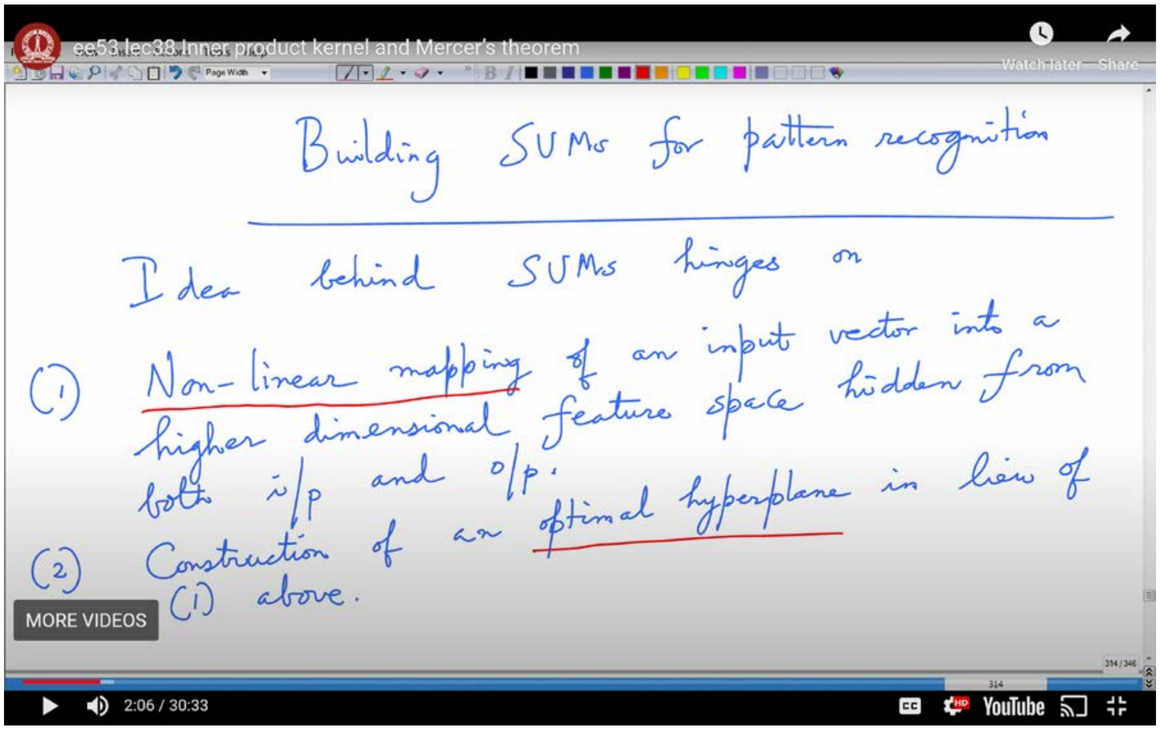
Neural Networks for Signal Processing-I

Prof. Shayan Srinivasa Garani
Department of Electronic System Engineering
Indian Institute of Science, Bengaluru

Lecture – 38
Inner Product Kernel and Mercer’s Theorem

Continuing our discussion on support vector machines (SVMs), let's explore how SVMs can be applied to pattern recognition problems. To delve into this topic, we'll need to understand the concept of inner product kernels, which involves Mercer's theorem among other concepts. So, let's focus on these aspects in today's lecture.

(Refer Slide Time: 02:06)



Building SVMs for pattern recognition

Idea behind SVMs hinges on

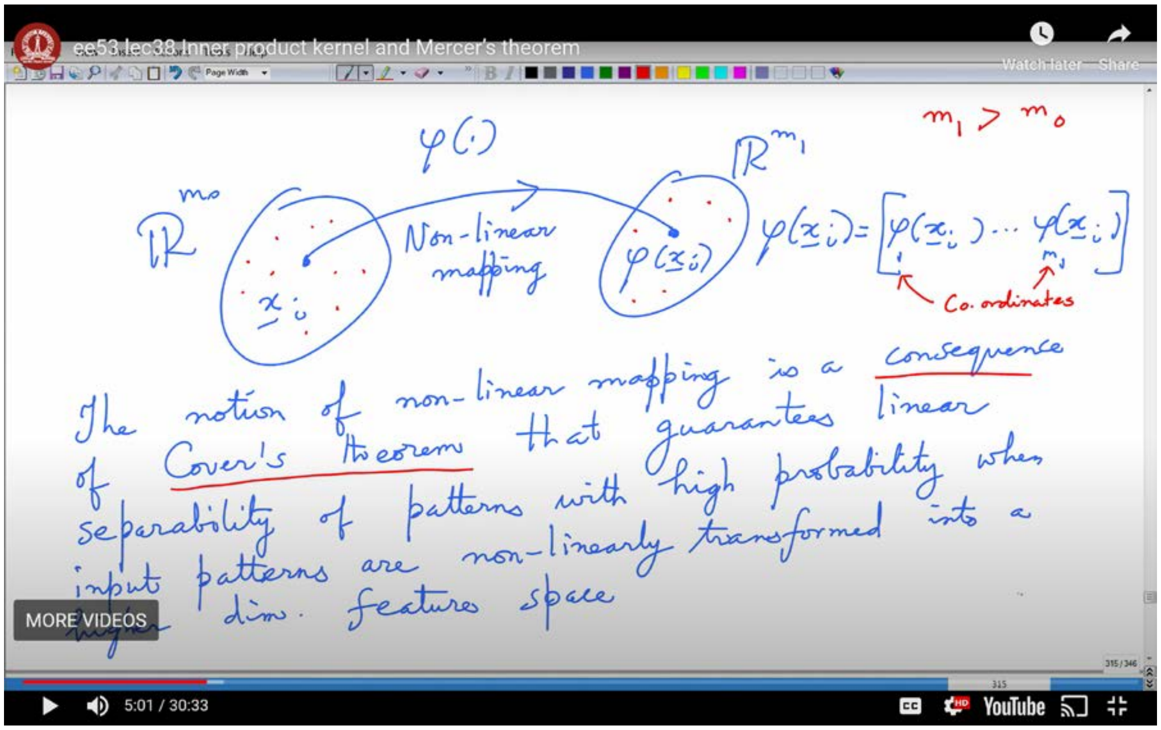
(1) Non-linear mapping of an input vector into a higher dimensional feature space hidden from both i/p and o/p.

(2) Construction of an optimal hyperplane in lieu of (1) above.

The fundamental idea behind SVMs rests on two crucial elements. First, there's the non-linear mapping of an input vector into a higher-dimensional feature space, a transformation that is hidden from both the input and the output. Second, we need to construct an optimal hyperplane within this new feature space to achieve the desired pattern separability.

To break this down, we start by taking an input vector from its original space and lifting it into a higher-dimensional feature space through a non-linear transformation. Once in this higher-dimensional space, we introduce the notion of an optimal hyperplane to address the pattern separability problem.

