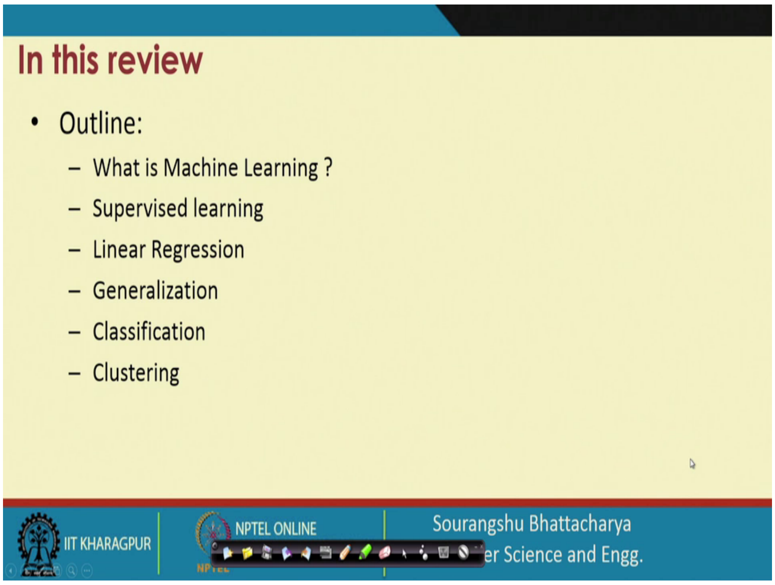
Scalable Data Science
Prof. Sourangshu Bhattacharya
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 05
Background on Machine Learning

Hello students, welcome to the fifth lecture on NPTEL course on Scalable Data Science. Today we are going to discuss background on machine learning. I am Professor Sourangshu Bhattacharya of Department of Computer Science and Engineering, IIT, Kharagpur.

(Refer Slide Time: 00:35)

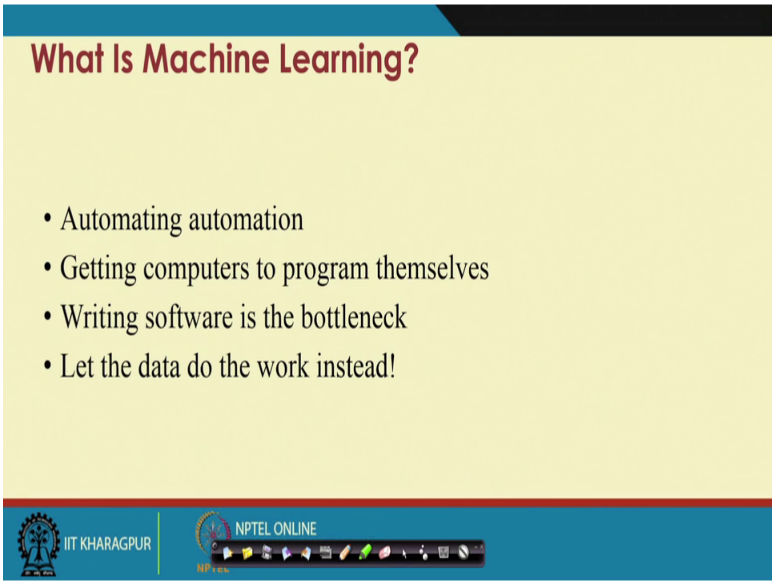


In this review

- Outline:
  - What is Machine Learning ?
  - Supervised learning
  - Linear Regression
  - Generalization
  - Classification
  - Clustering

So, today is review lecture in which we will review, so we will cover what is machine learning, the basic idea of machine learning, what is supervise learning and we will see an example of supervise learning which is linear regression. And then we will see the concept of generalization which is a concept in supervise learning, then we will see the classification problem and then we will see unsupervised learning problem of clustering.

(Refer Slide Time: 01:15)



What Is Machine Learning?

- Automating automation
- Getting computers to program themselves
- Writing software is the bottleneck
- Let the data do the work instead!

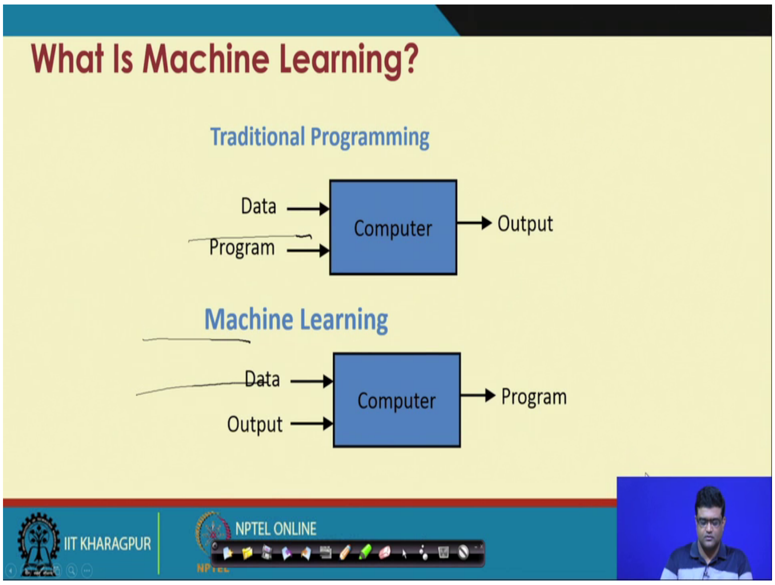
IIT KHARAGPUR
NPTEL ONLINE

So, what is machine learning? So, one way to think about machine learning is that it is automating automation. So, that term automation was used, so the term automation was used for when manual tasks were done by computers. So, and we wrote programs to perform this repetitive manual tasks in a quicker and more efficient manner, ok

So, now machine learning is the next step where we write programs which or rather we develop programs which can write themselves. So, the programs they write themselves and they start from using data, and they and they develop into a working program as we shall see some of the examples. So, so the reason this came about is because for many tasks writing the program or the software became the bottle neck. Now, why did writing the program become the bottle neck?

So, one example is let us say spam classification where one has to write many rules in order to detect whether particular email is spam or not. So, the program writing the program the program for it became too complex and hence it became a bottleneck. And as mentioned so in machine learning instead we use the data to write the program all by itself.

(Refer Slide Time: 03:32)



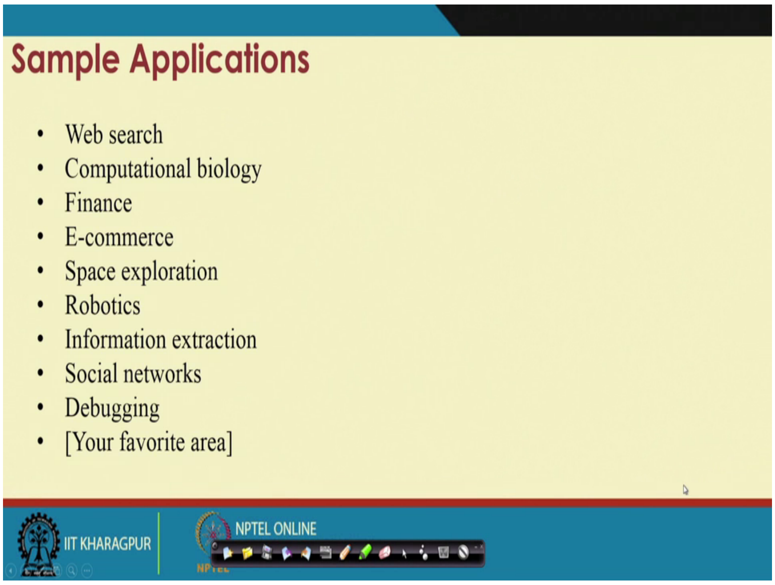
What Is Machine Learning?

Traditional Programming

Machine Learning

So, this is the picture. So, in traditional programming the inputs are data. So, we provide the data and we provide the program to a computer and the computer computes the output. In case of machine learning we provide the data and the desirable output to the computer, and the computer tries to come up with a program or machine learn program as we say which can compute the output given this data.

(Refer Slide Time: 04:16)

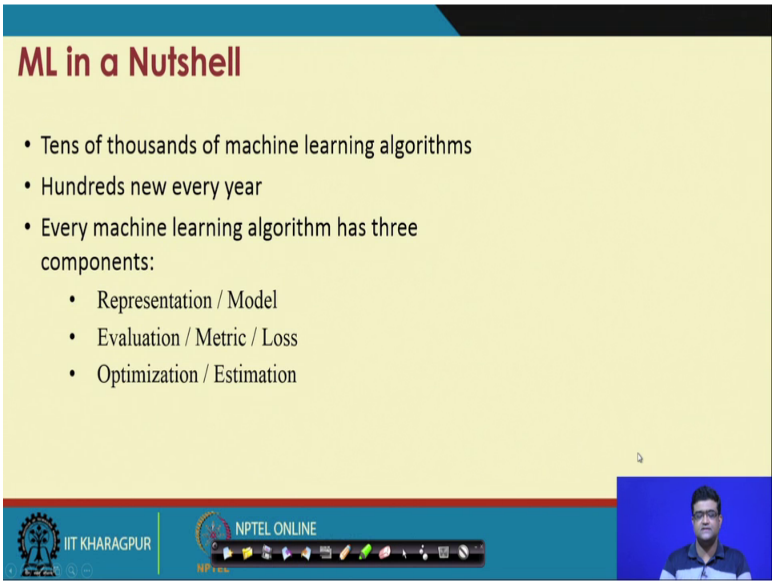


Sample Applications

- Web search
- Computational biology
- Finance
- E-commerce
- Space exploration
- Robotics
- Information extraction
- Social networks
- Debugging
- [Your favorite area]

So, there are many example applications for example, web search the ranking in web search users machine learning too computational biology finance E-commerce. So, some of the recommendation algorithms used in E-commerce use machine learning space exploration robotics again some of the control algorithms use machine learning, information extraction, social networks and so many applications.

(Refer Slide Time: 04:59)



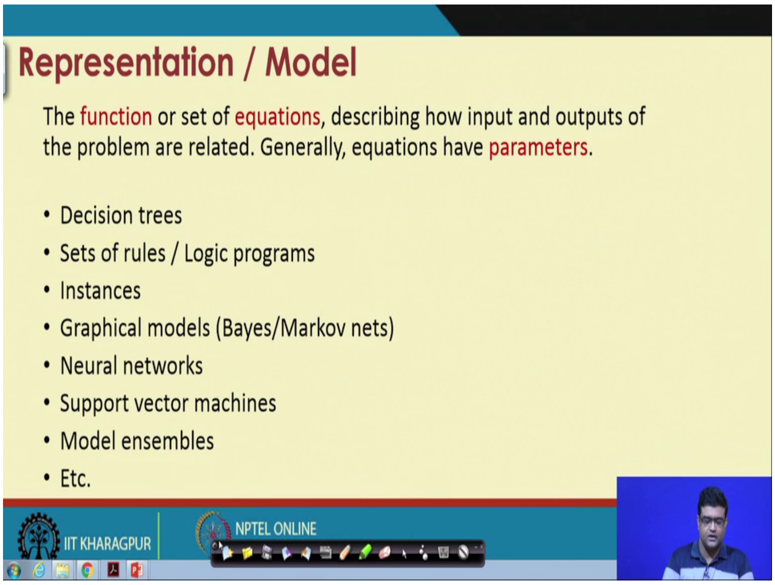
ML in a Nutshell

- Tens of thousands of machine learning algorithms
- Hundreds new every year
- Every machine learning algorithm has three components:
  - Representation / Model
  - Evaluation / Metric / Loss
  - Optimization / Estimation

So, machine learning now is a tool which finds global applications. So, there are as you all know there are tens of thousands of machine learning algorithms which already exists and hundreds, hundred more are proposed or developed every year.

