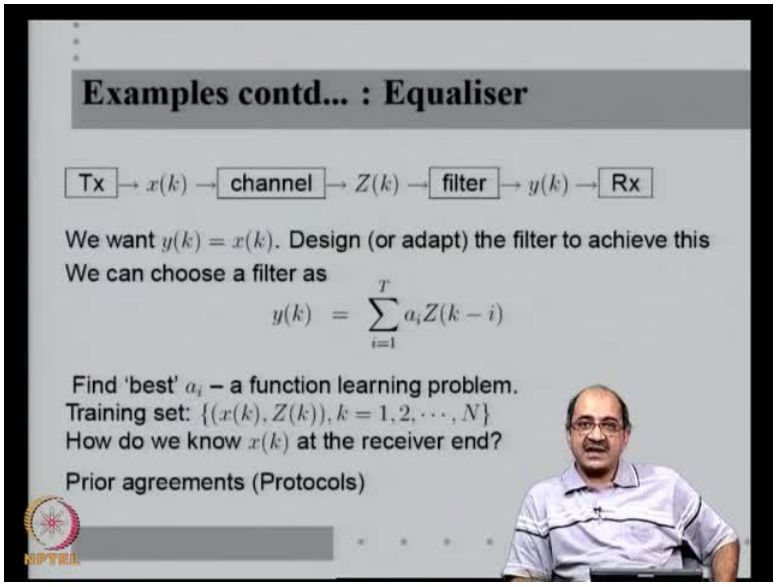


Here, y i take continuous values. So, here what do I call the prediction variable? Here, y is continuous. Rather than taking finitely many values, however I still need to know that given these examples, if you give me a new X what will be the corresponding y. So similar learning techniques are needed to infer the underlined functional relationship between X and y. So this is called function learning. Just, like we saw examples of pattern recognition, we can also look at examples of function learning. So the simplest example of function learning is time series prediction. Given a series of numbers x 1, x 2, up to x n or x n minus 1, I want to predict the next number x n.

What could it be? So, I want to find a best function which predicts x n. I have put a hat on top of x n. I will call x n hat is some function of x n minus 1, x n minus 2, up to x n minus p. So, I want to predict the next number as the function of the previous numbers. For example, I may want to predict stock prices, like what will be the stock price tomorrow based on the stock prices or the past 2 days? I may want to predict exchange rates if I am a banker. I may make lot of profit by buying and selling foreign exchange. So, I want to predict exchange rate fluctuations. Many of the problems on speech signal compression, you might have studied only linear prediction models for speech.

So, given some speech I may not want to transmit the whole of it, if I can predict some part of the speech based on the previous speech. Then, I can once again learn a function to predict. There are quite a few compression algorithms that use such prediction models. However, more general function learning problems can include prediction that not do not use only the previous values, but use extra variables also. I may want to predict rainfall this year based on rainfall of the previous year's data or previous few years data and also the mean measurements I made this year.

(Refer Slide Time: 28:18)



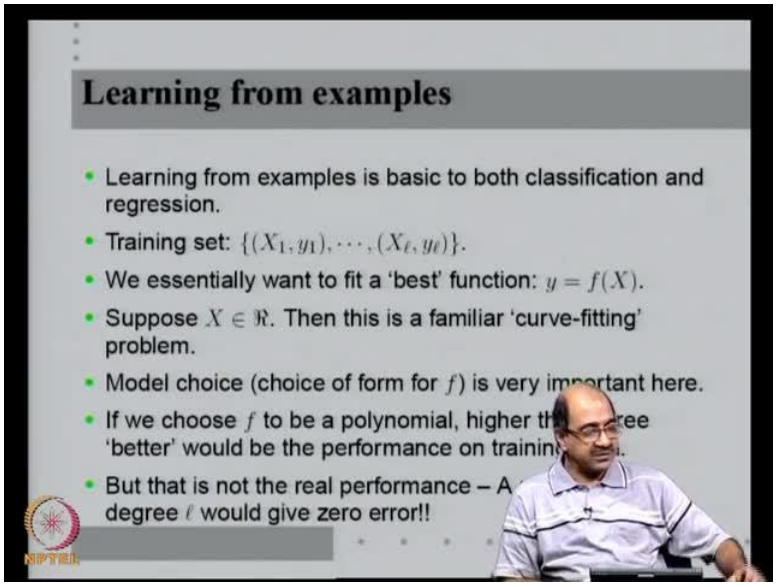
So, in general any system identification problem is also a function learning problem. We will also consider function learning problem which are often called the regression problems and classification regression which are often viewed together because similar kind of learning techniques are needed for each other. So, a simple example of a function learning task for those of you who studied communication is a transmitter. Likewise, in any communication system sends some signal through a channel, because the channel disturbs their signal in the transmitters, so put out X k. The channel puts out Z k.

Now, at the receiver end I want to design a black box which I call a filter which takes in input at Z k and gives me some y k. I want to do the filter design in such a way that y k is closed to x k. For example, I may decide to get my y k as some linear combination of the past values of Z. Very often that kind of filters work in this problem. But, what I am asking is what a i's will give me the best chance of y k being close to x k. So, if I know

some x k's that are transmitted, then I can measure the channel input Z k. So, then I pass it through this filter. So, the filter gives me some y k. I can ask, is y k close to x k or not?

So, my training examples here would be the Z k at the input, that is what takes the role of X i and y k at the input, that takes the role of y. So, I can get x k, Z k as the input, where x k is the true output. I should take that Z k is the input. So, I should have written at Z k, x k and this learning has to be done at the end of the receiver. So, the receiver has to know what was sent, but how does he know? If he already knows, he does not need the filter. Often what happens is that, we have some communication protocols whereby we know when a new session starts, so I use that information to measure Z k for those time instances. Then, I adopt my filtered coefficient a i so that the filter output is as close to **x** k as possible. This is a standard function learning task, which is often called as the equalizer.