(Refer Slide Time: 05:01)


ee53 lec38 Inner product kernel and Mercer's theorem
Watch later
Share
$m_1 > m_0$
$\varphi(\cdot)$
$\mathbb{R}^{m_0}$
$\mathbb{R}^{m_1}$
Non-linear mapping
$\underline{x}_i$
$\varphi(\underline{x}_i)$
$\varphi(\underline{x}_i) = [\varphi_1(\underline{x}_i) \dots \varphi_{m_1}(\underline{x}_i)]$
Co-ordinates
MORE VIDEOS
5:01 / 30:33

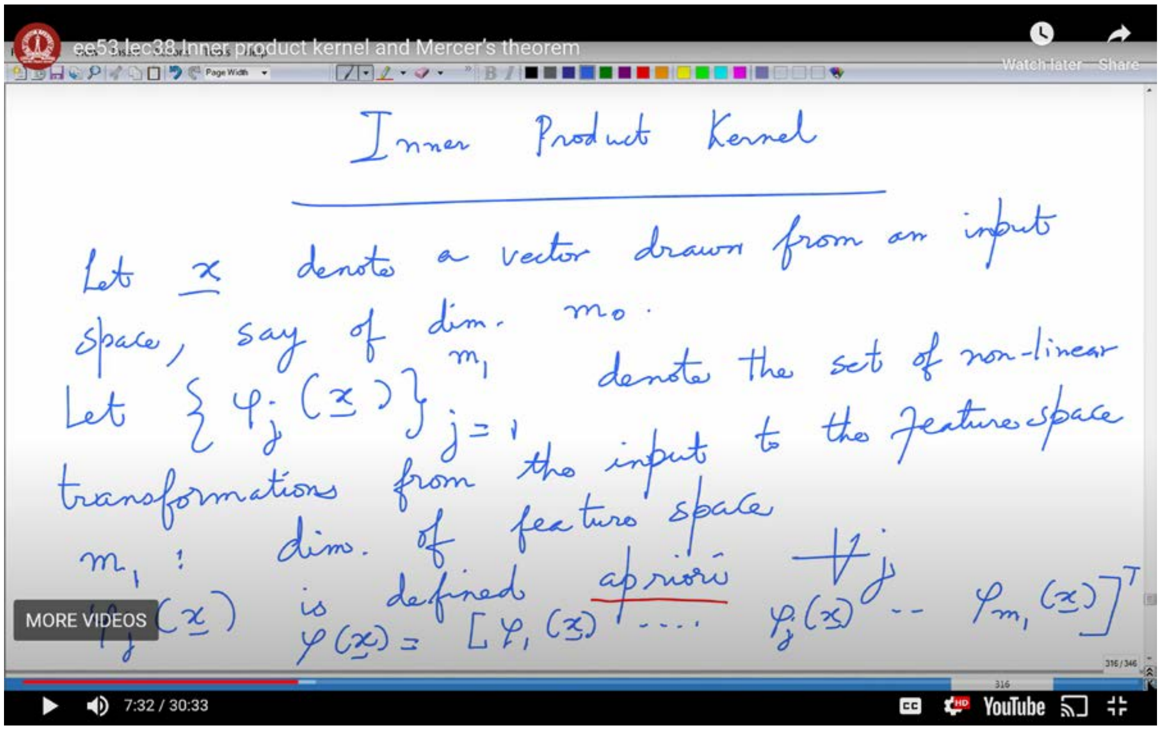
The notion of non-linear mapping is a consequence of Cover's theorem that guarantees linear separability of patterns with high probability when input patterns are non-linearly transformed into a high dim. features space

Visualize it this way: we have a set of points in an M_0-dimensional input space. These points are then non-linearly mapped to a corresponding set of points in an M_1-dimensional space, where typically M_1 is greater than M_0. While it's possible to have M_1 equal to M_0, we generally work with the inequality $M_1 > M_0$, though non-linear transformations where $M_1 = M_0$ are also valid.

This non-linear mapping, represented as $\varphi(x_i)$, can be expressed in terms of coordinates within the M_1-dimensional feature space as $\varphi_1(x_i), \varphi_2(x_i), \dots, \varphi_{M_1}(x_i)$. The concept of non-linear mapping is rooted in Cover's theorem, which we discussed in an earlier module.

Cover's theorem assures us that patterns are linearly separable with high probability when the input patterns are non-linearly transformed into a higher-dimensional feature space.

(Refer Slide Time: 07:32)



Inner Product Kernel

Let $\underline{x}$ denote a vector drawn from an input space, say of dim. m_0.

Let $\{\varphi_j(\underline{x})\}_{j=1}^{m_1}$ denote the set of non-linear transformations from the input to the feature space

m_1 : dim. of feature space

$\varphi_j(\underline{x})$ is defined apriori $\forall j$

$\varphi(\underline{x}) = [\varphi_1(\underline{x}) \ldots \varphi_j(\underline{x}) -- \varphi_{m_1}(\underline{x})]^T$

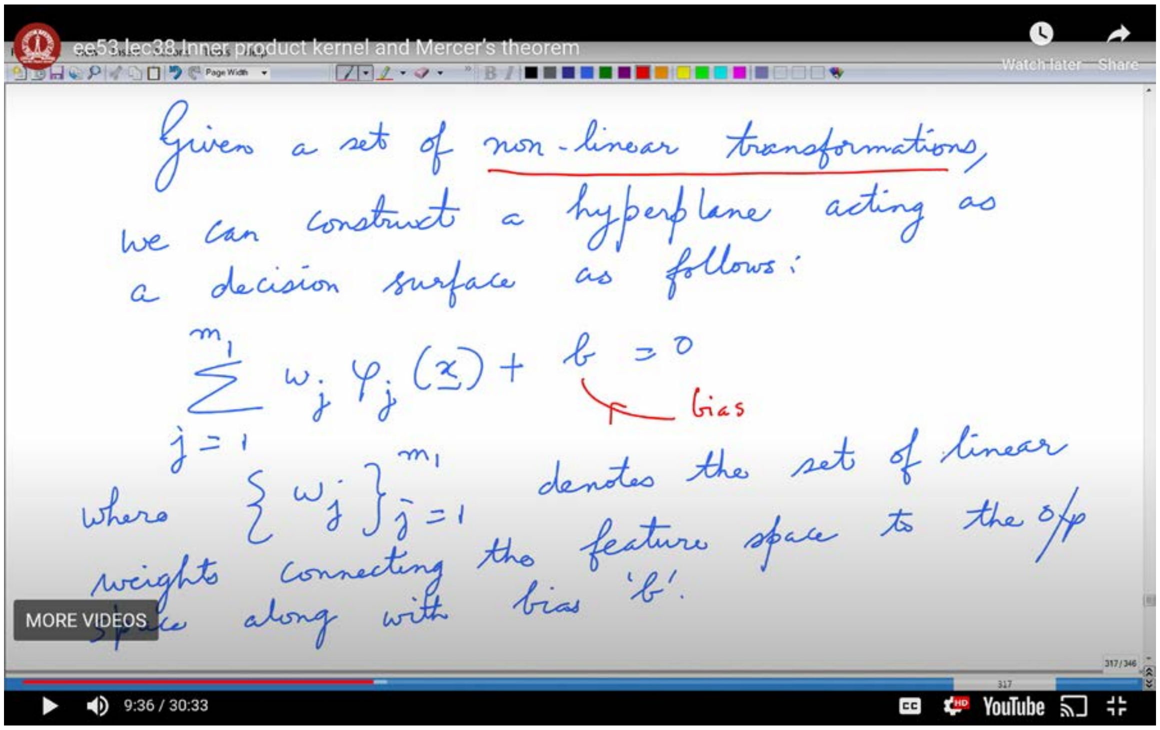
This principle from Cover's theorem, when combined with the SVM design philosophy, where optimal hyperplanes are constructed within the feature space, provides a powerful method for solving pattern separability problems. Essentially, by lifting the input data non-linearly into a feature space and applying the SVM within this space, we can effectively address the pattern recognition tasks at hand.