And but if we want to broadly categorize what a machine learning algorithm is, ok, so every machine learning algorithm has roughly 3 components. The first component is the representation model which is we will describe what representation model is. The second component is the evaluation is called the evaluation or the metric or the loss. And third component is the optimization or the estimation component. So, whenever you are writing a machine learning program you need to worry about these 3 components, how these 3 components should look for a particular machine learning problem?

(Refer Slide Time: 06:23)



Representation / Model

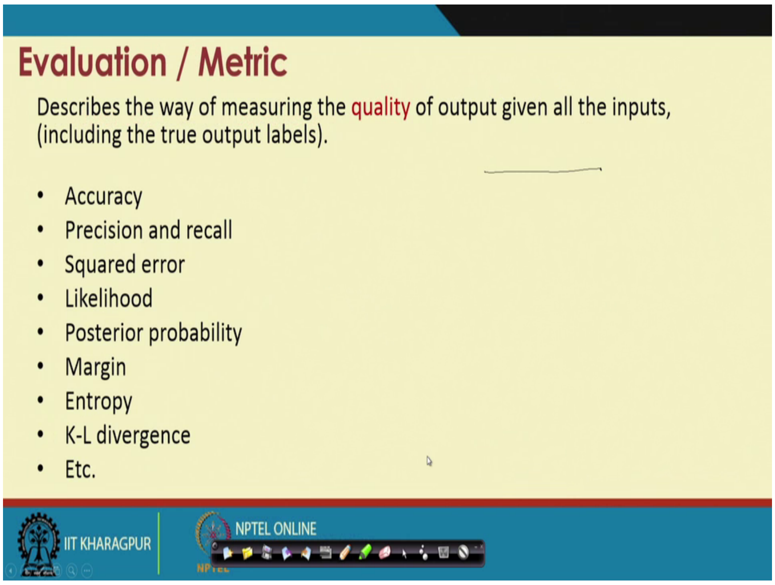
The function or set of equations, describing how input and outputs of the problem are related. Generally, equations have parameters.

- Decision trees
- Sets of rules / Logic programs
- Instances
- Graphical models (Bayes/Markov nets)
- Neural networks
- Support vector machines
- Model ensembles
- Etc.

So, the first component is representation or the model. So, what is the representation or the model? It is a function or a set of equations which describe how the inputs and the outputs of the problem are related. So, for example, there could be a model or. So, consider the span classification class. So, there the model would take would be a set of equation which takes has input let say spam email or normal email and classifies when the output is 0 or 1, where 0 means its it is a normal email and 1 means its spam email. So, normally this function would have certain parameters. We will describe what these parameters are. And these parameters are crucial because these parameters are what we learn in order to come with the good machine learning program.

And the set of parameters depend on the particular machine learning model. So, here are some examples for machine learning models decision trees, set of rules, instances, graphical models, neural networks, support vector machines etcetera, so all these are models for machine learning.

(Refer Slide Time: 08:03)



Evaluation / Metric

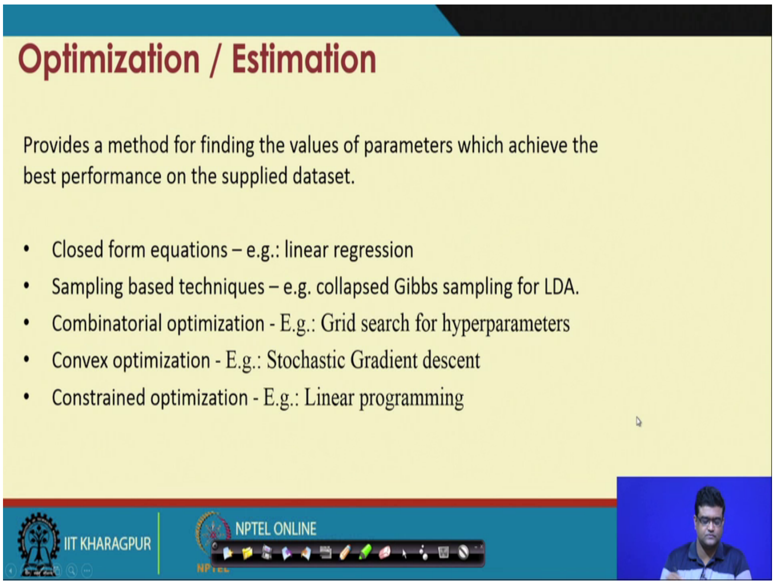
Describes the way of measuring the quality of output given all the inputs, (including the true output labels).

- Accuracy
- Precision and recall
- Squared error
- Likelihood
- Posterior probability
- Margin
- Entropy
- K-L divergence
- Etc.

The second one the second component that one must worry about is the evaluation metric, the evaluation or the metric, So, this describes a way of measuring the quality of the output given all the inputs, including in incase of for example, supervise learning the true output levels.

So, for example, in case of the spam classification task good evaluation metric could be accuracy which measures the fraction of emails which are classified currently. So, spam emails are classified a spam and non spam emails classified as non spam. So, depending on the problem at hand many evaluation measures or metrics are used and examples include accuracy, precision recall, squared error in case of regression. So, accuracy and precision recall are typically measured in case of classification problem. Likelihood, posterior, probability etcetera are used in case of probabilistic models. Margin, entropy these are also general K-L divergence these are also general matrix which are used sometimes in case of certain modules.

(Refer Slide Time: 09:46)



Optimization / Estimation

Provides a method for finding the values of parameters which achieve the best performance on the supplied dataset.

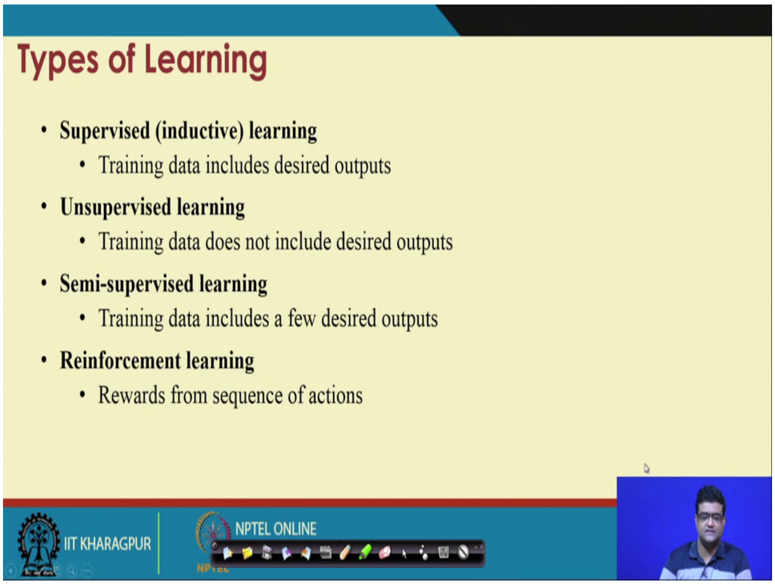
- Closed form equations – e.g.: linear regression
- Sampling based techniques – e.g. collapsed Gibbs sampling for LDA.
- Combinatorial optimization - E.g.: Grid search for hyperparameters
- Convex optimization - E.g.: Stochastic Gradient descent
- Constrained optimization - E.g.: Linear programming

The third component is the optimization or estimation component. So, this component provides a method for finding the values of parameters which achieve the best performance on the supplied data sets. So, typically the supplied data set is called the training dataset. So, again the actual method used for estimating the parameters or optimizing the parameters will depend on the model and the metric that you use.

So, for example, in case of linear regression you have closed form equations from which you can compute the optimal model or the optimal representation or optimal parameters of the model. Sometimes in some probabilistic models one would use sampling based techniques like Gibbs sampling or some other kind of sampling based techniques for estimating the parameters. In some cases one would use a combinatorial optimization technique such as the simple techniques of grid search or sometimes logarithmic grid search for example, in order to estimate sometimes hyper parameters of certain models.

Probably the convex optimization is the most commonly used estimation technique where. So, so an example of this is stochastic gradient descent. So, this can be used when the objective function or the loss function that is that the model has is convex and then one can optimize it using the convex optimization techniques. Sometimes one can also use certain constraint optimization techniques for example, linear programming and some other more general constraint optimization techniques.

(Refer Slide Time: 12:02)



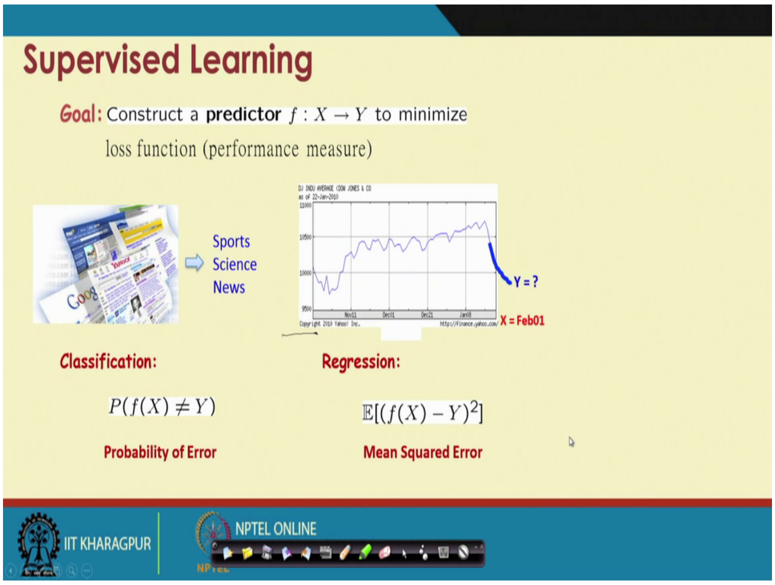
Types of Learning

- Supervised (inductive) learning
  - Training data includes desired outputs
- Unsupervised learning
  - Training data does not include desired outputs
- Semi-supervised learning
  - Training data includes a few desired outputs
- Reinforcement learning
  - Rewards from sequence of actions

So, with this we come to the types of learning. So, broadly the traditional machine learning can be can be split into two types - one is the supervised learning, the other is the unsupervised learning. So, in case of supervised learning the labels are for the training data are supplied, whereas in case if unsupervised learning you only have the training data point but you to infer the labels all by yourself. So, example of supervised learning is classification task we just described and example of unsupervised learning is the clustering task where you are given the data points and you have to find natural groupings among the data points and assign each data points to one of the groups.

So, in addition to this two other new forms of learning are semi supervised learning where which is kind of combination of supervised and unsupervised learning where a few desired outputs are given but not all the training data has labels. Another form of learning is called reinforcement learning, where rewards for a sequence of action are given are provided and one has to learn a policy which will decide given a state an action and next state. In this talk we would mostly discuss supervised and unsupervised learning techniques.