(Refer Slide Time: 30:31)



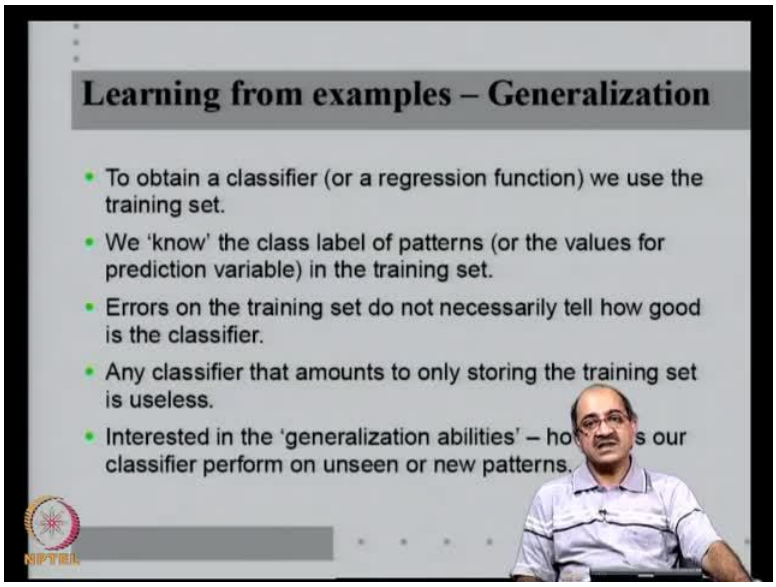
So, learning from examples by going back is basic to both classification and regression. In both cases, we have a training set X l, y l. There are some examples. So, x's are the feature vectors or input and y's are the class labels or the output. We essentially want to fit a function y is equal to f of X, the function is binary valued or finitely many valued. If it is a classifier it could be contrast valued. If it is a function learning problem in both cases, given some training set we want to fit a best function. Well, this is not an

unfamiliar problem for many of us. If X belong to R, this is the familiar curve fitting problem that all of us did in class 12, right?

Given some numbers, X 1 by 1, X 2 by 2, we want to fit a curve. We can fit a straight line or fit a quadratic. What do we do? We do least squares curve fitting. So, given X 1, y 1, X 2, y 2 etcetera, we say that y is equal to m X plus c. So, I will calculate at m, X i plus c and see the error from y. Based on that error I try to find the best m and c for the line. This is the curve fitting problem, but 1 point I want to emphasize here is that nobody tells me whether f should be linear, whether f should be quadratic, or whatever. So, the choice of this form for f which is called model choice is very important from the statistics point of view.

Just to understand what it means, suppose X is a single real number where X belongs to R, let us say we choose f to be a polynomial, then if I choose a higher degree polynomial I can get better fit. Using the training set I can fit it better. For example, if I have 20 points Xi, y i and if I choose the 28 degree polynomial then I can get perfect fit. I can get 0 error on the X i, y i. If I have 2 points, I can get 0 error with a straight line, but that is not for the fit we want hat we will be fitting all the noise. So, the question of what f to do is important and this is not determined by how well I do on the training data. I can always find f and do perfect on the training data, but they are not useful. So, what we want in learning from examples is generalization.

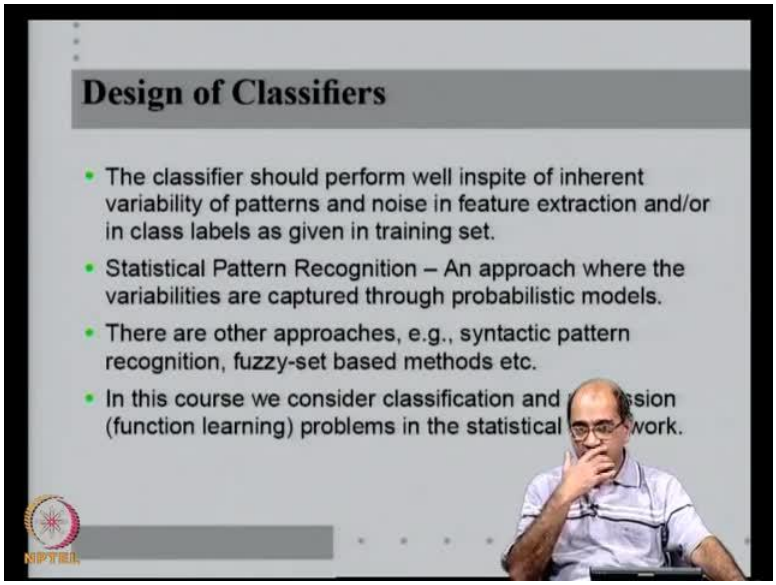
(Refer Slide Time: 32:50)



For example, when we are learning our characters from our teacher, we do not want photographic memory of what teacher wrote on the board. Somehow, looking at many a's we need to abstract out the concept of a. Now, we can recognise any kind of a anybody writes. So, to obtain a classifier or a regression function, we use the training set, but we know the class labels of the patterns in the training set. Hence, really others in the training set is not necessarily a good indicator of how good the classifier eventually is. For example, any classifier that amounts to only storing the training set is useless. We are interested in the generalization abilities of our classifier.

We ultimately need to be able to give some guarantees and how our classifier which is designed on some training set will do a new unseen pattern. The training set is reduced to design it, but just reducing the other to 0 on the training set is not necessarily good always. Of course, we will see when it is good, when it is not good and so on. This is part of what is called statistical learning theory. We will come to that later, but at this point of time we want to remember that what we are interested in whether or not the classifier that we have learnt from our training set generalizes well.