To implement this, we introduce the concept of an inner product kernel. This kernel allows us to work with the feature space indirectly, without explicitly computing the high-dimensional mappings. The inner product kernel plays a pivotal role in the SVM framework, enabling us to handle complex, non-linear separability problems with elegance and efficiency.

Let x represent a vector drawn from an input space of dimension M_0. Now, let $\varphi_j(x)$ for j = 1 to M_1 denote a set of non-linear transformations from the input space to the feature space.

This concept is crucial. In this context, I have a feature vector in an M_1-dimensional space, where each coordinate, indexed by j from 1 to M_1, represents a non-linear mapping. Each of these mappings operates on the vector x, though the mappings can differ for each coordinate. Essentially, we have different functions at each coordinate j, and $\varphi_j(x)$ is defined a priori.

(Refer Slide Time: 09:36)



Given a set of non-linear transformations, we can construct a hyperplane acting as a decision surface as follows:

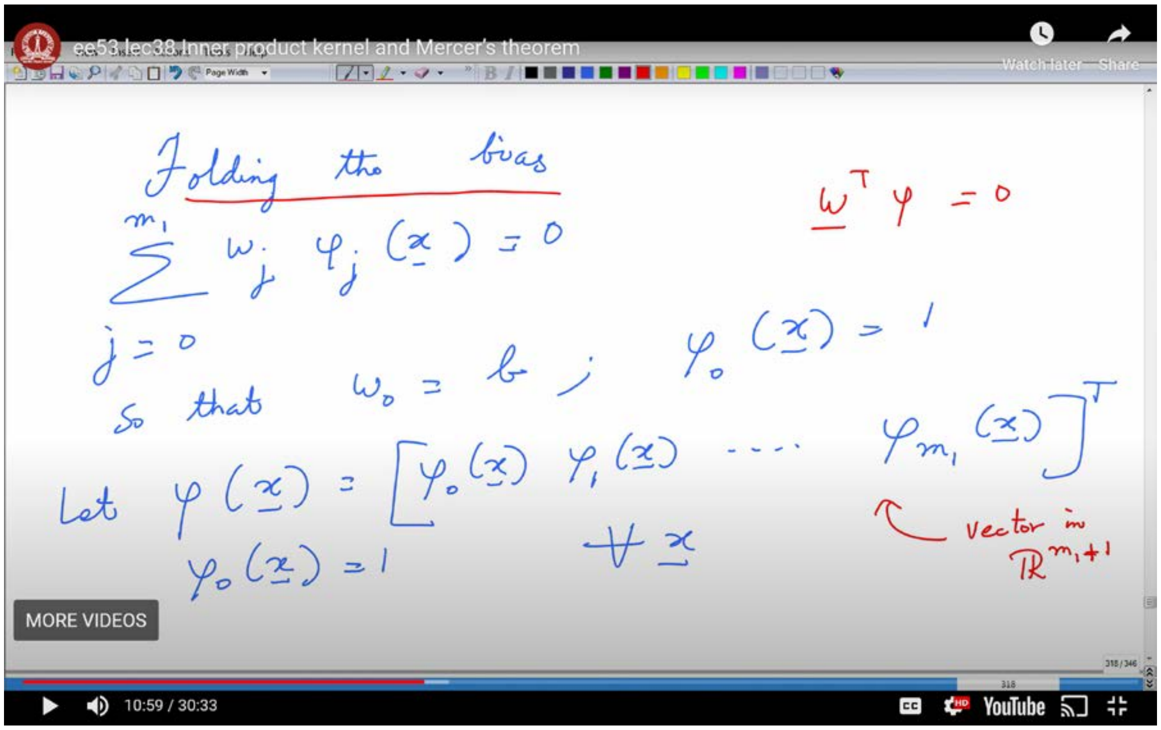
$$\sum_{j=1}^{m_1} w_j \varphi_j(\underline{x}) + b = 0$$

where $\{w_j\}_{j=1}^{m_1}$ denotes the set of linear weights connecting the feature space to the o/p space along with bias 'b'.

This may become clearer when we explore examples of inner product kernels and the corresponding feature spaces, particularly in the context of solving the XOR problem using SVMs. However, in more abstract terms, you can think of $\varphi(x)$ as a vector in the M_1-dimensional feature space, with coordinates $\varphi_1(x)$ through $\varphi_{M_1}(x)$. Each coordinate is a scalar quantity, so $\varphi(x)$ itself is a vector in the feature space, a critical detail to grasp.

While I have defined this feature vector, I have yet to link it to the kernel. We'll build on this idea as we progress. Given a set of non-linear transformations, our goal is to construct a hyperplane that can serve as a decision surface. Although I haven't introduced the kernel concept yet, we'll get into that when discussing Mercer's theorem. For now, imagine that from the kernel, we obtain this feature vector in the M1-dimensional feature space. Given these non-linear transformations, and aiming for an optimal hyperplane for linear separability based on the feature vectors derived from the non-linear transformation of the input vector x, we want to construct this hyperplane in the feature space.

(Refer Slide Time: 10:59)



Folding the bias

$\sum_{j=0}^{m_1} w_j \varphi_j(\underline{x}) = 0$

$\underline{w}^T \varphi = 0$

So that $w_0 = b$; $\varphi_0(\underline{x}) = 1$

Let $\varphi(\underline{x}) = [\varphi_0(\underline{x}) \ \varphi_1(\underline{x}) \ \dots \ \varphi_{m_1}(\underline{x})]^T$

vector in $\mathbb{R}^{m_1+1}$

$\varphi_0(\underline{x}) = 1 \quad \forall \underline{x}$

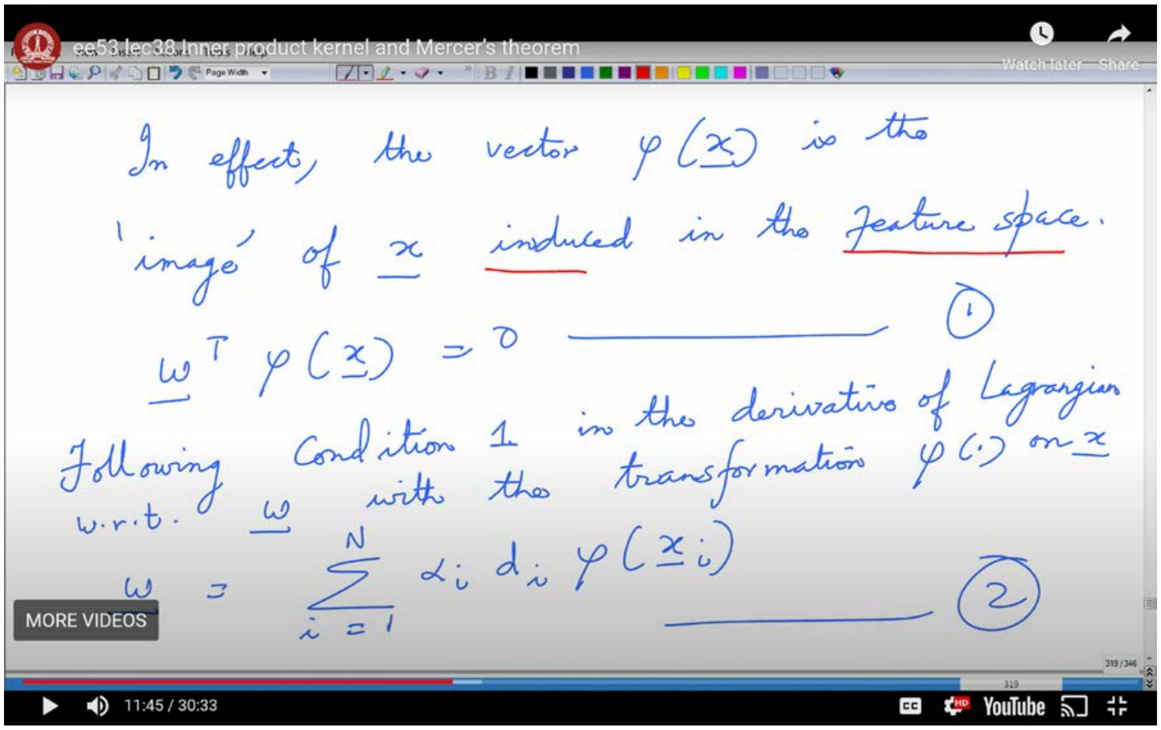
The equation for the hyperplane is given by:

$$\sum_{j=1}^{M_1} w_j \varphi_j(x) + \text{bias} = 0$$

Here, w_j represents the components of the weight vector w, running from w_1 to w_{M_1}, and the bias term is included as well. This equation is similar to what we've encountered earlier when discussing multi-layer perceptrons (MLPs), among other topics.

To simplify, we can express this equation in a more compact form by incorporating the bias into the weight vector.

(Refer Slide Time: 11:45)



In effect, the vector $\varphi(\underline{x})$ is the 'image' of $\underline{x}$ induced in the feature space.

$$\underline{w}^T \varphi(\underline{x}) = 0 \text{ ——— } ①$$

Following Condition 1 in the derivative of Lagrangian w.r.t. $\underline{w}$ with the transformation $\varphi(\cdot)$ on $\underline{x}$

$$w = \sum_{i=1}^{N} \alpha_i d_i \varphi(\underline{x}_i) \text{ ——— } ②$$

By doing so, we rewrite the equation as:

$$\sum_{j=0}^{M_1} W_j \varphi_j(x) = 0$$

which can be expressed as:

$$W^T \varphi(x) = 0$$

or, equivalently:

$$\varphi(x)^T W = 0$$

depending on which form is more convenient. By folding the bias into this linear equation, we set w_0 equal to the bias, and $\varphi_0(x)$ equal to 1. With this augmentation, the feature vector $\varphi(x)$ becomes:

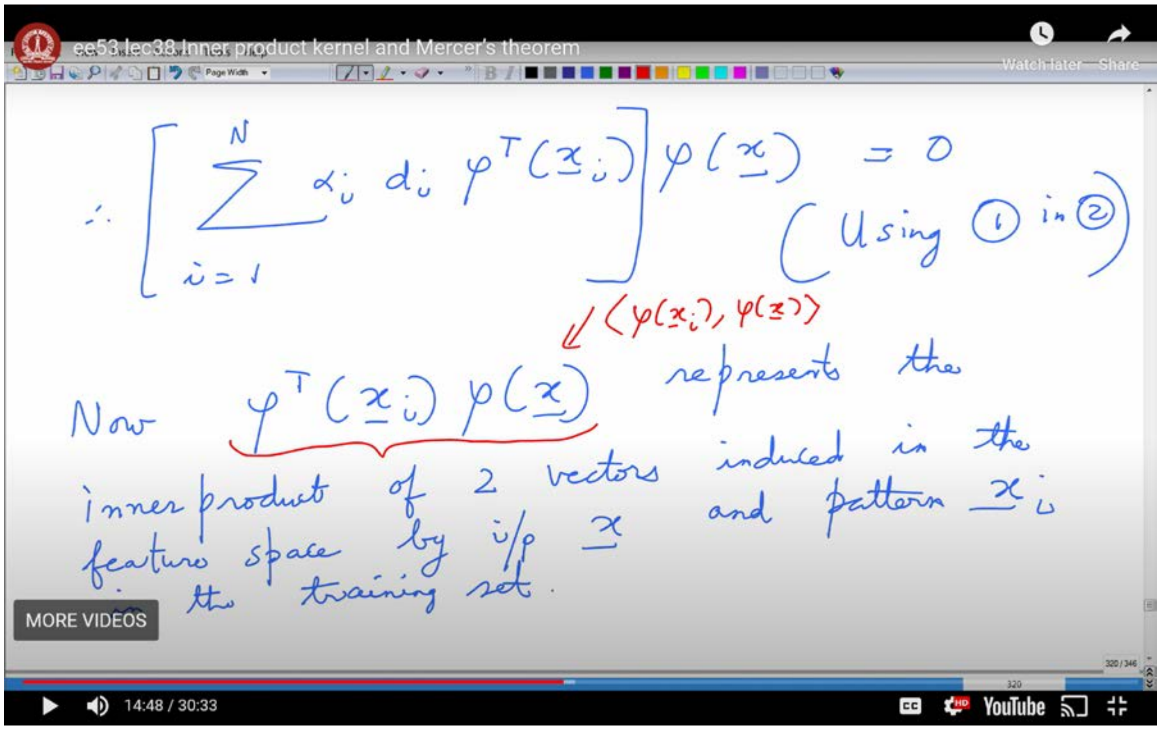
$$\varphi(x) = [\varphi_0(x), \varphi_1(x), \dots, \varphi_{M_1}(x)]$$

This is now a vector in R^{M_1+1}, reflecting the inclusion of the coordinate corresponding to the bias, where $\varphi_0(x) = 1$ ensures the validity of the equation.

In essence, the vector φ(x) represents the image of x induced in the feature space. Consequently, we need to determine a hyperplane w such that:

$$W^T\varphi(x) = 0$$

(Refer Slide Time: 14:48)



$$\therefore \left[\sum_{i=1}^{N} \alpha_i d_i \varphi^T(\underline{x}_i)\right]\varphi(\underline{x}) = 0$$

(Using ① in ②)

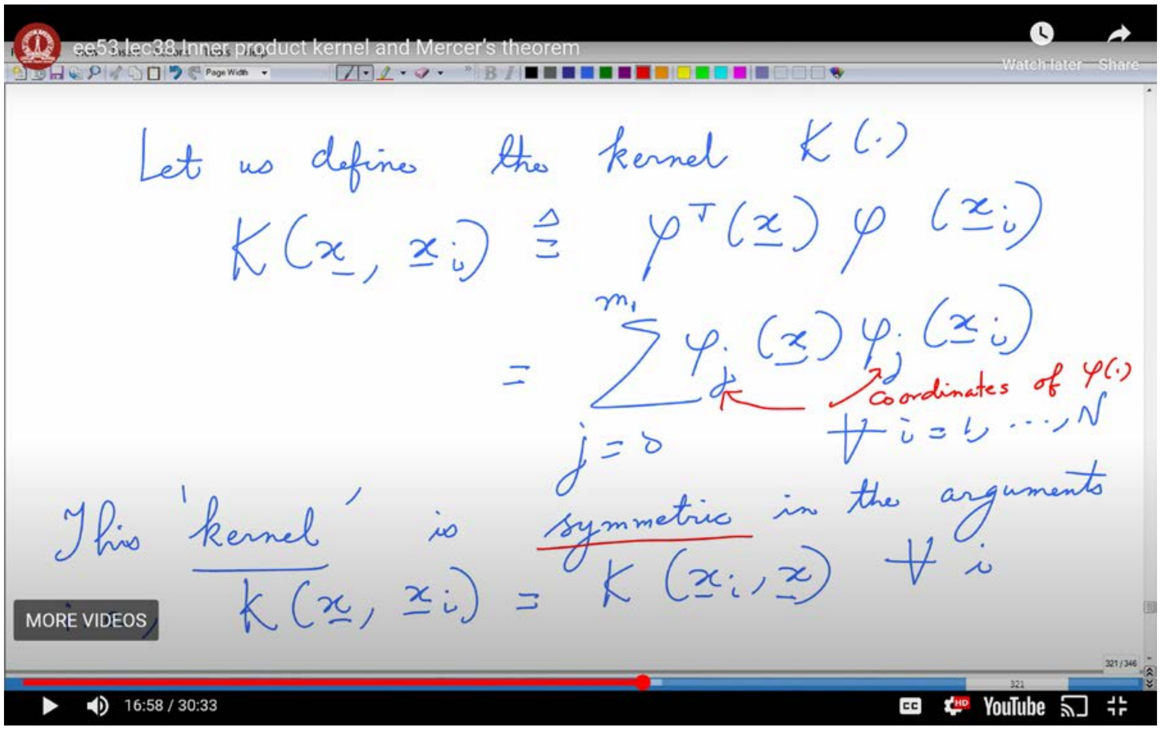
$\langle \varphi(\underline{x}_i), \varphi(\underline{x}) \rangle$