(Refer Slide Time: 14:07)



Supervised Learning

Goal: Construct a **predictor** $f : X \to Y$ to minimize loss function (performance measure)

Classification:

$P(f(X) \neq Y)$

Probability of Error

Regression:

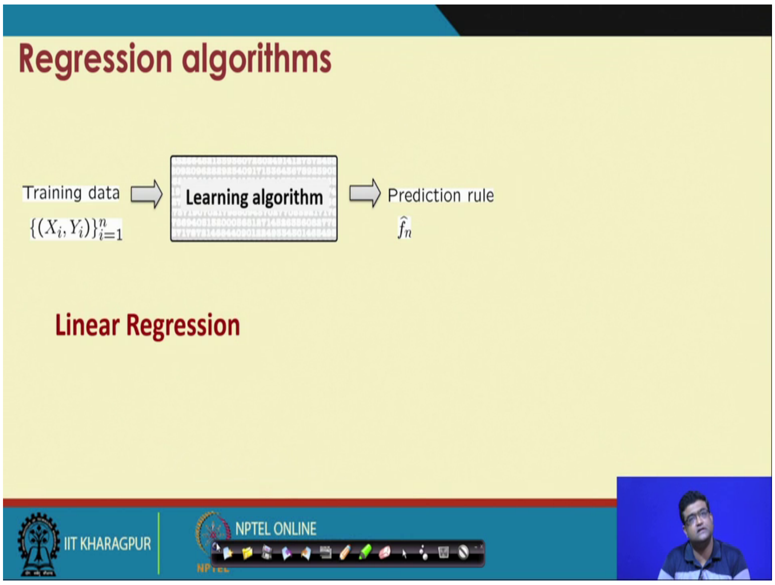
$\mathbb{E}[(f(X) - Y)^2]$

Mean Squared Error

So, what is supervised learning? So, as mentioned earlier the goal is to construct the predictor say f. So, f is your predictor and X is the set of all inputs and Y is the set of all outputs. So, examples are, so given a web page or text document or news article you may want to classify it into one of these 3 categories say sports, science news, etcetera. And in this case a good way of measuring the performance of such a classifier or such a machine learning predictor would be to calculate the probability that given a document X the predicted value which is f of X is not equal to Y; this is something like the error rate. So, this is the probability of error or the error rate for the classifiers.

So, in this case as you can see the labels are discrete, another class of supervised learning problems are these regression problems. So, example is let say you want to predict the stock market value for or the average stock market index for set of shares at a feature date. So, your input is all the past stock market values and your output is your output which is Y is the average stock market value at a feature date.

(Refer Slide Time: 16:45)



Regression algorithms

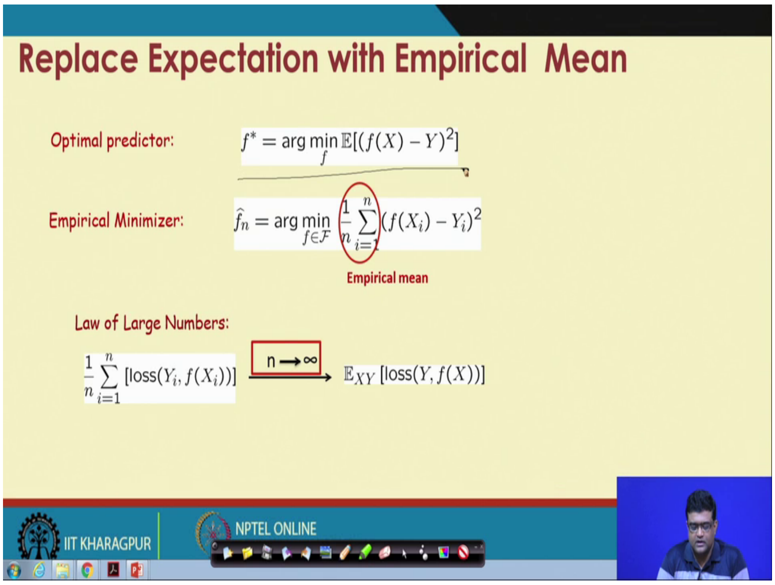
Training data $\{(X_i, Y_i)\}_{i=1}^{n}$ ⇨ Learning algorithm ⇨ Prediction rule $\hat{f}_n$

Linear Regression

And a good way to measure the performance of a predictor like this the stock market predictor would be the expected means squared error. So, this is the, so this is the difference square of difference between the predicted value and the true value which is given. So, this is the general scheme of things. So, we will now go into the first supervised learning problem that we are going to discuss which is the regression problem or the linear regression problem.

So, the problem is you are given a set of training data points. So, you are given a set of training data point in terms of X i and Y i. So, for example, X i is the vector of past stock prices and Y i is the feature stock price. And you want to have a learning algorithm which will give you prediction rule using which you can predict given any at any point in time you can predict the future stock prices.

(Refer Slide Time: 17:52)



Replace Expectation with Empirical Mean

Optimal predictor: $f^* = \arg\min_f \mathbb{E}[(f(X) - Y)^2]$

Empirical Minimizer: $\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n}\sum_{i=1}^{n}(f(X_i) - Y_i)^2$

Empirical mean

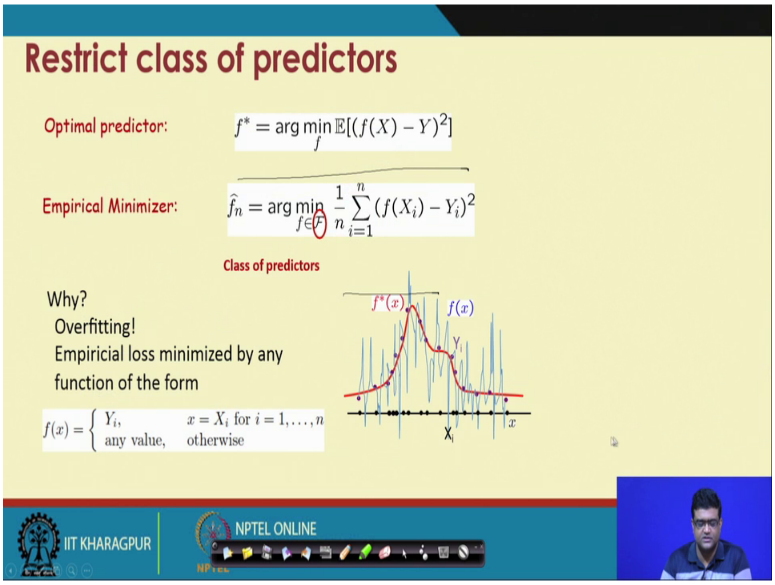
Law of Large Numbers:

$$\frac{1}{n}\sum_{i=1}^{n}[\text{loss}(Y_i, f(X_i))] \xrightarrow{\boxed{n \to \infty}} \mathbb{E}_{XY}[\text{loss}(Y, f(X))]$$

IIT KHARAGPUR NPTEL ONLINE

So, as discussed earlier the optimal predictor would be the one that minimizes the one that minimizes this expected error, expected mean squared error. Now, the problem is that it is not possible to compute this expected mean squared error but what we can compute is the empirical mean squared error which is that given a set of training data point instead of the expectation you replace it with the empirical mean and you can compute this. And as your number of training data points increases the empirical mean becomes a good proxy for the actual expected error.

(Refer Slide Time: 18:48)



Restrict class of predictors

Optimal predictor: $f^* = \arg\min_f \mathbb{E}[(f(X) - Y)^2]$

Empirical Minimizer: $\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n}\sum_{i=1}^{n}(f(X_i) - Y_i)^2$

Class of predictors

Why?

Overfitting!

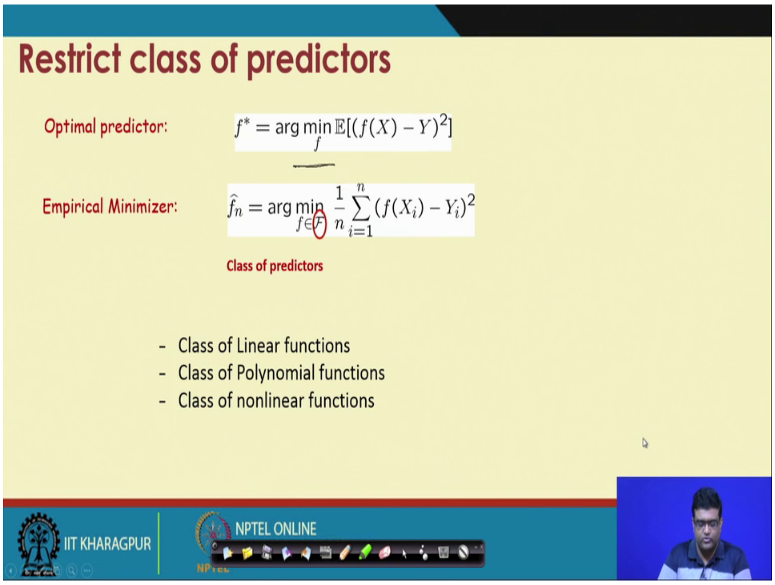
Empiricial loss minimized by any function of the form

$$f(x) = \begin{cases} Y_i, & x = X_i \text{ for } i = 1, \ldots, n \\ \text{any value}, & \text{otherwise} \end{cases}$$

IIT KHARAGPUR NPTEL ONLINE

Now, more over sometimes you may want to restrict the class of predictor. So, for example, earlier we were minimizing over all predictors but this may not be possible for many practical reasons. So, the first reason could be that you cannot write in a parametric manner the class of all predictors where as for example, you can write the class of all linear predictors. Another reason could be over fitting, we will see about over fitting more later.

(Refer Slide Time: 19:40)



Restrict class of predictors

Optimal predictor: $f^* = \arg\min_f \mathbb{E}[(f(X) - Y)^2]$

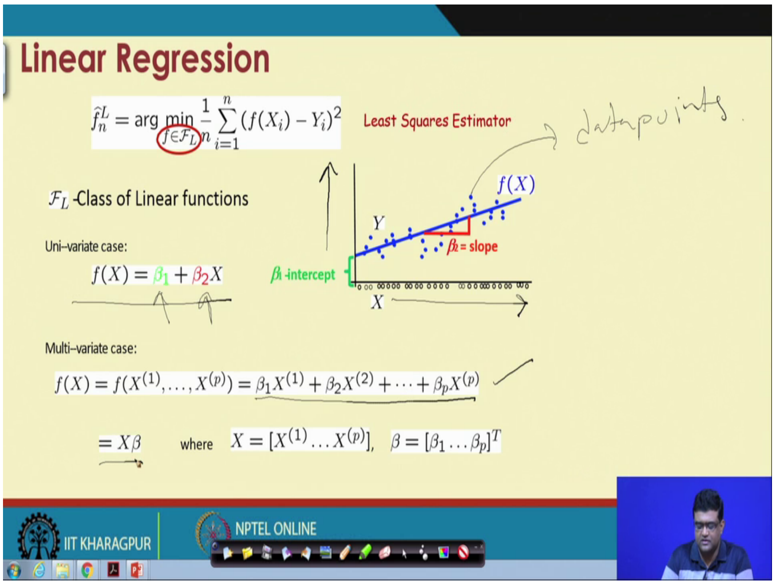
Empirical Minimizer: $\hat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2$

Class of predictors

- Class of Linear functions
- Class of Polynomial functions
- Class of nonlinear functions

So, some common class of predictors that I used for example, in case of regression are the class of linear function which we shall see also the class of polynomial functions or the class of some general non-linear function. So, the first one would give rise to linear regression, the second one would give rise to a polynomial regression and the third one could be captured using neural network type approach.