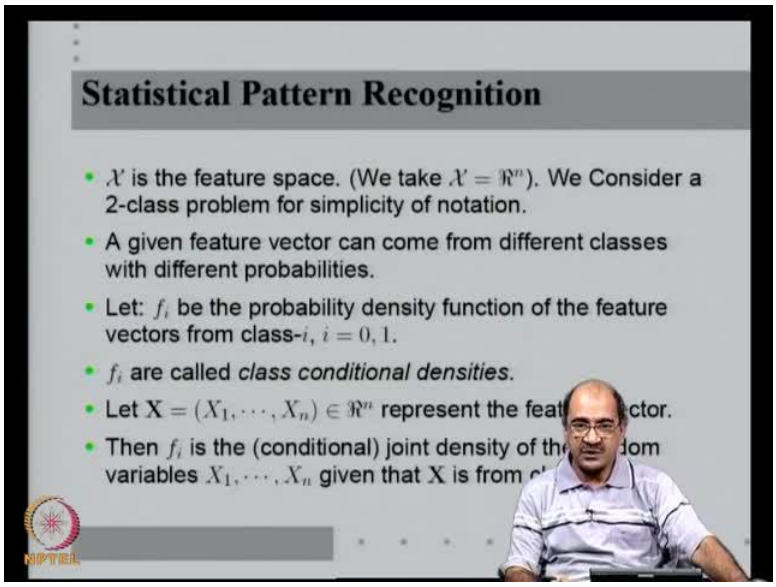
(Refer Slide Time: 34:18)



So, the classifier should perform well in spite of inherent variability of patterns and noise in the feature vector under the class labels. Ultimately it should generalize well. Now, statistical pattern recognition is what we considered in this course. It is an approach to classifier design or classifier learning where variability's in the patterns, variability's in

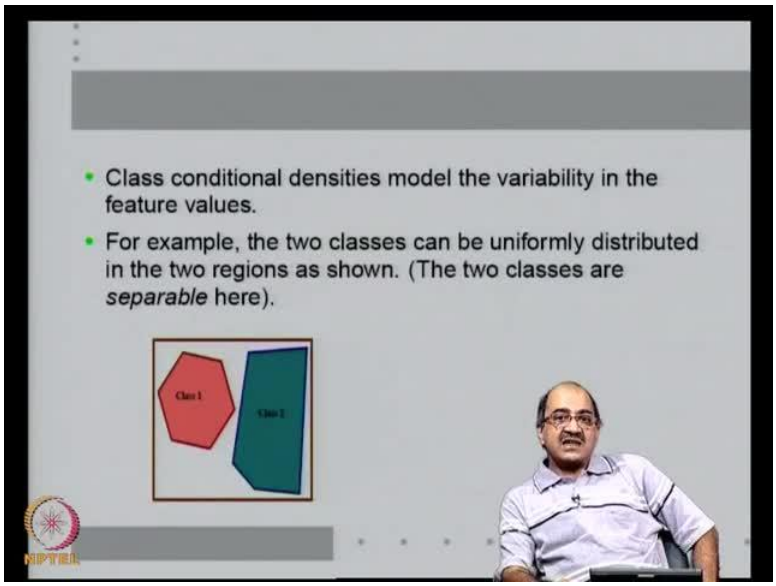
the feature vectors, what are the real difficulties in designing classifier. These variability's are captured through probabilistic models. Hence, we can think of a feature vector coming from 1 class as a random variable, which can take different values with different probability. So, the reason why measurements made on a's written by different people will give you different numbers is that the feature vectors of class a is actually a random variable.

(Refer Slide Time: 35:26)



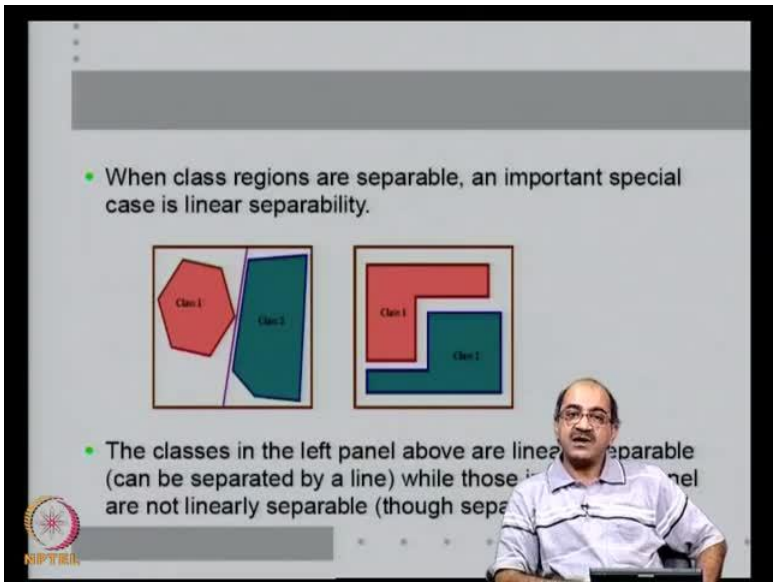
Hence, it has an associated density function, which tells me what all values it takes with the type probability. Statistical pattern recognition is not the only approach. There are other things, syntactic pattern recognitions so on. We will not be considering any of them in this course. We will be considering mainly statistical pattern recognition. In statistical pattern recognition the feature space is X. We consider two class problem. For our initial notation, a given feature vector can come from different classes with different probabilities. So, let us denote by f subscript i, the probability density function of feature vector that comes from class i. So, f 0 for class 1, f 1 for class 1 and f i for class i are called as class conditional densities.