Now $\underbrace{\varphi^T(\underline{x}_i)\,\varphi(\underline{x})}$ represents the inner product of 2 vectors induced in the feature space by i/p $\underline{x}$ and pattern $\underline{x}_i$ in the training set.

In other words, the inner product of the weight vector W and the feature vector φ(x) must be zero, this is the equation of the hyperplane. We observed a similar condition in the derivation of the SVM for linearly separable patterns, where we formulated the Lagrangian and took the partial derivative with respect to the weight vector, setting it equal to zero for optimality.

So, we arrived at this equation through matrix calculus. By differentiating the relevant matrices, we found that w is equal to $\sum_{i=1}^{n} \alpha_i d_i x_i$. However, in this context, x_i is not the original input vector but rather a vector in the feature space. Therefore, when dealing with problems that involve non-linear mapping, x_i must be replaced by $\varphi(x_i)$. This is a key detail.

(Refer Slide Time: 16:58)



Let us define the kernel $K(\cdot)$

$$K(\underline{x}, \underline{x}_i) \triangleq \varphi^T(\underline{x})\, \varphi(\underline{x}_i)$$

$$= \sum_{j=0}^{m_1} \varphi_j(\underline{x})\, \varphi_j(\underline{x}_i) \quad \leftarrow \text{coordinates of } \varphi(\cdot) \quad \forall i = 1, \dots, N$$

This 'kernel' is symmetric in the arguments

$$K(\underline{x}, \underline{x}_i) = K(\underline{x}_i, \underline{x}) \quad \forall i$$

Directly from Condition 1 in the derivation of the Lagrangian, when we apply this non-linear transformation to x_i, the weight vector w becomes $w = \sum_{i=1}^{n} \alpha_i d_i \varphi(x_i)$. This leads us to what I'll refer to as Equation 2. Now, substituting Equation 1 into Equation 2, we find that $w^T \varphi(x) = 0$.

So, the transposition applies to $\varphi(x)$, leading to:

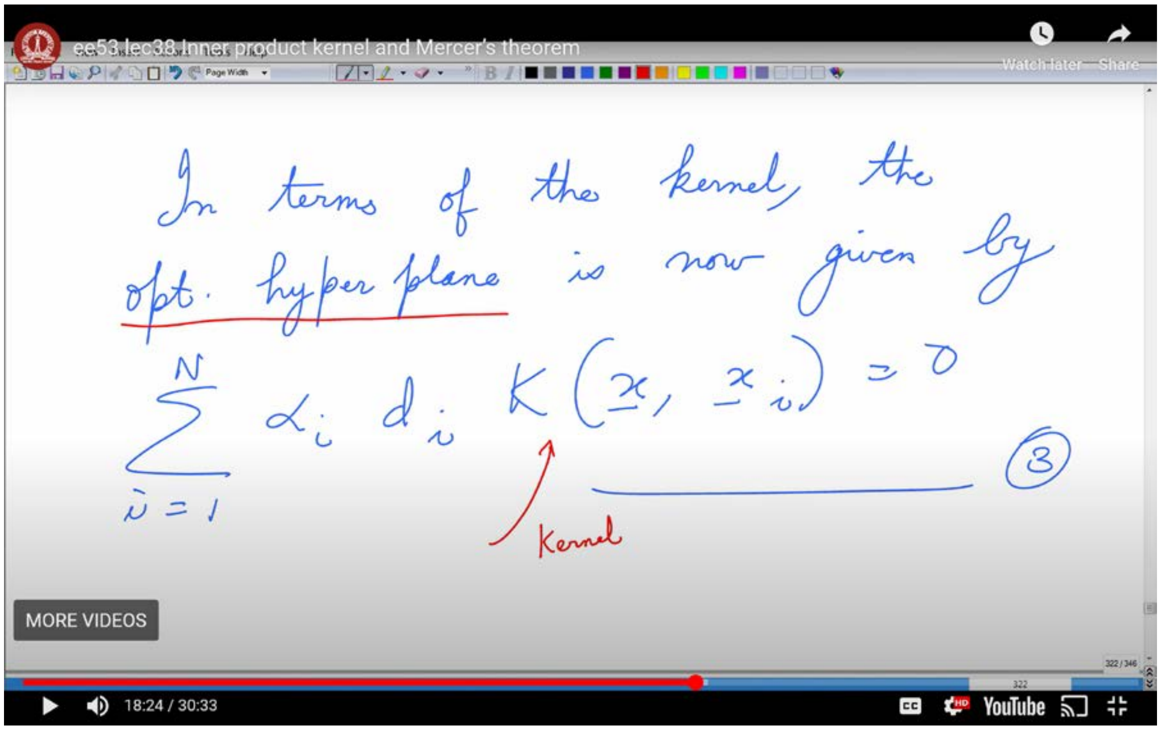
$$w^T = \sum_{i=1}^{n} \alpha_i d_i \varphi(x_i)^T$$

Now, when we multiply this by $\varphi(x)$, we get:

$$\left(\sum_{i=1}^{n} \alpha_i d_i \varphi(x_i)^T \right) \varphi(x) = 0$$

This expression represents the inner product of two vectors: φ(x) and φ(x_i). This inner product is induced in the feature space by the input x and the training pattern x_i. This is a crucial point in building this relationship.