(Refer Slide Time: 20:16)



Linear Regression

$$\hat{f}_n^L = \arg\min_{f \in \mathcal{F}_L} \frac{1}{n} \sum_{i=1}^{n} (f(X_i) - Y_i)^2$$ Least Squares Estimator

$\mathcal{F}_L$ -Class of Linear functions

Uni–variate case:

$$f(X) = \beta_1 + \beta_2 X$$

Multi–variate case:

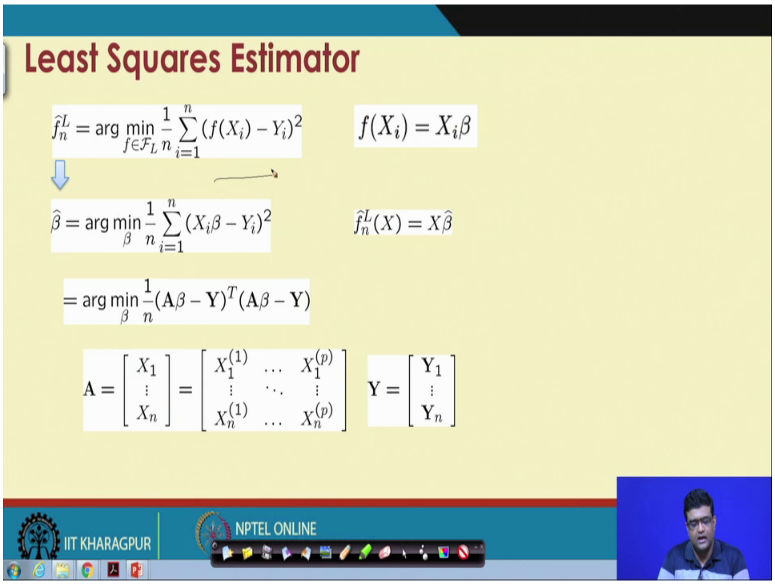
$$f(X) = f(X^{(1)}, \dots, X^{(p)}) = \beta_1 X^{(1)} + \beta_2 X^{(2)} + \dots + \beta_p X^{(p)}$$

$= X\beta$ where $X = [X^{(1)} \dots X^{(p)}]$, $\beta = [\beta_1 \dots \beta_p]^T$

So, this is the linear regression model. So, suppose, so in the uni-variate case where your input is just one number. So, this X is just one number X is plotted in this axis. And your output is also one number which is the Y and your given many data points so these blue points here. So, each of these blue points are data points. And the task is, so in case of linear estimation the task is to fit a line which is the blue line here through this data points. And this line any such line is given by two parameters beta 1 and beta 2. So, beta 1 is the intersect in this case and beta 2 is the slope.

So, this can be extended also to the general multi variate case where you have many inputs. So, in that case your f of X becomes this summation over beta i, X i for all the inputs i, and this can be written in matrix notation or in vector notation like this.

(Refer Slide Time: 21:51)



Least Squares Estimator

$$\hat{f}_n^L = \arg\min_{f\in\mathcal{F}_L} \frac{1}{n}\sum_{i=1}^{n}(f(X_i)-Y_i)^2 \qquad f(X_i) = X_i\beta$$

$$\hat{\beta} = \arg\min_{\beta} \frac{1}{n}\sum_{i=1}^{n}(X_i\beta - Y_i)^2 \qquad \hat{f}_n^L(X) = X\hat{\beta}$$

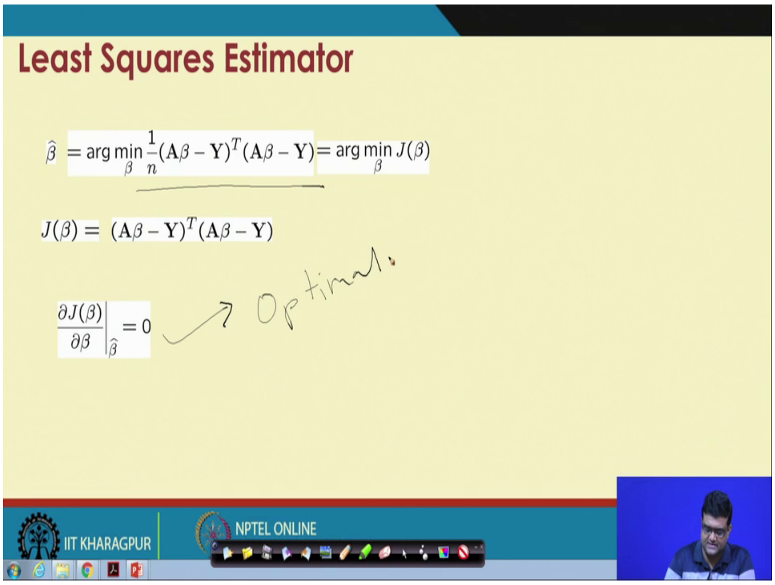
$$= \arg\min_{\beta} \frac{1}{n}(\mathbf{A}\beta - \mathbf{Y})^T(\mathbf{A}\beta - \mathbf{Y})$$

$$\mathbf{A} = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix} = \begin{bmatrix} X_1^{(1)} & \dots & X_1^{(p)} \\ \vdots & \ddots & \vdots \\ X_n^{(1)} & \dots & X_n^{(p)} \end{bmatrix} \qquad \mathbf{Y} = \begin{bmatrix} \mathrm{Y}_1 \\ \vdots \\ \mathrm{Y}_n \end{bmatrix}$$

So, we have already discussed the objective function. So, this is the least square objective function here ah. Now, we want to minimize this under the constraint or under the constraint that f X is linear function as described earlier. And if we, so we can write this summation out and you can check that we can convert. So, so we can convert this equation which has a summation in it into this equation which is a matrix equation.

So, this involves a matrix A, so A is the matrix of all data points. So, the rows of the matrix A are all the data points and this bold Y matrix. So, this is the A matrix and this is the bold Y matrix which is the value for all data points, and you can see that this two are same. You can just expand this out to see that this is indeed equal to this.

(Refer Slide Time: 23:30)



Least Squares Estimator

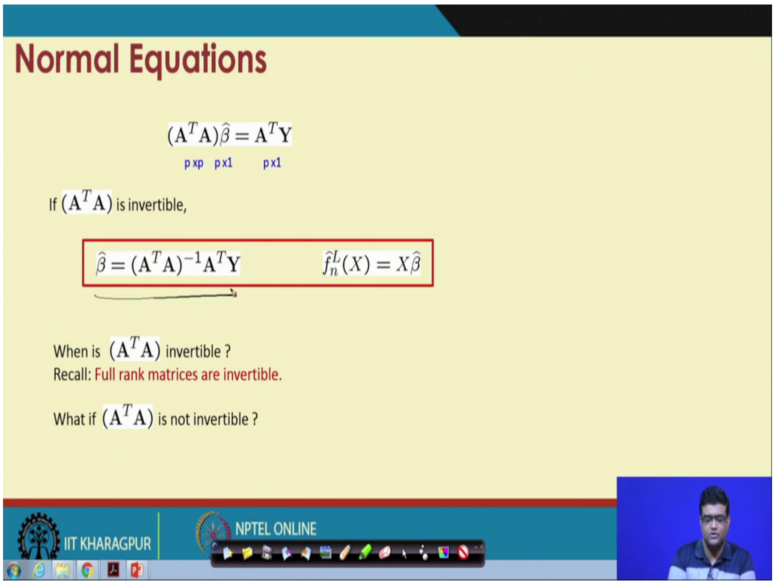
$$\hat{\beta} = \arg\min_{\beta} \underline{\frac{1}{n}(A\beta - Y)^T(A\beta - Y)} = \arg\min_{\beta} J(\beta)$$

$$J(\beta) = (A\beta - Y)^T(A\beta - Y)$$

$$\left.\frac{\partial J(\beta)}{\partial \beta}\right|_{\hat{\beta}} = 0$$ ↗ Optimal.

Now, once you have this you can just differentiate you can just differentiate this equation or this rather the formula you can just differentiate this formula with respect to beta which is the parameters, and set it to 0. So, that is how you can get the optimal betas, ok, as we have discussed in the optimization lecture.

(Refer Slide Time: 24:07)



Normal Equations

$$\underbrace{(A^TA)}_{p\,x\,p}\underbrace{\hat{\beta}}_{p\,x1} = \underbrace{A^TY}_{p\,x1}$$

If (A^TA) is invertible,

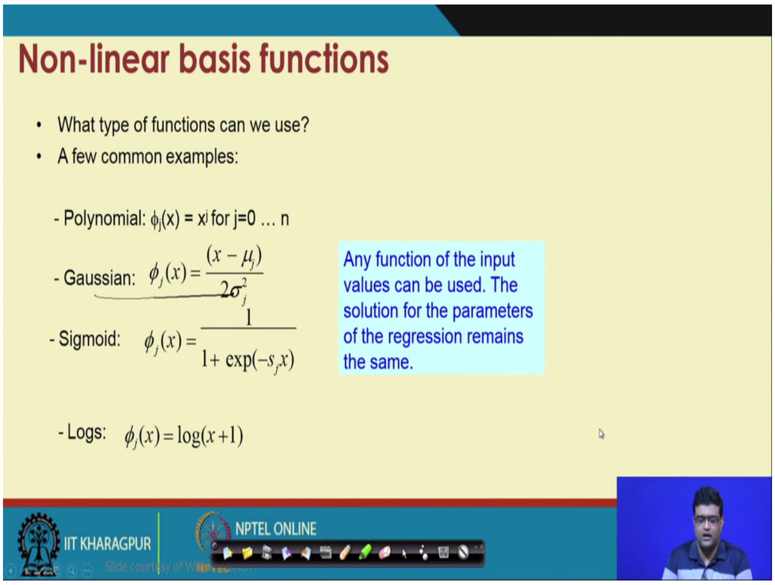
$$\boxed{\underline{\hat{\beta} = (A^TA)^{-1}A^TY} \qquad \hat{f}_n^L(X) = X\hat{\beta}}$$

When is (A^TA) invertible ?
Recall: Full rank matrices are invertible.

What if (A^TA) is not invertible ?

And once you do this you get you get what is called the normal equations. So, the normal equations are basically these are the normal equations. So, this is a closed form solution for the beta or the parameter given the data matrix A and the matrix of labels Y. Now, when, so you can compute this when a transpose a is invertible if a transpose a is not invertible we can use a kind of regularizer.

(Refer Slide Time: 25:00)



Non-linear basis functions

- What type of functions can we use?
- A few common examples:

- Polynomial: $\phi_j(x) = x^j$ for j=0 … n
- Gaussian: $\phi_j(x) = \frac{(x-\mu_j)}{2\sigma_j^2}$
- Sigmoid: $\phi_j(x) = \frac{1}{1+\exp(-s_j x)}$
- Logs: $\phi_j(x) = \log(x+1)$

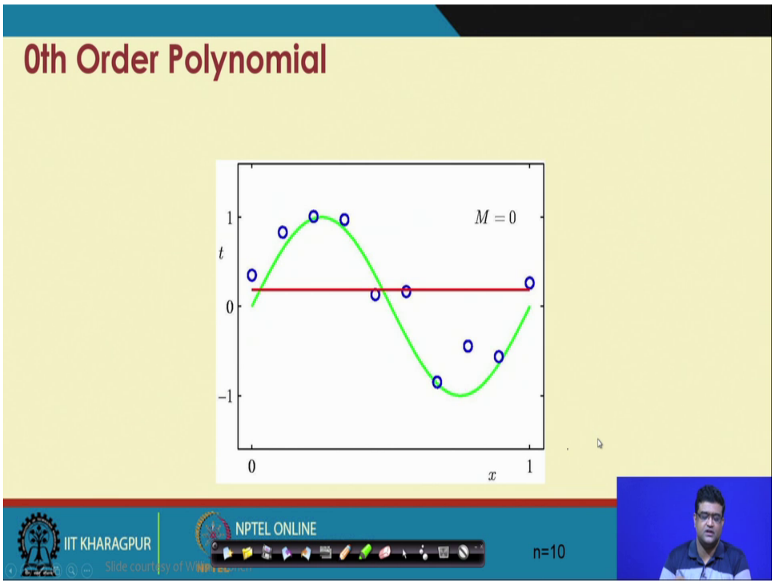
Any function of the input values can be used. The solution for the parameters of the regression remains the same.