(Refer Slide Time: 36:18)



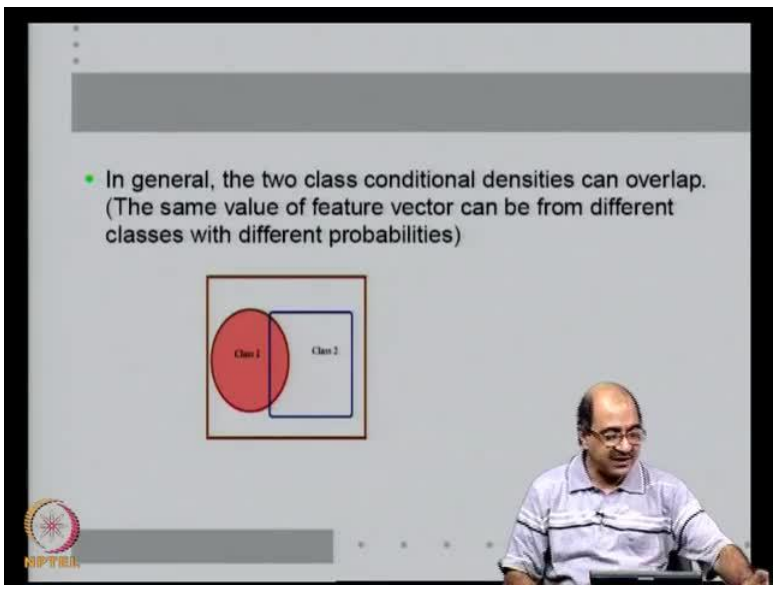
So, if I think X 1, X 2, X n are the components of an n dimensional feature vector, we think of them as random variables and then the conditional joint density of X 1 to X n conditioned on X coming from class 0 or class i is f i. So, f 0 is the conditional joint density of X 1 to X n conditioned on the feature vector X coming from class 0. So, class conditional densities model the variability in the feature values. For example, I can have 2 class problem where the 2 classes can be uniformly distributed in the two regions. By that what I mean is, all feature vectors coming from class 1 are uniformly distributed in the region written class 1. Similarly, for class 2. This is of course, a very nice pattern recognition problem. Such things do not come often. Because, here the 2 classes are separable a particular feature vector value comes from only from 1 class.

(Refer Slide Time: 37:16)



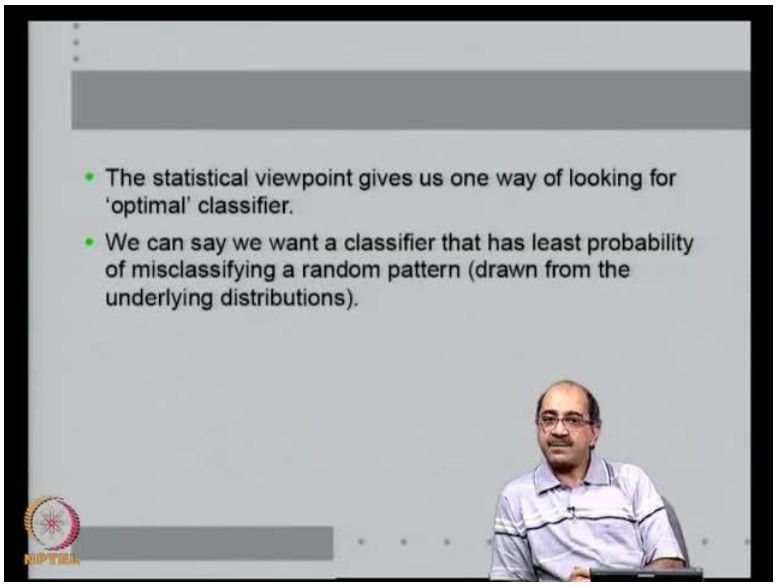
So, I just have to know the extent of values of the two classes to decide the classification. Even in separable classes ultimately, because the feature vectors are not two-dimensional. Hence, I cannot create such pictures all the time even when they are separable. I need to know what kind of surface will separate them. For us something that is very important is what is called linear separability. For example, in the left panel, the two classes are separable by a line, whereas on the right panel the two classes are not separable by a line. I cannot draw a line keeping all the class 1 patterns on side and class 2 patterns on the other side.

(Refer Slide Time: 37:48)



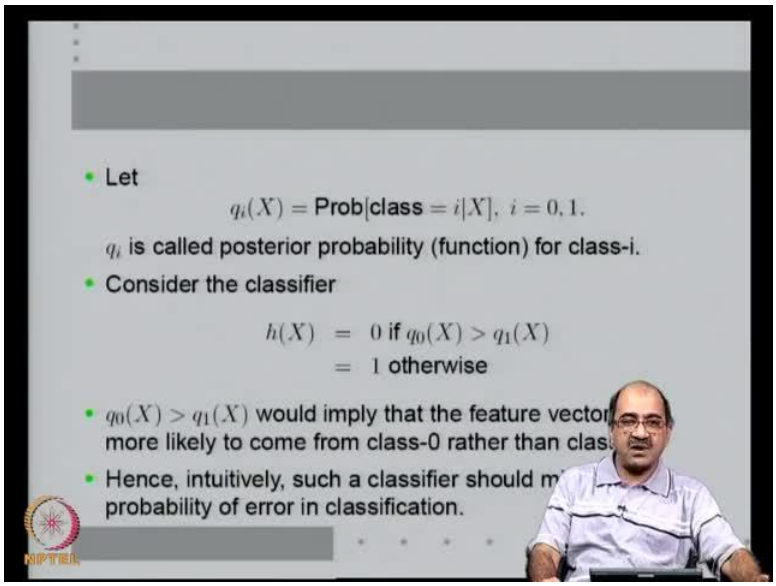
So, even when they are separable, classes may be linearly separable or may not be linearly separable. Of course, in general classes will not be separable. So the region over which class 1 feature vectors have non-zero probability and the region over which class 2 feature vectors have non 0 probability can overlap. Hence, we say that the class conditional densities overlap. Then of course, I cannot get a perfect classifier that makes 0 errors, but I may still want to have a classifier that is somehow good.