(Refer Slide Time: 18:24)



In terms of the kernel, the opt. hyper plane is now given by

$$\sum_{i=1}^{N} \alpha_i d_i \underset{\text{Kernel}}{K(\underline{x}, \underline{x}_i)} = 0 \qquad (8)$$

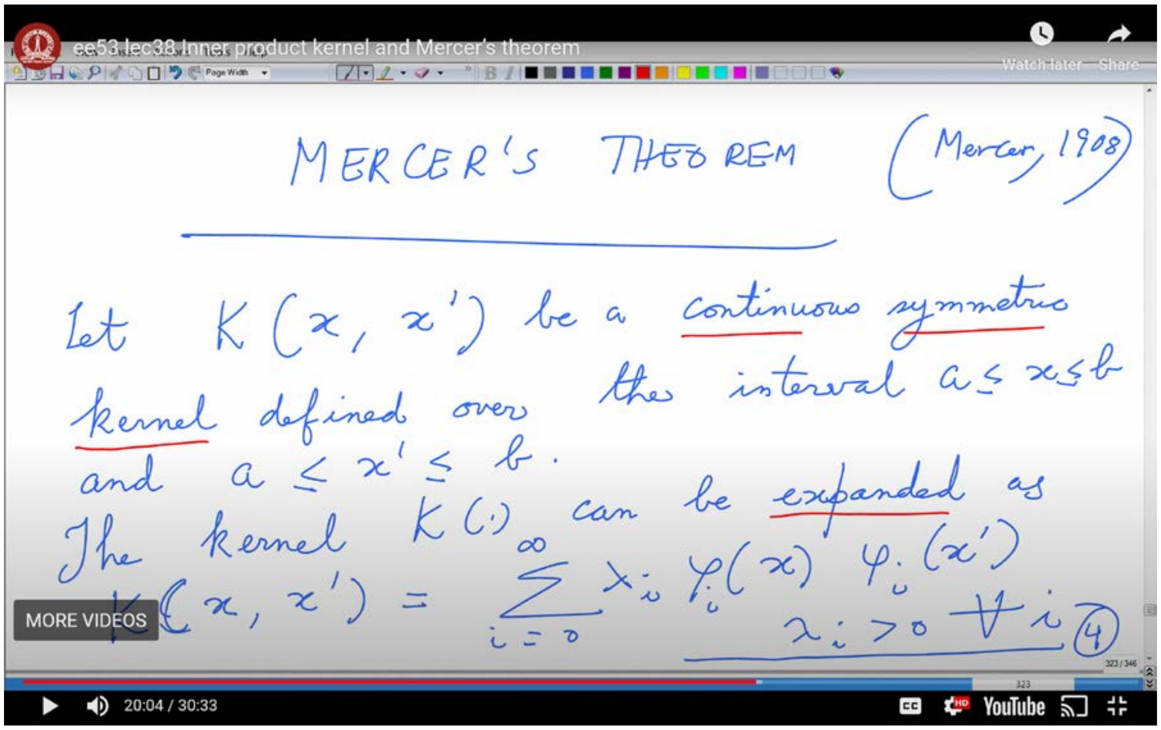
At this stage, we're ready to define the kernel, although we must still connect this kernel to its more intricate properties, such as those outlined in Mercer's theorem. But before delving into Mercer's theorem, let's clarify that the inner product of φ(x) with φ(x_i) is defined as our kernel k(x, x_i). Here, x_i is a vector in the input space, and x is another vector, both linked via this non-linear transformation.

Compactly, we can write this kernel as:

$$k(x, x_i) = \sum_{j=0}^{M_1} \varphi_j(x)\varphi_j(x_i)$$

where the index j represents the coordinates. This kernel is symmetric in its arguments, a property you can easily verify. Indeed, $k(x, x_i) = k(x_i, x)$ for all i. Recall from our discussions on regression estimation that kernels must satisfy two essential properties: a normalization constraint and a symmetry constraint. Interestingly, this kernel inherently possesses these properties.

(Refer Slide Time: 20:04)



ee53 lec38 Inner product kernel and Mercer's theorem

MERCER'S THEOREM (Mercer, 1908)

Let $K(x, x')$ be a continuous symmetric kernel defined over the interval $a \leq x \leq b$ and $a \leq x' \leq b$.

The kernel $K(\cdot)$ can be expanded as

$$K(x, x') = \sum_{i=0}^{\infty} \lambda_i \varphi_i(x) \varphi_i(x')$$

$$\lambda_i > 0 \ \forall i \quad \boxed{4}$$

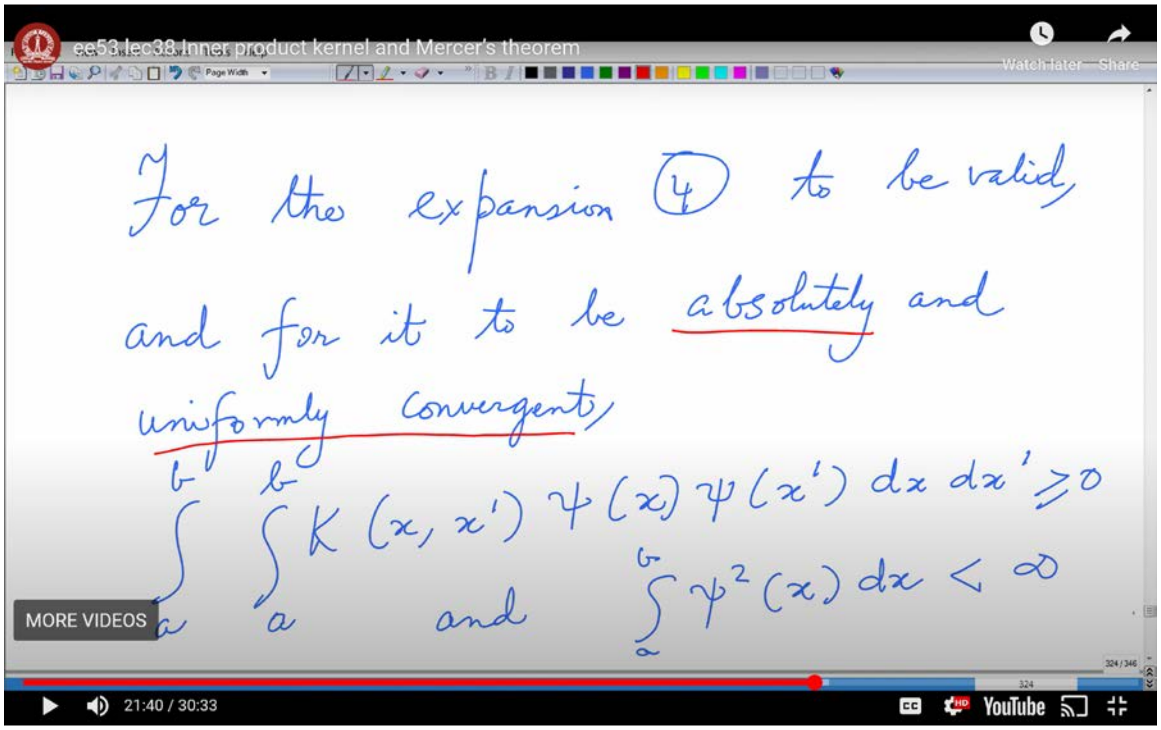
In terms of the kernel, the equation for the optimum hyperplane is now given by:

$$\sum_{i=1}^{n} \alpha_i d_i k(x, x_i) = 0$$

I will refer to this as Equation 2b. By rewriting Equation 2b in terms of the kernel's definition, we obtain what I'll call Equation 3. I have incorporated the notation 2a into Equation 2, so this is the equation for our optimum hyperplane.

Given the kernel k, the desired response d_i, and the Lagrange multiplier α_i, we can compute the optimum hyperplane using this formulation. With these tools, we now take a slightly different path into Mercer's theorem, a fundamental result established by Mercer in 1908, over a century ago. I will discuss the theorem's statement, its conditions, and how it applies to the concepts we are trying to establish here.