IIT KHARAGPUR

NPTEL ONLINE

Now, one can also use polynomial future map. So, one can instead of using a multi variant future map one can take a single number x has input and use all polynomials or Gaussian function or sigmoid function as the as the feature values for each data points. So, this is so sometimes called a feature transformation.

(Refer Slide Time: 25:42)



0th Order Polynomial

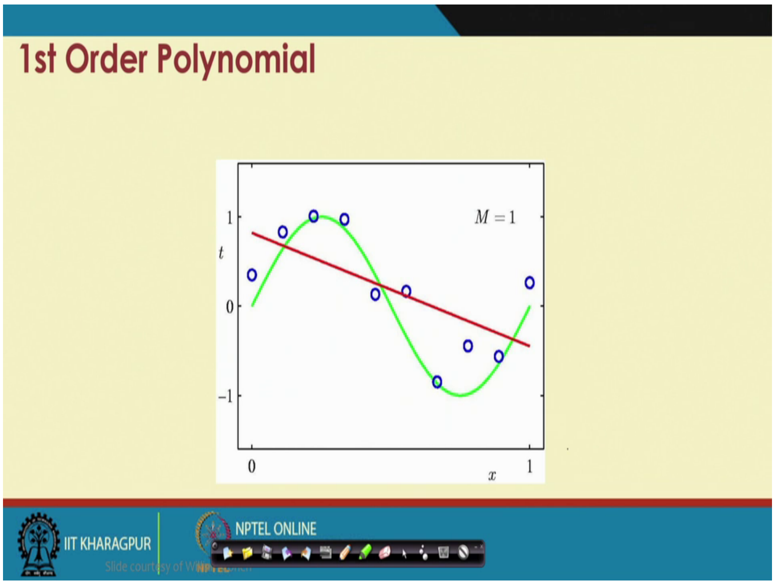
IIT KHARAGPUR

NPTEL ONLINE

n=10

So, if we if we use a such non-linear feature maps, we can get different orders of polynomial regression. So, now, consider this data points where these blue dots are the true data and you want to fit polynomial through this data. So, the red, so this is generated from this sin curve which is the which is the which is the green line, which is the true curve from which the data has been generated and some noise has been added to the data and the red is the 0th order fitted polynomials. So, you can see 0th order fitted polynomial is nothing but a constant value.

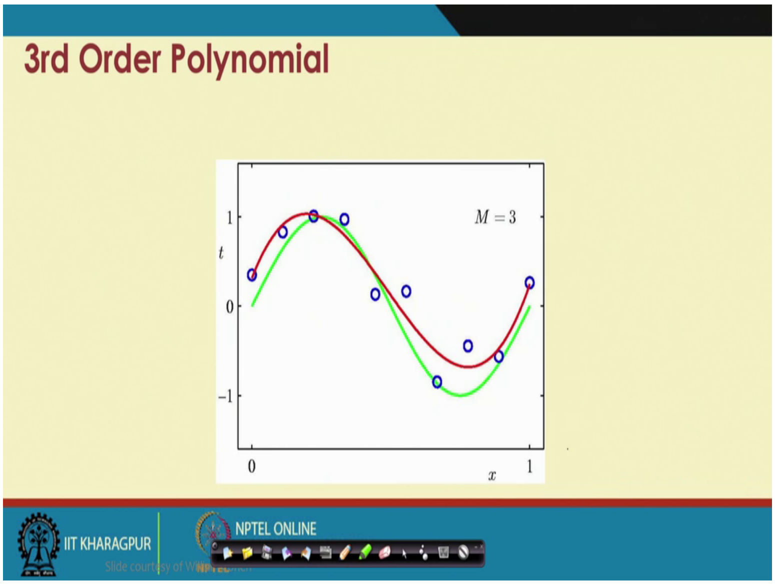
(Refer Slide Time: 26:35)



1st Order Polynomial

Now, this is the first order fitted polynomial which is general line segment in two-dimensional space.

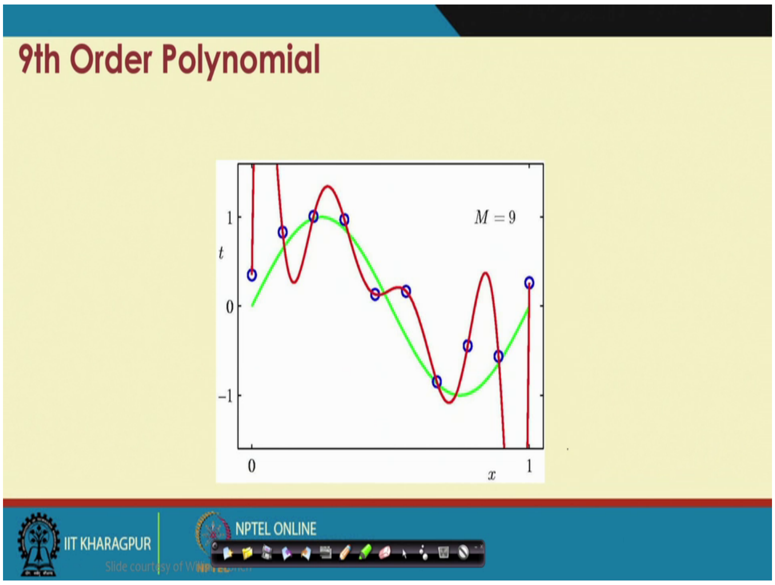
(Refer Slide Time: 26:44)



3rd Order Polynomial

This is the order 3 fitted polynomial which is the red line here.

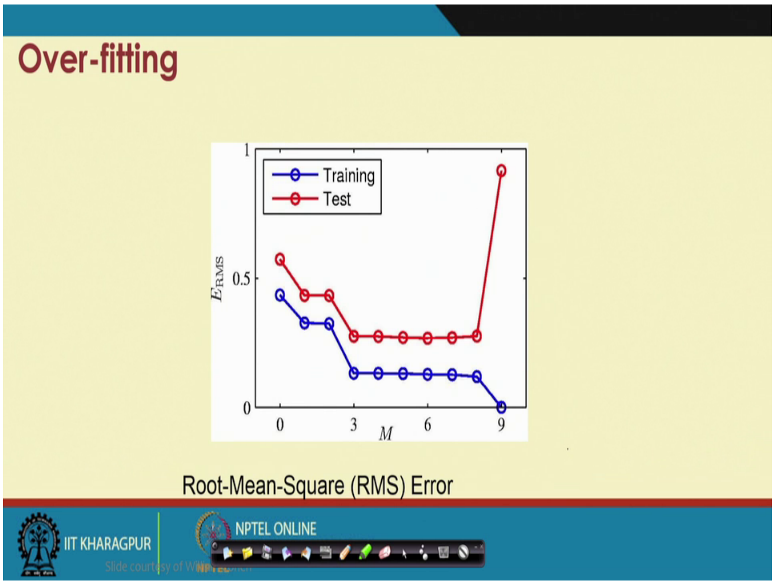
(Refer Slide Time: 26:54)



9th Order Polynomial

And this is the ordered 9 fitted polynomial. So, the red line here is the ordered 9 fitted polynomial. Now, intuitively you can think that the order 3 fitted polynomial was the best fit for the data but how do we know this, so we cannot know this beforehand. So, this order 9 fitted polynomial exhibits the phenomenon called over fitting.

(Refer Slide Time: 27:28)

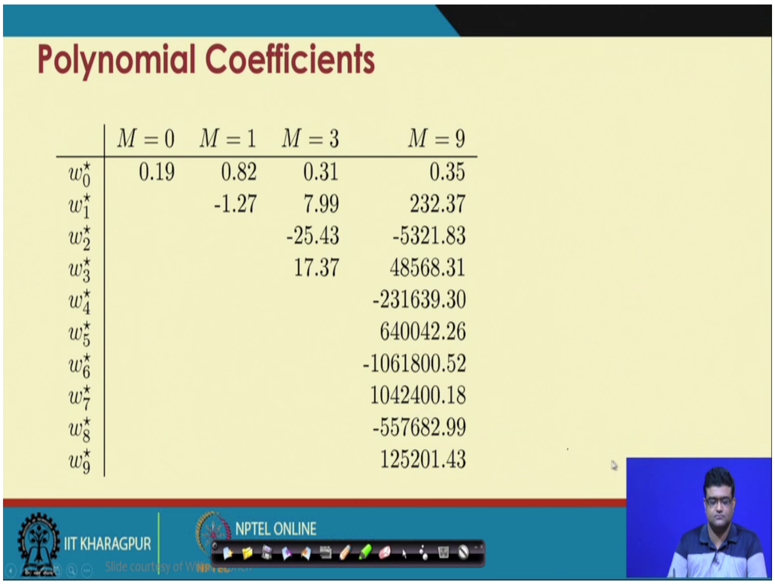


Over-fitting

Root-Mean-Square (RMS) Error

So, how does one know whether there is over fitting or not? So, for example, in this case if I plot the order of the polynomial verses the error on a test data set and a training data set. So, as you can see as you as you increase the number of the degree of the polynomial your training error reduces but the red line which is the test error reduce us to a certain point after which it increases drastically. So, this point is called the over fitting where the even though the training error is decreasing the test error increases.

(Refer Slide Time: 28:16)



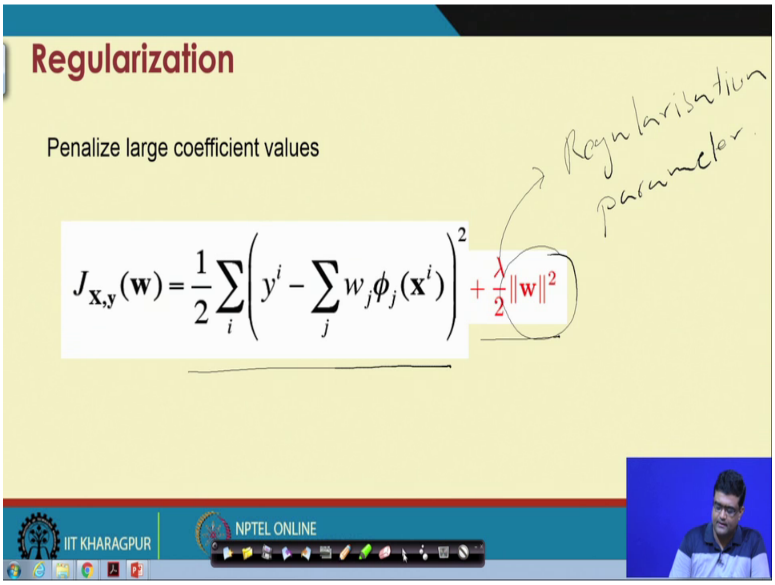
Polynomial Coefficients

	$M=0$	$M=1$	$M=3$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

IIT KHARAGPUR
NPTEL ONLINE

And here what I am reporting is the number of or the values of the betas or the coefficients of the polynomial. And as you can see for ordered 9 the coefficients are really very high in value.