(Refer Slide Time: 38:09)



So, this statistical view point thinking that feature vectors are random variables have their density functions, gives us one way of looking for what can be an optimal classifier. It at least tells us one kind of optimality criteria.

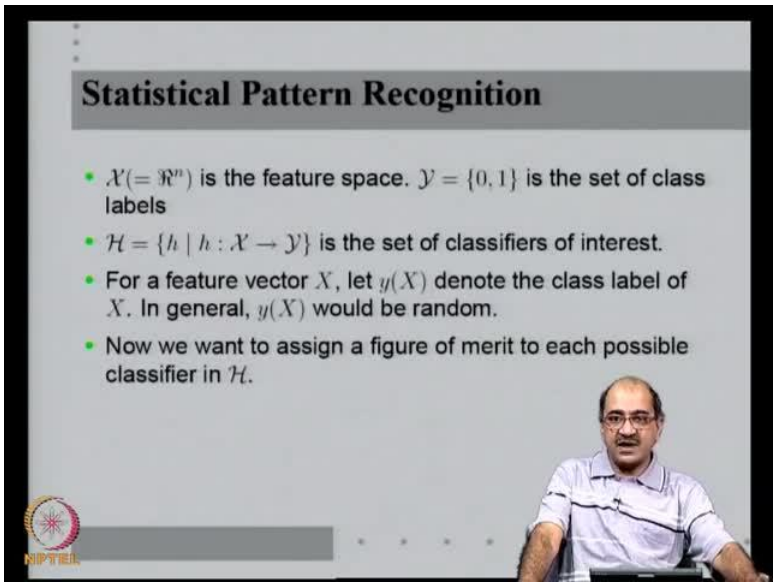
(Refer Slide Time: 38:31)



For example, we can say we want a classifier that has the least probability of misclassifying a random pattern. Let us for this class, quickly derive what we is meant by least probability of misclassification by putting some notation. Now, let us say q i of X is the probability that class is i for a given vector X , that is given the random variable X was the probability that it comes from class i. So, q 1 of X, obviously the function of X is conditioned on X. These are called posterior probabilities or posterior probability function for class i.

Now consider a classifier. A classifier is a function that maps X to 0, 1 because we are considering a 2 class problem. So, a classifier is simply defined for which all X it is 0, it is 1, and so on. So, I define a classifier h of X which takes value 0 if q 0 of X is greater than q 1 of X, else it takes value 1. Hence, when I given an X, I calculate the numbers q 0 of X and q 1 of X, if q 0 of X stands out to be higher than q 1 of X, then I will put it in class 0. Otherwise, I will put it in class 1 because q 0 is the probability that it comes from class 0. When q 0 of x greater than q 1 of X, we would intuitively imply that the feature vector X is more likely to come from class 0 rather than from class 1.

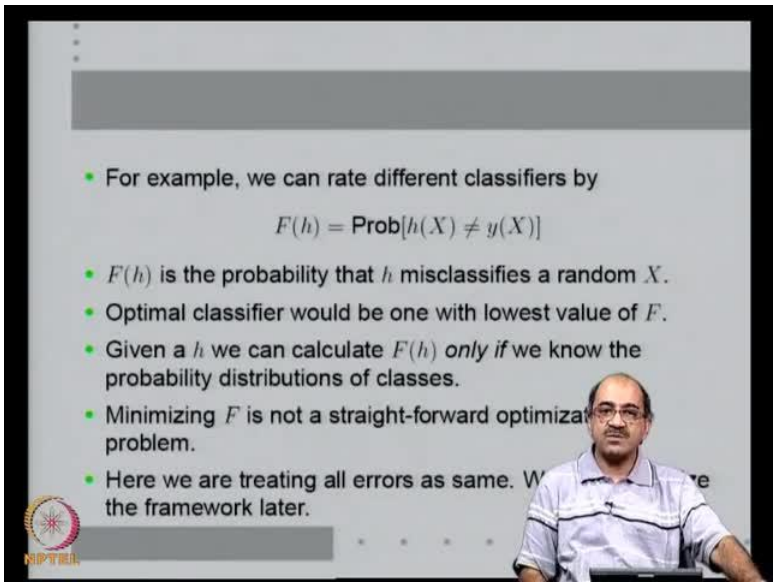
(Refer Slide Time: 39:58)



So, intuitively we should think that such a classifier will minimize probability of error in classification. So, we will actually prove it later, but let us just look at, how can we formalize at least the concept of probability of error. So, let us say X, which is for us an n dimensional real nuclear space is the feature space and y is equal to this set containing 0 and 1 is the set of class labels. So, we are considering a 2 class problem, right? Whenever considering the designing classifiers you should ask what all classifiers should we design, say linear classifiers or quadratic classifiers. So, we may have some set of functions which is the set of classifiers of interest to us. Any classifier is the function that with domain script X and raise y for each and each feature vector I assigns a class label.

Every such function is a classifier. There is some set of functions that I choose which the classifiers of interest to me are... Now, out of this bag of classifiers I want to know which is the best classifier? So, which means that for every such classifier I want to assign a figure of merit, to assign a figure of merit, we would need to know how well the classifier does on various feature vectors. So, for a given feature vector X, let us say y of X denotes the class label of X because this same X can come from different classes with different probabilities. This y of X is not a deterministic function. It is a random function given an X, y of X can be 0 with some probability, one with some other probability, etcetera. So, y of X is the function of a random variable X.

