Introduction to Data Analytics

Prof. Nandan Sudarsanam and Prof. B. Ravindran

Department of Management Studies and

Department of Computer Science and Engineering

Indian Institute of Technology, Madras

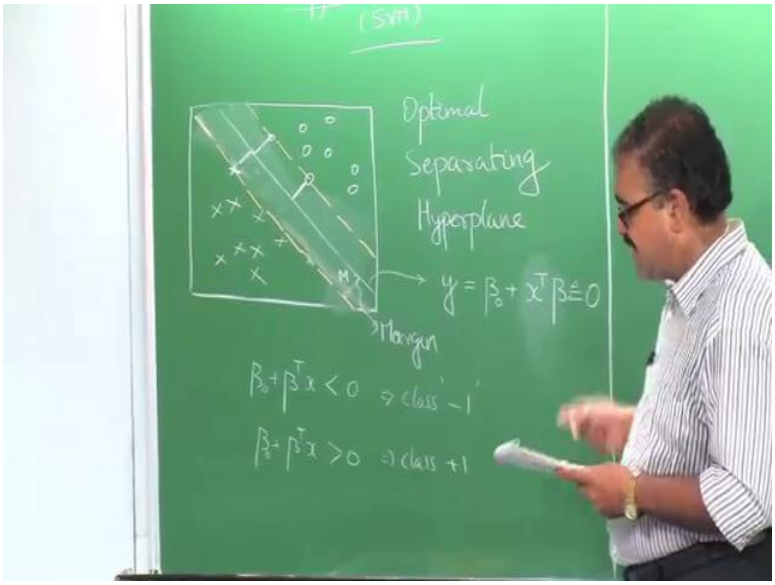
Module – 05

Lecture – 29

Support Vector Machines

Hello and welcome to this module on Support Vector Machines.

(Refer Slide Time: 00:19)



So, we have been looking at the variety of classifier so far and one of the things, let us look at the linear classifier. So, the one of the thing is, if I have data points that are even perfectly separable, here is a class and here is another class, you can see that they are very clearly separated. But, when I train a linear classifier it is not entirely clear, which of these many possible lines that could separate the data, would your classifier end up learning. There are many, many different lines that could separate the data and we are not sure, what your classifier would end up learning.

So, support vector machines initially were born out of and need to answer this question. Among all of these different lines or all of these different decision surfaces that you could use for separating the data given to you, which of those is the best decision surface? Some of all those alternatives which you think should be the best decision

surface. So, one answer to this question is to define an optimal separating, if an optimal separating hyper plane has the surface, such that the nearest data point to the surface is as far away as possible among all of it is surfaces.

So, here is a separating line and the nearest data point to that is that or that or that. So, if you think about it, so the nearest data point cannot belong to just one class. So, I could draw a line like this, but then there would mean that I am reducing the distance of the data point to the separating surface or if I go this way, again I will be the reducing the distance of the data point to the surface in one class or the other. When I say that you are maximizing the distance of the closest data point to the separating hyper plane, that essentially means that the closest data point from either class is at the same distance away from the hyper plane.

So, this distance and being same as this distance would be the same as this distance and this. So, this distance of the closest data point to the separating surface is known as the margin of the classifier, which will denote by m. So, the goal of finding a separating optimal separating hyper plane is essentially to find the classifier, such that this margin m is as large as possible. So, let us step back and think about what such a line means. You know in all linear classifiers we have seen so far, so we know that we are going to say something like.

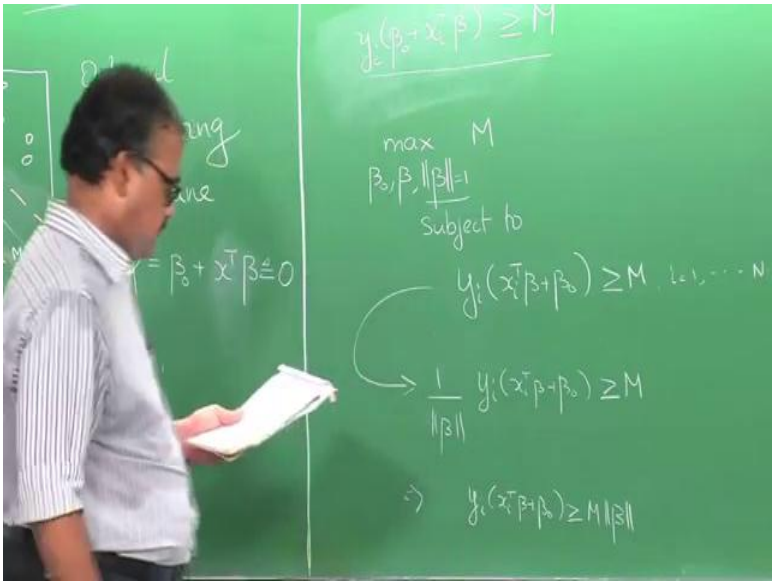
So, y is beta not plus beta transpose x, for convenience sake here I will write it as x transpose beta, since we are taking inner products that is fine. So, a line like this could essentially be obtained by setting this beta not plus x transpose beta equal to 0. So, all the data points on this line or those data points for which beta not plus x transpose beta evaluates to 0, so that is the equation of the line here. So, if it is negative beta not plus beta transpose x is less than 0, so we are going to say that x is of class minus 1 and if beta not plus beta transpose x is greater than 0, we will say that next class plus 1.

So, remember that, so we will, we using some kind of encoding for the class. The class could be does not buy a computer or buys a computer, he is sick, he is healthy. I mean the classes could be many different things, but numerically we are going to be assigning some encoding for the class and in this case, I choose to use minus 1 and plus 1 as the encoded. There is a reason for that as we will see shortly. So, if beta not plus beta transpose x is less than 0 and I say, it is class minus 1.