(Refer Slide Time: 28:44)



Regularization

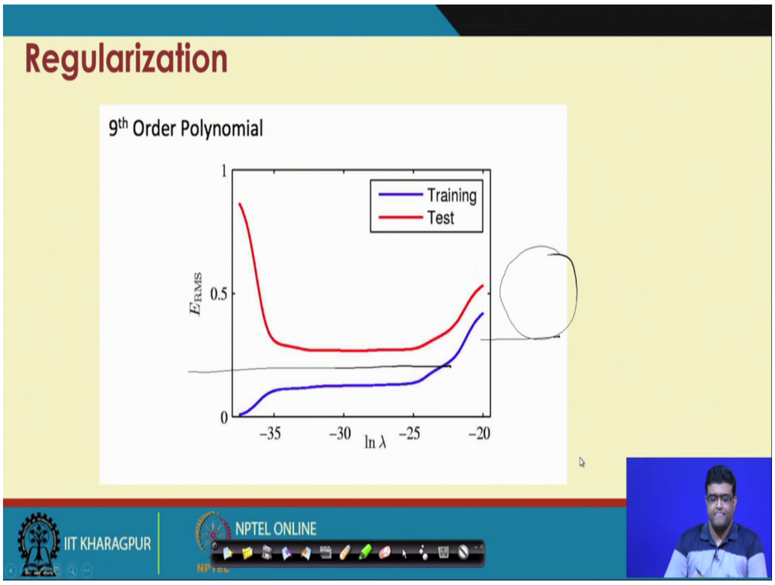
Penalize large coefficient values

$$J_{\mathbf{x},\mathbf{y}}(\mathbf{w}) = \frac{1}{2}\sum_i \left(y^i - \sum_j w_j \phi_j(\mathbf{x}^i)\right)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

Regularisation parameter

So, one way to stop this from happening is to add what is called a regularizer. So, in addition, so in addition to minimizing the squared error which is the first equation we also minimize the length of the parameter vector which is w in this case. So, we will be also minimize the length of the parameter vector.

(Refer Slide Time: 29:15)

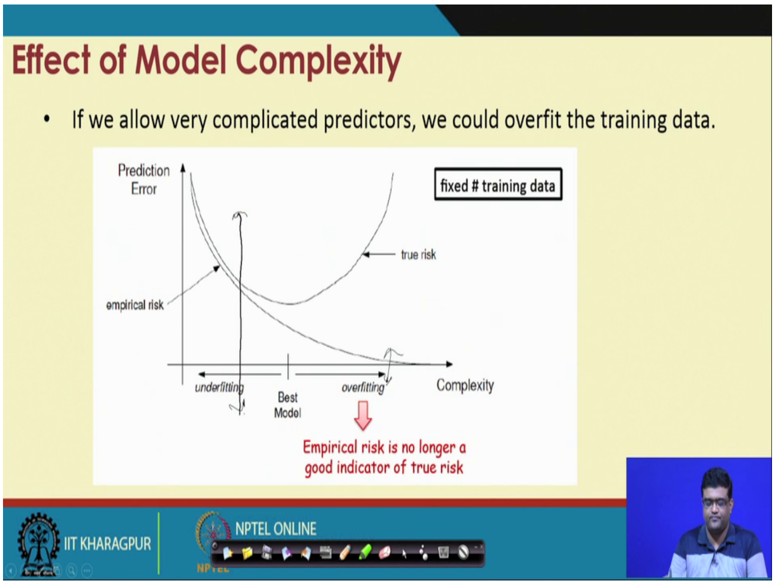


Regularization

And lambda is a parameter which is user controlled, so lambda is also called a regularization parameter. So, lambda is also called a regularization parameter. And if you vary the regularization parameter in log scale what you can see is as the regularization parameter increases the gap between the training and the test error reduces.

So, if the regularization parameter is very low there is still over fitting. So, this is the over fitting, so this here lambda is almost 0. And as the regularization parameter increases the gap between the training and the test error test error reduces and after a while both the training and the test error starts to increase at which point it is the over regularization. So, the best point is where the gap between the training and the test error is minimum and both the training and the test error are very low.

(Refer Slide Time: 30:49)



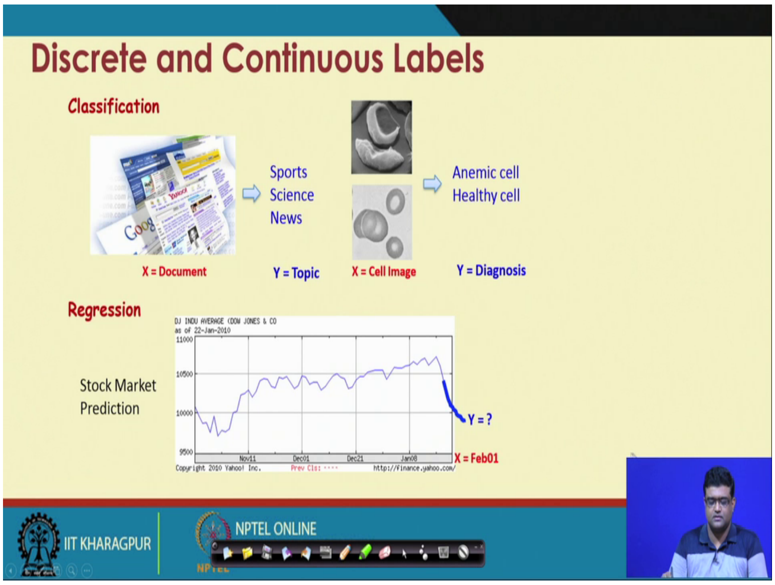
Effect of Model Complexity

- If we allow very complicated predictors, we could overfit the training data.

Empirical risk is no longer a good indicator of true risk

So, next, next we discuss the discrete levels.

(Refer Slide Time: 30:57)



Discrete and Continuous Labels

Classification

Sports
Science
News

X = Document Y = Topic

Anemic cell
Healthy cell

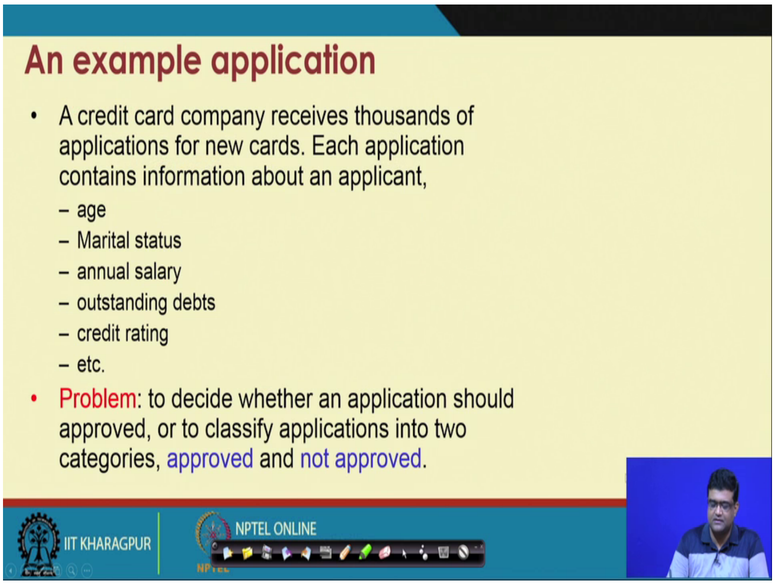
X = Cell Image Y = Diagnosis

Regression

Stock Market Prediction

As we have already described the problem of classification comes when the label Y is discrete.

(Refer Slide Time: 31:14)

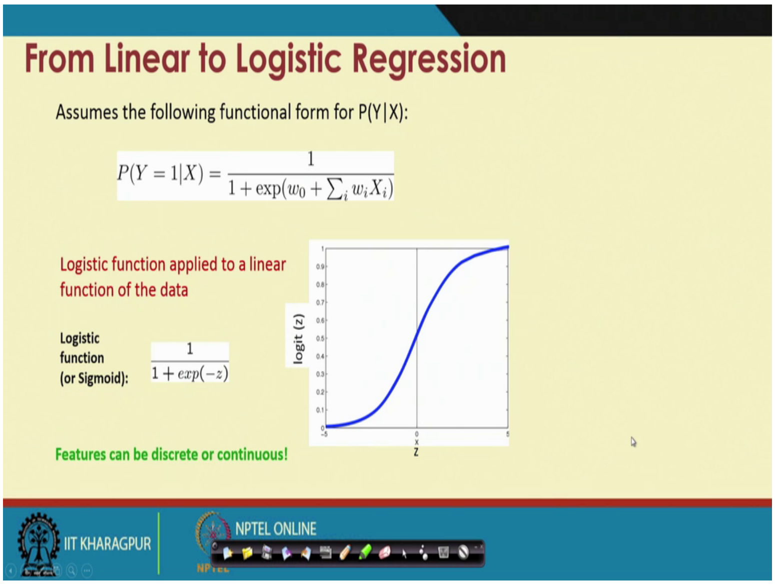


An example application

- A credit card company receives thousands of applications for new cards. Each application contains information about an applicant,
  - age
  - Marital status
  - annual salary
  - outstanding debts
  - credit rating
  - etc.
- Problem: to decide whether an application should approved, or to classify applications into two categories, approved and not approved.

So, for example, for example let us say a credit card company receives loan applications or new credit card applications and it has to classify an application to be either approved or not approved. So, in this case one cannot have a continuous output but one must have discrete values output in this case approved or not approved or in case of spam filtering spam or not spam.

(Refer Slide Time: 31:54)



From Linear to Logistic Regression

Assumes the following functional form for P(Y|X):

$$P(Y=1|X) = \frac{1}{1+\exp(w_0+\sum_i w_i X_i)}$$

Logistic function applied to a linear function of the data

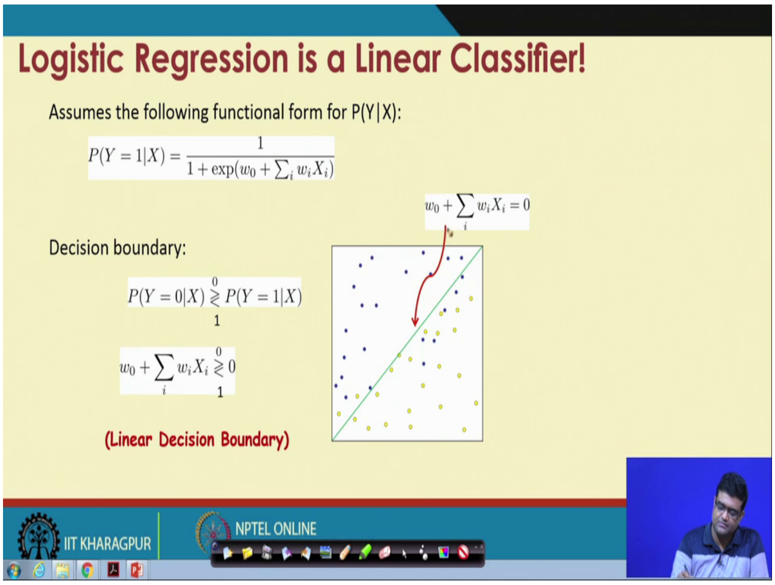
Logistic function (or Sigmoid): $\frac{1}{1+exp(-z)}$

Features can be discrete or continuous!

So, one there are many ways of doing classification. So, one way of performing classification is instead of having a linear equation describe the fittings of the data points. We describe it as a function that computes the probability of Y is equal to 1. So, Y can now only take value plus one or minus 1 and the probability of Y is equal to 1 is given by this sigmoid. So, this function 1 by 1 plus e to the power z is called the sigmoid function or the logistic function and the shape of this is something like this And this z is the linear function which is given here.

So, as you can see if this linear function z is very high then the probability of outputting one is very high, whereas if the linear function is low the probability of outputting one is 0 and in between there is a sharp transition between probability of 0 to 1. So, this is the property of the sigmoid function.

(Refer Slide Time: 33:23)



Logistic Regression is a Linear Classifier!