(Refer Slide Time: 41:29)



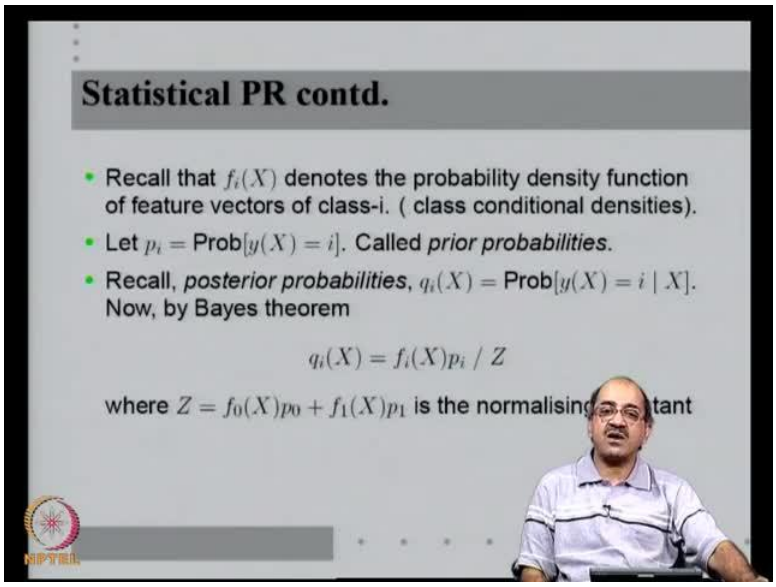
So, with that notation let us go and define some figure of merit so that for each classifier, we can say is how good or how bad it is. So, 1 figure of merit could be before each classifier h. I assign a number F of h, which is capital F of h, which is probability that h of X is not equal to y of X. The classifier would say that this h of X, is any y of X. So, what the nature would say is an X, right? That is the unknown true class for the pattern X. These probabilities are with respect to the underlying probability distributions of X, so the probability h of X is not equal to y of X, is the probability that under random variable X my classifier h will make an error.

So, F of h is the probability of misclassification by the classifier h. So, I am asking that among all the classifiers, which classifier has the least value of F of h. So, the optimal classifier can now be defined as 1 that is lowest value of F. Now, given a particular classifier h, we cannot calculate F of h unless we know all the underlying probability distributions. I need to know the distribution of X and I need to know what kind of function y is of X. Then, I can calculate that probability. Even if I can, I do not know how to minimize this F of h. So, it is not a straight forward optimization problem, but let that be. We will look at a solution techniques later on.

Right now, we just want to formulate what is an optimal classification. Here is one more thing that is does not take care. I am only looking at errors. So, I am asking what is the probability it makes an error? I am not saying whether it is classifying class 0 pattern as

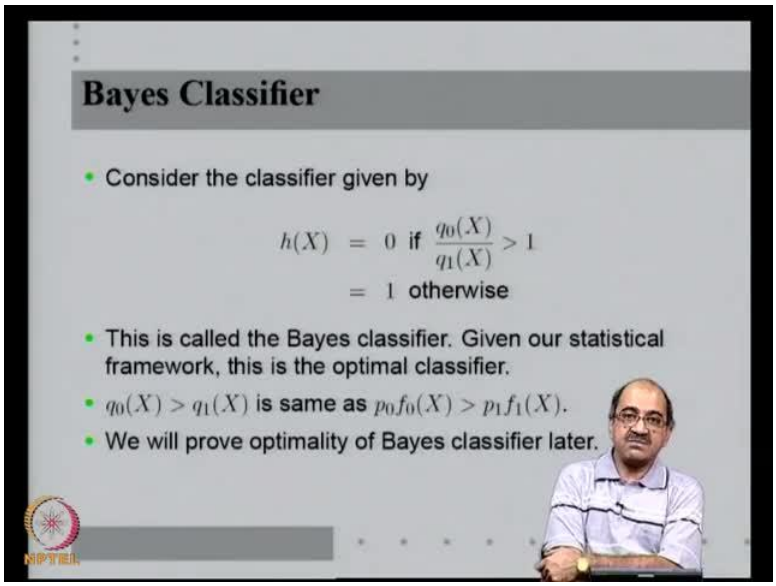
class 1 or class 1 pattern as class 0 pattern. So, all kinds of errors are same. In general, sometimes they are not. We will see examples. Later on I will come to that, but sometimes classifying a class 0 pattern as class 1 pattern may incur more cost than classifying a class 1 pattern as class 0 pattern.

(Refer Slide Time: 43:42)



In that case, just simply finding the probability of an making an error small may not be what you want, but it does not matter. Let us stick with this. We just want to minimize probability of making an error. Now, I will give you a classifier that minimizes the probability of making an error, which is called Baye's classifier. Before this, let us just recall some notation. f i of X is the probability density function of feature vectors of class, p i, which is the probability that y of X is equal to i. I will call prior probabilities, that is if I do not know any X, I am just asking what is the chance that I get a class a pattern. So, these are called a pair because I am just asking what is the chance I get class 1 patterns or what is the chance that I get class 0 patterns.

(Refer Slide Time: 44:53)



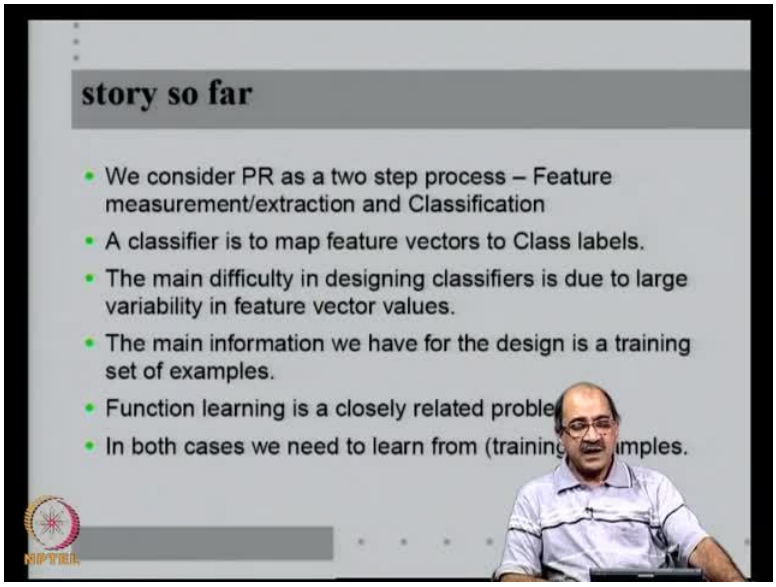
We only define posterior probabilities and by Baye's theorem I am sure all of you know that for q i of X there is a probability y of X equal to i, given X i can invert it. That is, f i of X which is the density of X conditioned on y of X is equal to i into probability of y of X. This is equal to i. So, q i of X is f i of X into p i by some normalising constant in the Baye's theorem. So Baye's theorem tells me that if I know class conditional densities and prior probabilities, I can calculate the posterior probabilities up to a normalising constant. Now, consider this classifier, h of X is equal to 0 if q 0 of X by q 1 of X is greater than 1. This is same the thing what we considered, that is q 0 of X greater than q 1 is 1. This classifier is called the Baye's classifier because I am actually calculating q 0 on q 1 using f 0, p 0 and p 1 using f 1.