(Refer Slide Time: 21:40)



For the expansion $\boxed{4}$ to be valid, and for it to be absolutely and uniformly convergent,

$$\int_a^b \int_a^b K(x, x')\psi(x)\psi(x')\,dx\,dx' \geq 0 \quad \text{and} \quad \int_a^b \psi^2(x)\,dx < \infty$$

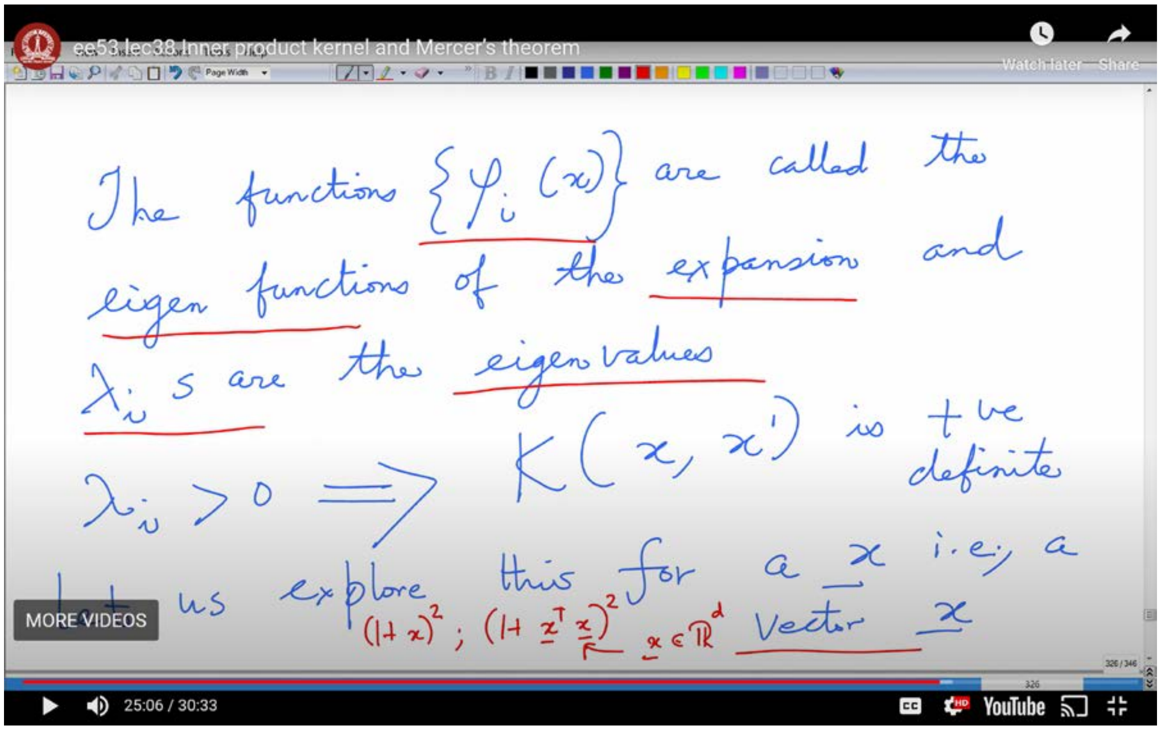
Let k be a continuous symmetric kernel defined over the closed interval [a, b]. In this scenario, both variables x and x' are defined over this closed interval [a, b]. The kernel k can then be expanded as follows.

The expansion we’re discussing is expressed as:

$$\sum_{i=0}^{\infty} \lambda_i \varphi_i(x) \varphi_i(x')$$

Here, λ_i is greater than zero for all i. This expression essentially represents an infinite summation, and if you look closely, you’ll notice that these φ_i functions resemble eigenfunctions, while the λ_i values are akin to eigenvalues.

(Refer Slide Time: 25:06)


ee53 lec38 Inner product kernel and Mercer's theorem

The functions $\{\varphi_i(x)\}$ are called the eigen functions of the expansion and λ_i s are the eigen values

$\lambda_i > 0 \implies K(x, x')$ is +ve definite

Let us explore this for a x i.e., a vector $\underline{x}$

$(1+x)^2$; $(1+\underline{x}^T\underline{x})^2$ $\underline{x} \in \mathbb{R}^d$

For this expansion to hold, similar to other series expansions we've encountered, like Fourier series, certain conditions must be met. Specifically, for this series to be valid, it needs to be absolutely and uniformly convergent. To ensure this, two critical conditions from analysis must be satisfied:

1. The double integral over the interval [a, b] of the kernel k(x, x') multiplied by ψ(x) and $\psi(x')$ must be non-negative:

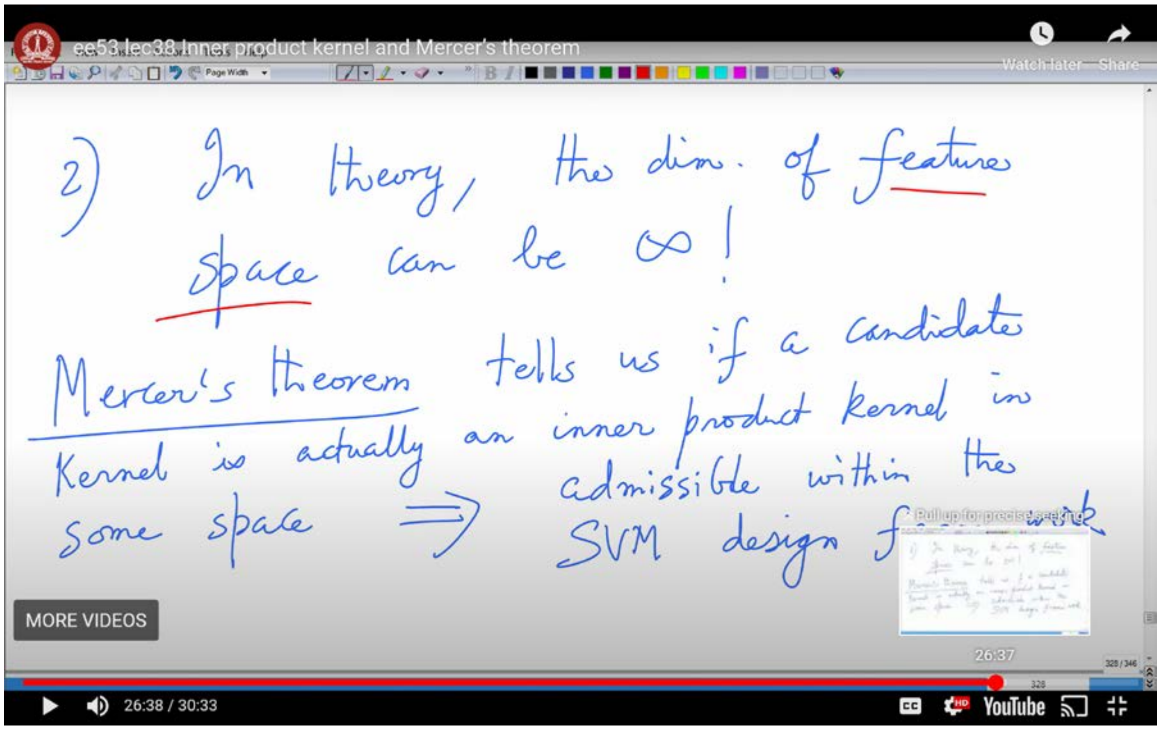
$$\int_a^b \int_a^b k(x,x')\psi(x)\psi(x')\ dx\,dx' \geq 0$$

2. The integral of the squared function ψ(x) over the interval [a, b] must be finite:

$$\int_a^b \psi^2(x)\ dx < \infty$$

These conditions are essential for the expansion to be both absolutely and uniformly convergent.

(Refer Slide Time: 26:38)



2) In theory, the dim. of features space can be ∞ !