Assumes the following functional form for P(Y|X):

$$P(Y=1|X) = \frac{1}{1+\exp(w_0+\sum_i w_i X_i)}$$

$$w_0 + \sum_i w_i X_i = 0$$

Decision boundary:

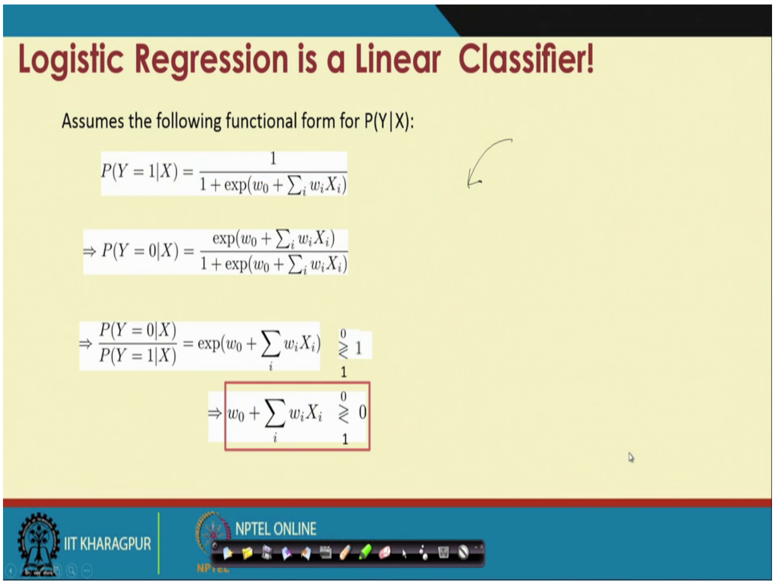
$$P(Y=0|X) \overset{0}{\underset{1}{\gtrless}} P(Y=1|X)$$

$$w_0 + \sum_i w_i X_i \overset{0}{\underset{1}{\gtrless}} 0$$

(Linear Decision Boundary)

So, in other words if you check this discussion boundary of this linear function f of x being equal to 0, you see that on one side of it which is the greater than 0 side lies the lies all the data points which for which probability of Y is 1 and on the other side lies all the data points for which probability of Y is equal to 1 is 0.

(Refer Slide Time: 34:09)



Logistic Regression is a Linear Classifier!

Assumes the following functional form for P(Y|X):

$$P(Y=1|X) = \frac{1}{1+\exp(w_0+\sum_i w_i X_i)}$$

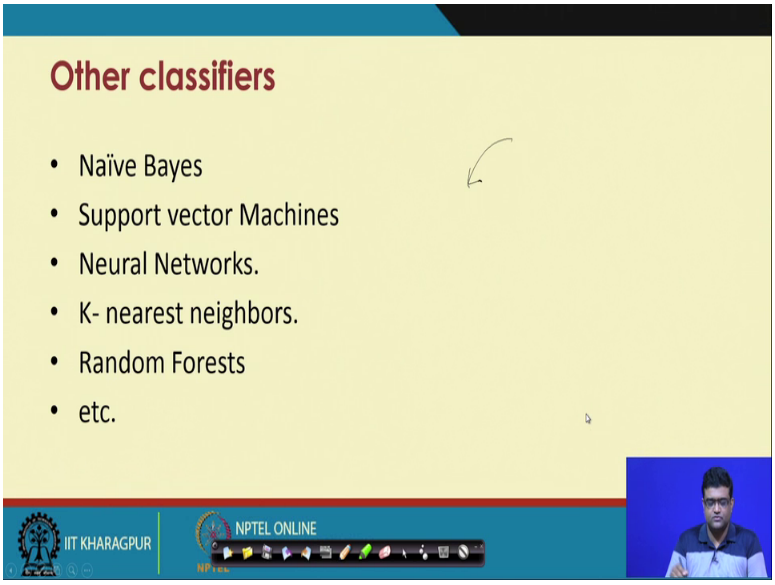
$$\Rightarrow P(Y=0|X) = \frac{\exp(w_0+\sum_i w_i X_i)}{1+\exp(w_0+\sum_i w_i X_i)}$$

$$\Rightarrow \frac{P(Y=0|X)}{P(Y=1|X)} = \exp(w_0 + \sum_i w_i X_i) \overset{0}{\underset{1}{\gtrless}} 1$$

$$\Rightarrow \boxed{w_0 + \sum_i w_i X_i \overset{0}{\underset{1}{\gtrless}} 0}$$

So, without going into this, so this is one way of designing classifiers which use linear functions of features as the input.

(Refer Slide Time: 34:15)

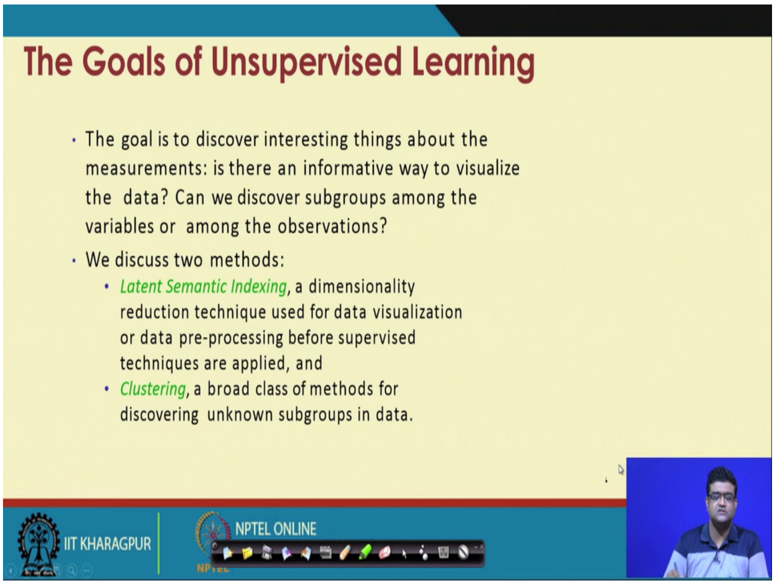


Other classifiers

- Naïve Bayes
- Support vector Machines
- Neural Networks.
- K- nearest neighbors.
- Random Forests
- etc.

There are many other ways of designing classifiers. So, for example, Naive Bayes is one way of designing classifier support vector machines is another way of designing classifier which also sometimes use linear functions and there can also be non-linear functions. Neural networks are another way of designing classifiers which support a wide variety of non-linear functions. The K-nearest neighbor and random forest are also ways of designing classifiers.

(Refer Slide Time: 35:08)



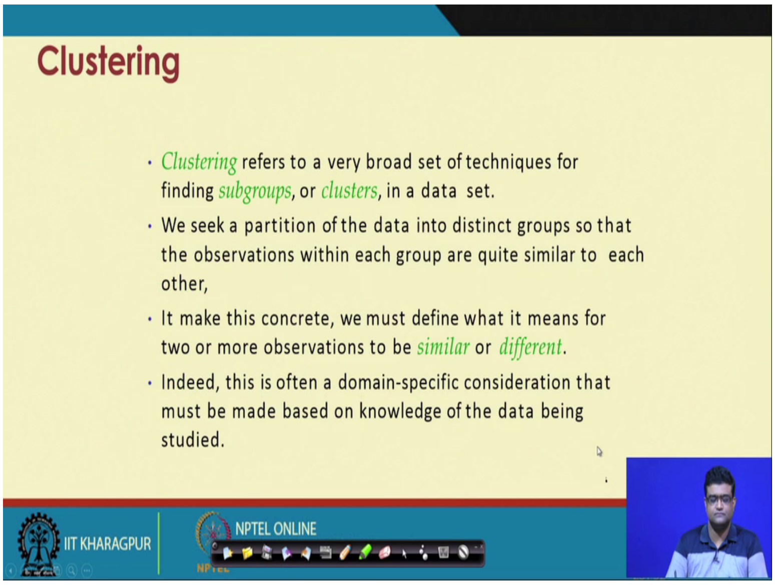
The Goals of Unsupervised Learning

- The goal is to discover interesting things about the measurements: is there an informative way to visualize the data? Can we discover subgroups among the variables or among the observations?
- We discuss two methods:
  - *Latent Semantic Indexing*, a dimensionality reduction technique used for data visualization or data pre-processing before supervised techniques are applied, and
  - *Clustering*, a broad class of methods for discovering unknown subgroups in data.

Now, we move on to unsupervised learning. So, the goal of unsupervised learning is to discover interesting features of the data. So, two main methods will be discussed, there are many unsupervised learning techniques and problems. So, two main unsupervised learning problems which we will use in this course are dimensionality reduction or also called latent semantic indexing which we will use in this course and the other is clustering.

So, we have covered the dimensionality reduction in another lecture. So, we will cover the clustering here.

(Refer Slide Time: 36:01)

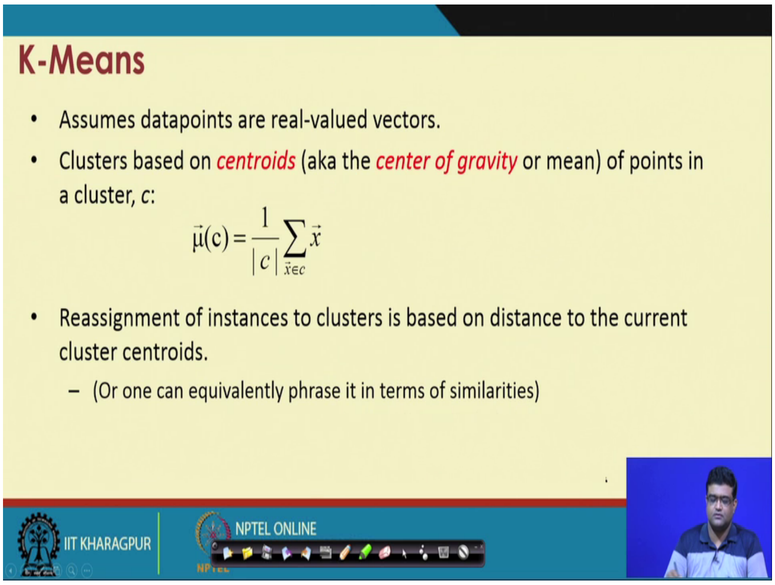


Clustering

- *Clustering* refers to a very broad set of techniques for finding *subgroups*, or *clusters*, in a data set.
- We seek a partition of the data into distinct groups so that the observations within each group are quite similar to each other,
- It make this concrete, we must define what it means for two or more observations to be *similar* or *different*.
- Indeed, this is often a domain-specific consideration that must be made based on knowledge of the data being studied.

So, clustering refers to very broad set of techniques for finding sub groups of or clusters in a data set. So, we seek to partition the data into discrete groups or sub groups such that the points within a partition are nearby in the space where this data points exists. And they are somehow similar which is to say that they are nearby in the space or, and point in different groups are somehow different or far away from each other.

(Refer Slide Time: 36:48)



K-Means

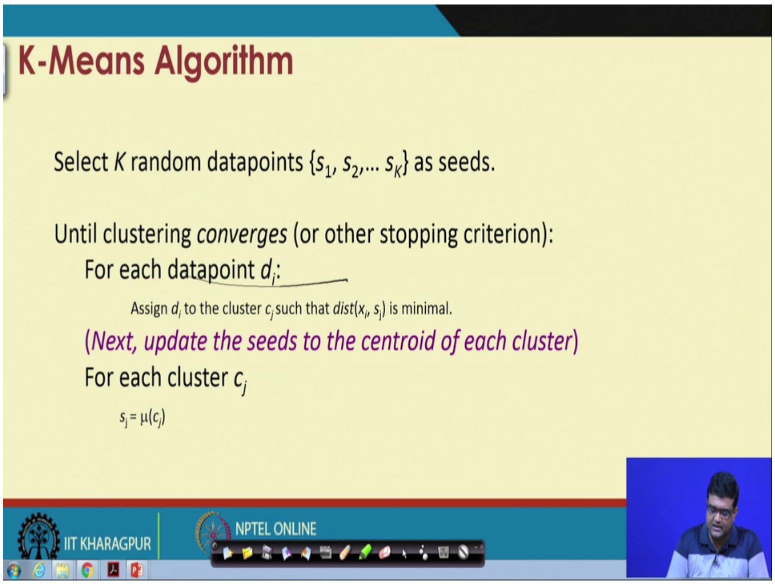
- Assumes datapoints are real-valued vectors.
- Clusters based on *centroids* (aka the *center of gravity* or mean) of points in a cluster, *c*:

$$\vec{\mu}(c) = \frac{1}{|c|}\sum_{\vec{x}\in c}\vec{x}$$

- Reassignment of instances to clusters is based on distance to the current cluster centroids.
  - (Or one can equivalently phrase it in terms of similarities)

So, a popular method of clustering is called the K-Means clustering. So, it assumes that the data data points are real valued vectors. So, let x be the real value data points, then each cluster in a K-Mean. So, we also assume that there are k clusters which is free defined and each cluster see in K-Means cluster is represented by centroid which is represented here by this mu and which is the centroids of the data points that belongs to this cluster.