From what I spoke to you earlier intuitively, q 0 is greater than q 1. This means that there is more chance that X is coming from class 0 than class 1. Hence, we would expect this to be optimal and i fit is in fact optimal. We will prove that the Baye's classifier is optimal in the sense that, no other classifier can have lower probability of error. Using the Baye's classifier, I have emphasized once again that q 0 on q 1 can be calculated if I know p 0 f 0. In particular, q 0 X greater than q 1 X is same as p 0 f 0 X greater than p 1 f 1 X.

So, I will always implement the Baye's classifier by knowing class conditional densities and prior probabilities. Of course, at this point of time, I do not know from where I am

getting them from, but we will come to that late. So, this is an example classifier. For the first class we at least look at one example classifier. This is a very good classifier. This is not any trivial classifier. It actually minimizes the probability of error. Since, this is the only thing we do not know, so we have designed a classifier that minimizes probability of error. The only thing we do not know is how actually we will calculate the classification decision. Given an X, to calculate h of X, I need to know q 0 on q 1, p 0, p 1 and the class conditional densities of 0 and f.

(Refer Slide Time: 47:06)



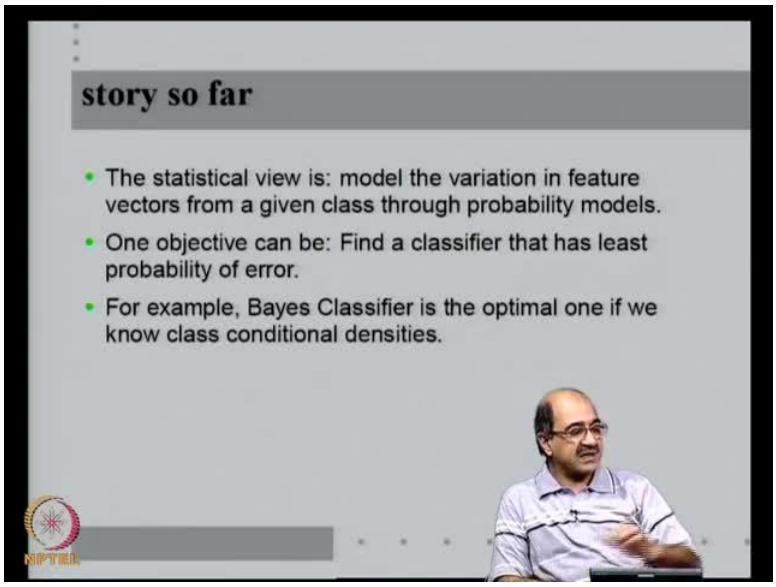
So, if somebody gives me p 0, p 1, class conditional densities f 0 and f 1, then I can implement Baye's classifier. I am at the moment not worrying about that. In the next class we will certainly prove the optimality of the Baye's classifier. So, let us sum up what we have said in the introduction talk. The machine recognition of patterns is a 2 step process. It consists of feature measurements and classification. The first block, given any pattern, could be an image, which could be a one-dimensional signal, which could be combination of these. It makes some measurements and converts into vector measurements; that is called the feature vector.

For us feature vector is a n dimensional real vector, that is it resides in the n dimensional real vector space and the classification block take such feature vectors and assigns class labels. The classifier maps feature vectors to class labels. So, it is the function from the feature space to the set of all class labels. The main difficulty in designing classifiers is

due to the large variability in feature vector values. So, when we take patterns belonging to the same class and make feature measurements there is lot of variability in this these values. Patterns coming from the same class can give you widely different values and pattern coming from different classes can give you very close values.

So, that is the difficulty that we have to handle and the only information we have for designing the classifier is a training set of examples. A training set of examples consists of X i, y i where X i is a feature vector and y i is the true class for the feature vector X i. I generate the training set by actually sampling from the respective distributions. We also saw that function learning is a closely related problem where the output instead of being a category, instead of being finitely valued, is actually a continuous valued number. In both cases we need to learn from the examples.