But, in this case what I really want, I do not want it to be this less than 0, but I want it to

be at least m away from the hyper plane. I want it to be m away from the line beta not plus beta transpose x equal to 0. So, I might use x transpose and beta transpose x interchangeably at points, but as you know they are inner products, so that is fine.

(Refer Slide Time: 07:18)



So, what I really want is, so y i is plus 1 I want beta not plus x i transpose beta to be greater than m. What happens if y i is minus 1? I really want it to be at least m away in that case as well, but then we know that beta not plus x i transpose beta would be negative, when the class is minus 1. So, what I do is I essentially just multiplied by the actual class variable and I want this whole distance. Because, if y i is plus 1 I would like this also to be plus, the positive and I want it to be at least m away from the hyper plane and y i is minus 1, this is going to be negative. So, the product is going to be positive and I want that to be at least m away from the hyper plane.

So, this is thick and strained that we want to satisfy and what is our goal. If we remember, our goal is to make sure that this m is as large as possible. So, what will do is, we will say maximize m beta not beta subject to… So, I am going to maximize the margin m over beta not and beta, subject to the constraints that y i times x i transpose beta plus beta not is greater than or equal to m, for every data point in my training data.

So, this kind be done assuming that all the data is nicely separated. So, and I can actually draw a linear surface that separates the data. So, if a kind of linear surface that separates data, then I can come up with at least one surface that satisfies this constraints for some value of m and essentially, I have to find value of m that is maximum here. But, one

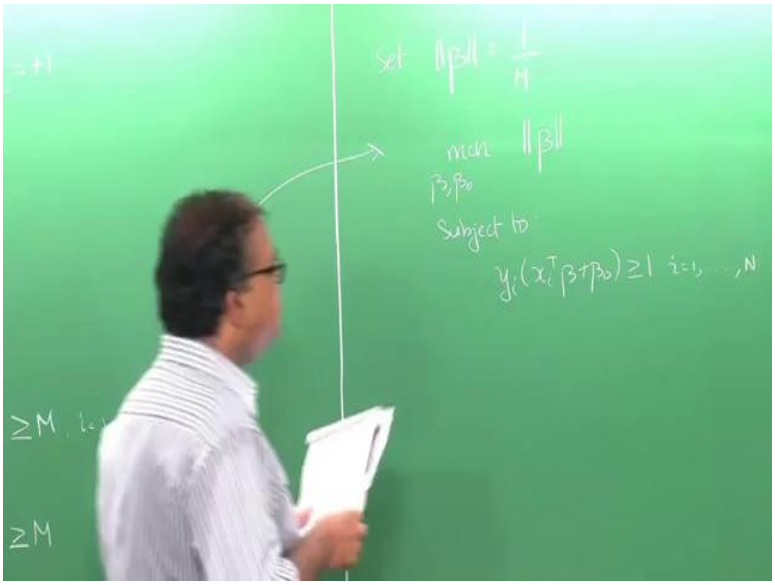
thing if you look at this equation or the constraint that we have written, so I can arbitrarily increase the value of beta and make this value as large as I want.

So, I need to have some constraint on beta as well. So, what we will do is, we will constraint the norm of beta to be equal to 1. So, we will not look at all possible weights beta not and beta, we will only look at those weights insist that the size of beta is constraint to be 1. So, the norm of beta is, you could take the Euclidean norm of beta, I am saying that the norm of beta should be 1. So, I hope the formulation of the optimization problem so far is clear.

So, it is essentially saying that I want all my data points to be at least a distance m away from the hyper plane and subject to that constraint and subject to my beta be norm one, I want to maximize the margin. So, this is a pretty works and constraint, so we can try to get rid of it by changing the other inequality constraints to by normalizing them with the data. So, this again allows me to achieve the same effect of not getting a high value for m just by increasing the size of beta, because I am dividing by the size of beta.

So, that achieves the same constraint and you can essentially write it like that. So, one thing that we should note here is that, if a specific beta satisfies these constraints, any positively scale version of beta would also satisfies the constraints. I can just multiplied by some positive number, if it is originally all, for all the exercise was giving me negative values larger than m or minus m or positive values larger than m, just multiplying it by a positive quantity will not change anything. It will still give me negative values that are lesser than minus m or positive values that are greater than m. Therefore I can essentially choose a specific value for beta, such that this evaluates to 1.

(Refer Slide Time: 13:21)



So, I said, so accept norm beta equal to 1 by m, so that this constraint becomes y i x i transpose beta is greater than equal to 1 subject to the constraint that, you are finding the smallest such beta. So, this optimization problem then becomes, this is optimization problem of maximizing the margin, now essentially becomes the problem of finding the smallest beta, such that this conditions are satisfied. So, this is essentially means that my margin here is going to be 1 over now beta.

So, to make it mathematically more convenient I am going to minimize the quadratic form of that. So, essentially I will be minimizing this square of beta, since it is norm any way. So, this would be positively to begin with, so I can minimize this square, that is not a problem and so that is my final optimization problem. So, this is the final optimization problems, where I am saying that, so together with these constraints a kind of define a slab around the separating hyper plane, I define a slab around these separating hyper plane of with 1 by beta. So, making sure that there are no data points with in this region, so I am trying to now maximize the width of this region, so that there are no data points in that region, that is essentially the idea behind this optimization problem.

So, this defines the basic optimization problem in the case of support vector machines. So, in the next module we will look at, how do you go about setting up a solution for this optimization problem.