(Refer Slide Time: 37:53)



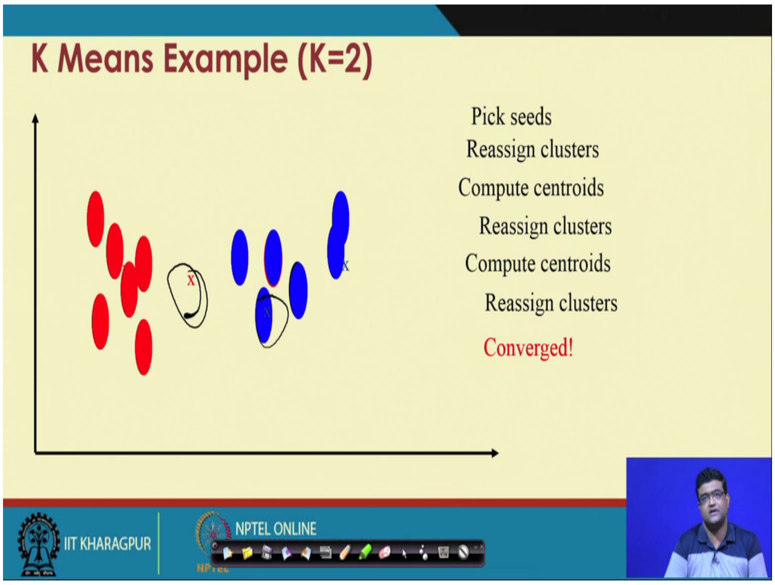
K-Means Algorithm

```
Select K random datapoints {$s_1, s_2, \ldots s_K$} as seeds.

Until clustering converges (or other stopping criterion):
    For each datapoint $d_i$:
        Assign $d_i$ to the cluster $c_j$ such that $dist(x_i, s_j)$ is minimal.
    (Next, update the seeds to the centroid of each cluster)
    For each cluster $c_j$
        $s_j = \mu(c_j)$
```

So, what you can see is that the centroid is the location or the representation of representative of cluster which minimizes the yeah. So, the K-Means clustering algorithm words as follows. ah

(Refer Slide Time: 37:58)



K Means Example (K=2)

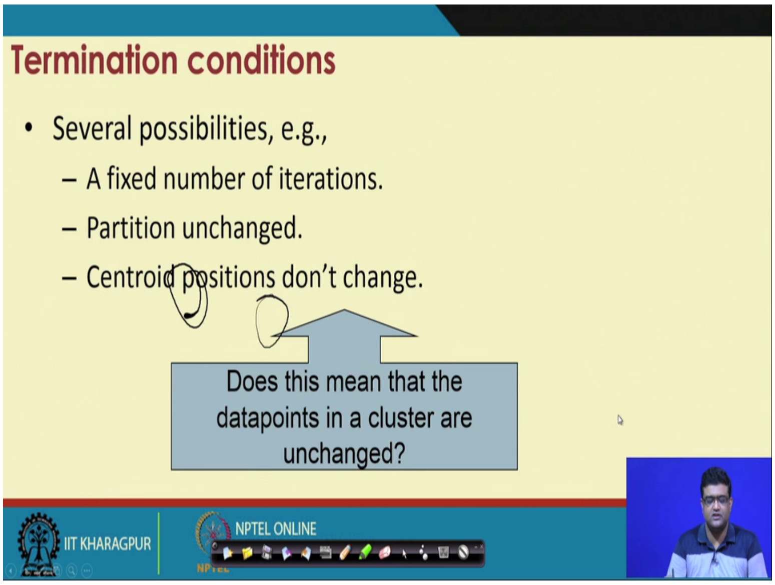
Pick seeds
Reassign clusters
Compute centroids
Reassign clusters
Compute centroids
Reassign clusters
Converged!

So, first given the set of data points we select K random seeds or we select K random data points has seeds and then we iterate as follows. So, for each data point we assign that data point to its nearest cluster, and once we have done this for each data point for each cluster we recalculate the centroid of that cluster and we assign the seed of the cluster or the centroid of the cluster as the recomputed center.

So, this is (Refer Time: 39:07) of that of the K-Means clustering algorithm. So, you have the data point. So, first you pick the seeds. So, we went to cluster into two clusters, red and blue. So, once you have computed the seeds you assign the closest data points to or each data point to the closest seed. So, this is the assignment in this particular case. Now, you compute the recompute the centroid of each, so there the centroid of the red is somewhere here; so the centroid of the red is now somewhere here and the centroid of the blue is now here.

And you go to the next iteration where you now relable the data points as once that are closest to their centroid and then you recompute the centroid. And you go ahead like this and until any more reassignment has changed. So, at that point in time K-Means clustering algorithm is said to have converged.

(Refer Slide Time: 40:20)

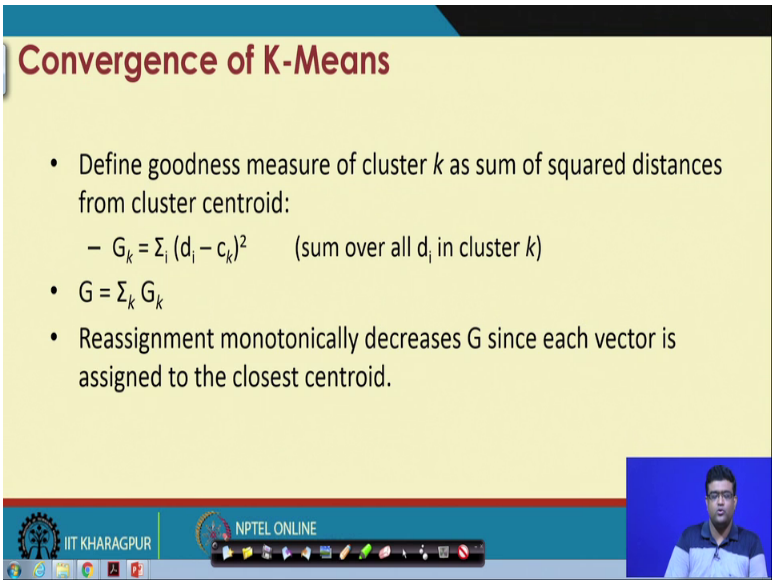


Termination conditions

- Several possibilities, e.g.,
  - A fixed number of iterations.
  - Partition unchanged.
  - Centroid positions don't change.

Does this mean that the datapoints in a cluster are unchanged?

(Refer Slide Time: 40:23)

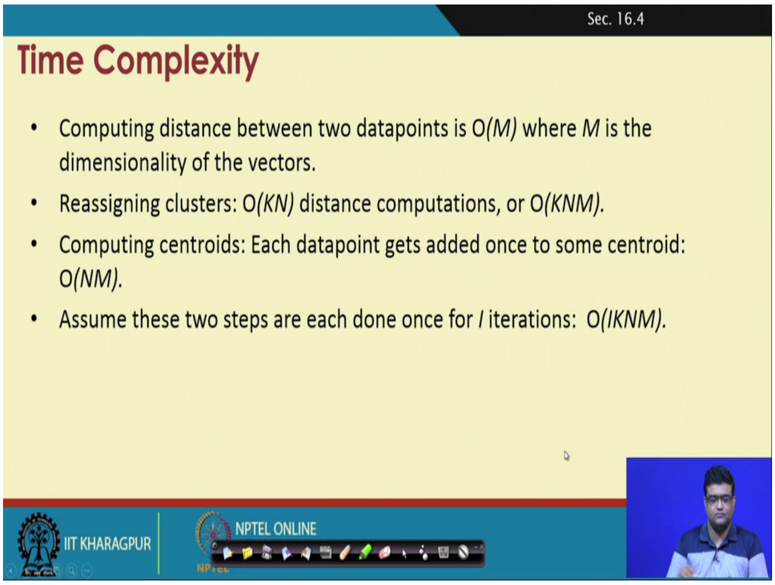


Convergence of K-Means

- Define goodness measure of cluster *k* as sum of squared distances from cluster centroid:
  - $G_k = \Sigma_i (d_i - c_k)^2$ (sum over all d_i in cluster k)
- $G = \Sigma_k G_k$
- Reassignment monotonically decreases G since each vector is assigned to the closest centroid.

And you can show that for any input indeed K-Means clustering will converge.

(Refer Slide Time: 40:39)



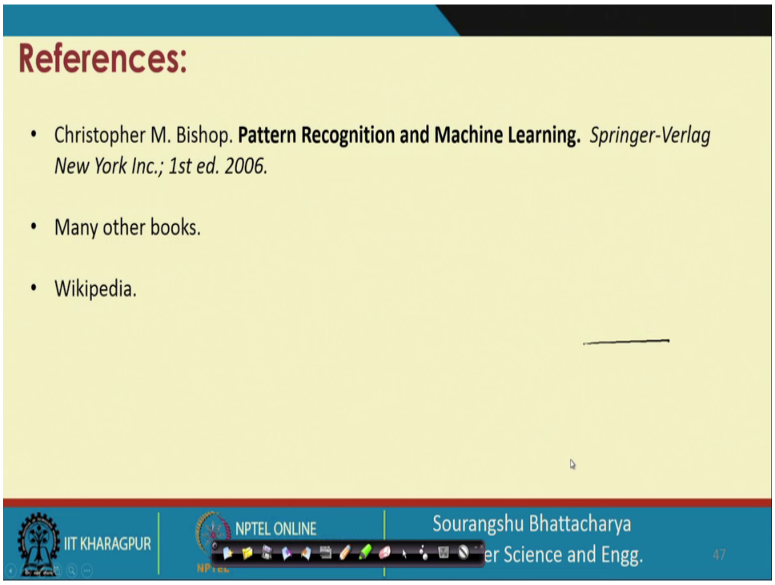
Sec. 16.4

Time Complexity

- Computing distance between two datapoints is O(*M*) where *M* is the dimensionality of the vectors.
- Reassigning clusters: O(*KN*) distance computations, or O(*KNM*).
- Computing centroids: Each datapoint gets added once to some centroid: O(*NM*).
- Assume these two steps are each done once for *I* iterations: O(*IKNM*).

And the time complexity of K-Means clustering algorithm is if you have to run it for I iterations is order of IKM.

(Refer Slide Time: 41:02)



References:

- Christopher M. Bishop. **Pattern Recognition and Machine Learning.** *Springer-Verlag New York Inc.; 1st ed. 2006.*
- Many other books.
- Wikipedia.

So, these are the references for this particular lecture. So, most of the content is you can find in any machine learning book, namely in particular the book by Christopher Bishop. And also you can find most of the content in Wikipedia.

Thank you.