(Refer Slide Time: 49:04)

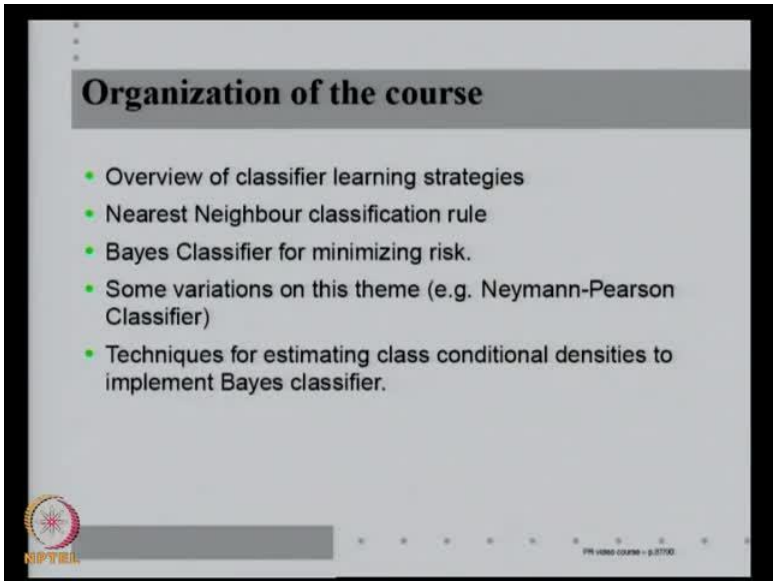


The statistical view is that we model the variation feature vectors of a given class through probability models. So, we will say that all feature vectors coming from class 0 take different values at different probabilities as given by a density function or mass function f 0 X. So, they are called the class conditional densities. So, the statistical view models variations in feature vector values through probability models.

For example, in this class what we saw that such a view point gives us at least 1 advantage. We can define an optimal classifier. For example, we can ask to find a classifier that has the least probability of making an error. The characterization, least

probability of error make sense because we have a statistical view point and we are modelling feature vectors as random variables with given class conditional densities. So, for that criteria of optimality having least value of probability of error being your criteria of optimality, Baye's classifier is the optimal 1. So, if we know the class conditional densities and prior probabilities, we can calculate the posterior probabilities.

(Refer Slide Time: 50:29)



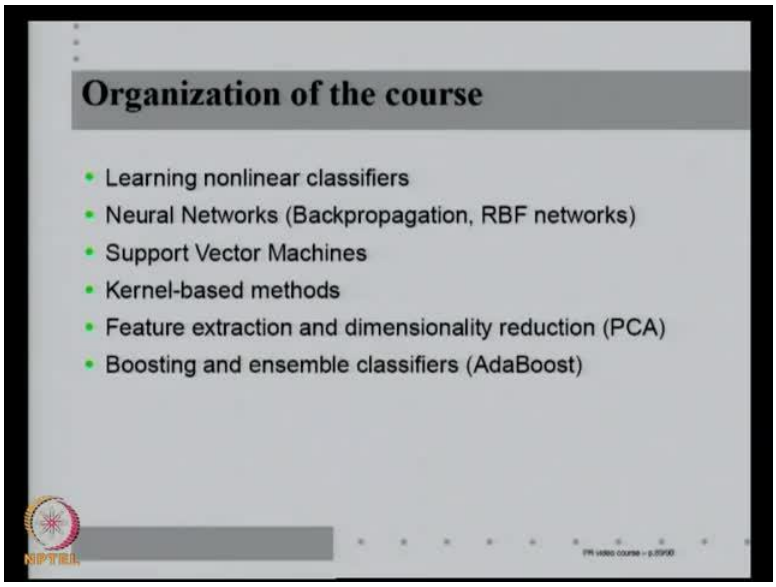
Then the Baye's classifier can be implemented and as we shall in the next class that Baye's classifier is optimal, in the sense that no other classifier can have lower probability of error than Baye's classifier. Because, this is the first class I will end the class with a rough organization of the course. So, in the next 1 or 2 talks, we will look at the general overview of classifier learning statistics. So, we will look more closely at the Baye's classifier and also we will look at many other kinds of classifiers. Then I will give you a rough overview of the various classifier design methodologies and how they fit in.

Then, we will go into each of them in depth. We will first look at the Baye's classifier that minimizes, what I call risk. This is the generalization of probability of error. Then, we will look at some variations of the Baye's classifier. Then we will look at how to implement a Baye's classifier, how do 1 get class conditional densities and prior probability so that Baye's classifier can be implemented. Then we will look at if I cannot get class conditional densities, what else can I do? We will look at the techniques what

are called linear disciplined functions. We will look at special algorithms for learning linear disciplined functions. We will also look at function learning using linear models. What are called linear least space regression? So, once having seen some examples of how to learn classifiers, we will take a step back and look at the theory of learning.

We will look of course, we would not do at any great depth. This is a first level course of pattern recognition. So, we will look at a very simple overview of what is called statistical learning theory and empirical decriminalization which is what most classifiers follow. So, we will give some statistical theory to justify the kind of classifier design strategies that we followed. Then, we will look at non-linear classifiers till all the discriminant function that we will, we will look at in the beginning are all linear. Then, we will look at learning non-linear classifiers.

(Refer Slide Time: 52:34)



We will essentially look at two kinds of methods there; one is called neural networks and the other is called support vector machines. Support vector machines belong to a class of methods called kernel based methods. So, we will look at in detail, the designing of non-linear classifiers both for using neural networks and using support vector machines. Then we will look at a little more theory of support of machines called kernel based methods. Kernel based methods today are possibly the best of the self-strategies for designing classifiers. Then we will look at a few special topics. So, two topics we will certainly look at are, some feature extraction dimensional reduction methods called as principle

component analysis and then we will look at another technique for classifier design which is called boosting technique. From the first part or almost till the end, we will be only looking at, given the training set how to learn many classifiers.

The idea is that, if we cannot learn 1 classifier then it is very good if I learn many independent classifiers. Using this same training set, can I somehow get better than a single classifier? So, if I ten different classifiers, each of them is asked to do the classification of the same unknown pattern and then we take majority decision. Sometimes, we may do better than using any one classifier. So, such boosting methods for example, are similar to the audience poll in Kaun Banega Crorepathi kind of programs, where each member of the audience may know the correct answer with probability only 50 percent or 60 percent. But if you ask (( )) of them we would know, what is the correct answer? So, ensemble classifiers is something that has become very important in the field in the last 10 years. So that is also one topic, we will consider. After that time permitting, we will look at a few more advanced topics in classifications such as bayesian networks.

So, that is the overview of the course, and thank you at the end of the first lecture.