Mercer's theorem tells us if a candidate Kernel is actually an inner product kernel in some space ⇒ admissible within the SVM design framework

Now, let's take a vector field approach to this problem. The scalar function ψ(x) can be interpreted as the induced self-norm of a vector φ(x). Here, φ(x) is a vector that could be infinite-dimensional, defined as:

$$\varphi(x) = (\varphi_0(x), \varphi_1(x), \varphi_2(x), \dots)$$

To better understand this, consider a finite case. Suppose we define ψ(x) as:

$$\psi(x) = (1 + x)^2$$

over the interval [0, 1]. We can expand $(1 + x)^2$ as:

$$1 + 2x + x^2$$

This can be viewed as the dot product of two vectors:

$$(1, \sqrt{2}x, x^2)$$

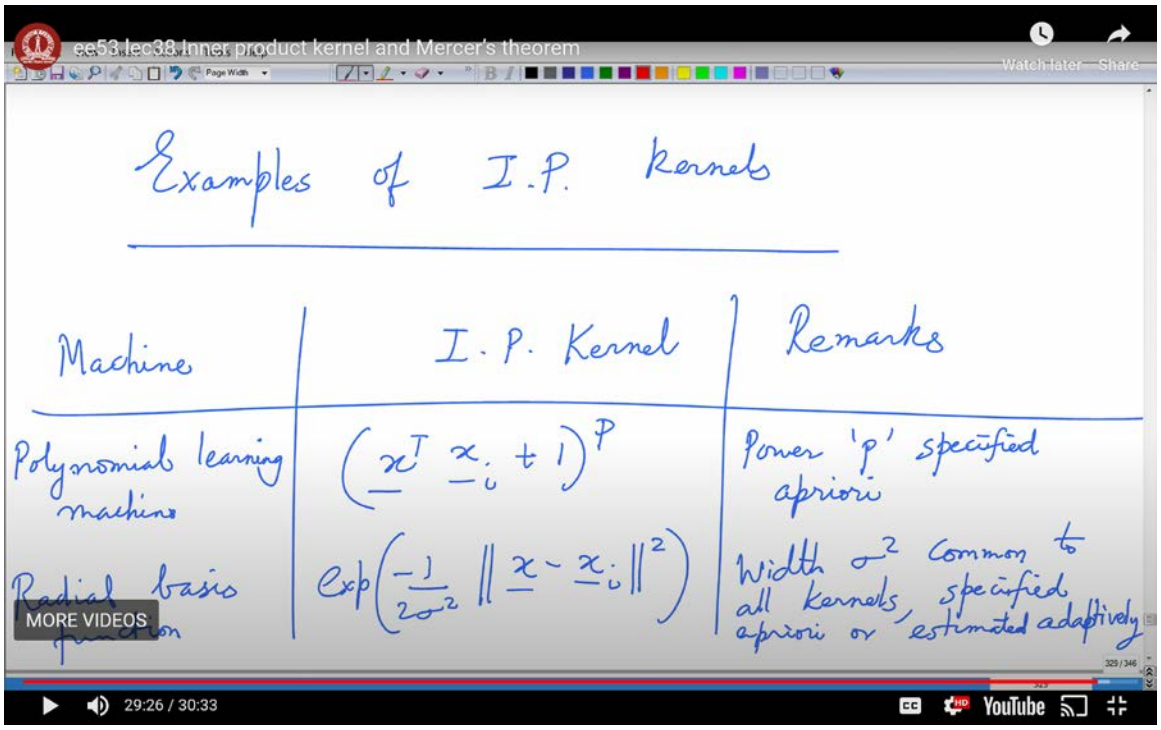
So, this scalar-valued function $\psi(x)$ can be expressed as an inner product of two vectors. In this context, the functions $\varphi_i(x)$ are the eigenfunctions of the expansion, with the corresponding eigenvalues λ_i. Thus, λ_i $\varphi_i(x)$ forms what we call an eigenpair.

If $\lambda_i > 0$ for all i, then the kernel k(x, x') is positive definite. This concept can be generalized beyond single-variable cases. For instance, instead of the expansion $(1 + x)^2$, we could consider:

$$(1 + x^T x)^2$$

where x is a vector in some d-dimensional space. This is a straightforward extension of the kernel for vectors.

(Refer Slide Time: 29:26)



Examples of I.P. Kernels

Machine	I.P. Kernel	Remarks
Polynomial learning machine	$(\underline{x}^T \underline{x}_i + 1)^p$	Power 'p' specified apriori
Radial basis function	$\exp\left(\frac{-1}{2\sigma^2} \lVert \underline{x} - \underline{x}_i \rVert^2\right)$	Width σ^2 Common to all kernels, specified apriori or estimated adaptively

Now, how does this relate to Mercer's theorem and our discussion of inner product kernels? According to Mercer's theorem, for any $\lambda_i \neq 1$, the ith image, $\sqrt{\lambda_i}\varphi_i(x)$, induced in the feature space by the ith coordinate in $\varphi(x)$, is an eigenfunction of the expansion. Remember, $\varphi(x)$ is a vector with coordinates $\varphi_1(x)$, $\varphi_2(x)$, $\varphi_3(x)$, and so on in the feature space.

This point is crucial: the ith image, $\sqrt{\lambda_i}\varphi_i(x)$, is an eigenfunction of the expansion. In theory, the dimensionality of the feature space could be infinite, which is useful in mathematical analysis. However, in practical machine learning scenarios, where datasets have finite dimensions, we must focus on expansions that involve a finite number of terms.

Mercer's theorem offers a critical insight that is highly valuable in the design of Support Vector Machines (SVMs). The theorem helps determine whether a candidate kernel can indeed be considered an inner product kernel in some feature space. For a kernel to be admissible within the SVM framework, it must satisfy certain conditions that ensure absolute and uniform convergence. If these conditions, as outlined in Mercer's theorem, are met, the kernel is valid for use in SVMs. This connection is vital.

We started from the input space, transitioned to the feature space, and in this feature space, we considered classifiers. The optimum hyperplane, which we aim to find, can be expressed in terms of an inner product kernel. This kernel must be defined as the inner product of two vectors obtained through a non-linear transformation from the input space, this is a crucial detail. Furthermore, the kernel must satisfy the specific properties dictated by Mercer's theorem, making this connection essential in SVM design.

Now that we've established the relationships between the input vector, feature vector, inner product kernel, and the conditions set by Mercer's theorem, let's explore some examples of how to construct these inner product kernels.

One example is the polynomial learning machine. In this case, the inner product kernel can take the form:

$$(1 + x^T x_i)^p$$

This expression can be expanded using the binomial theorem. The power p can be specified in advance. If p is large, the expansion will have a significant number of terms, which naturally leads to a higher-dimensional feature space. This increase in dimensionality enhances the potential for finding a linear classifier that can effectively separate the patterns.

However, the polynomial learning machine is not the only option. Radial Basis Functions (RBF) also provide a powerful alternative. An example of an RBF kernel is:

$$\exp\left(-\frac{1}{2\sigma^2}|x - x_i|^2\right)$$

Here, σ^2 represents the spread parameter, which we discussed during the design of RBF networks. This parameter σ^2 can either be predefined for all kernels or, preferably, estimated adaptively. The adaptive estimation method is often more effective, as it allows for better tuning of the RBF to the specific problem at hand.

Another potential inner product kernel is based on the hyperbolic tangent function. Under certain conditions on the parameters of the tanh function, it can also serve as an inner product kernel. We will revisit some of these cases in more detail as part of the homework exercises. So, with this understanding, we’ll stop here for now.