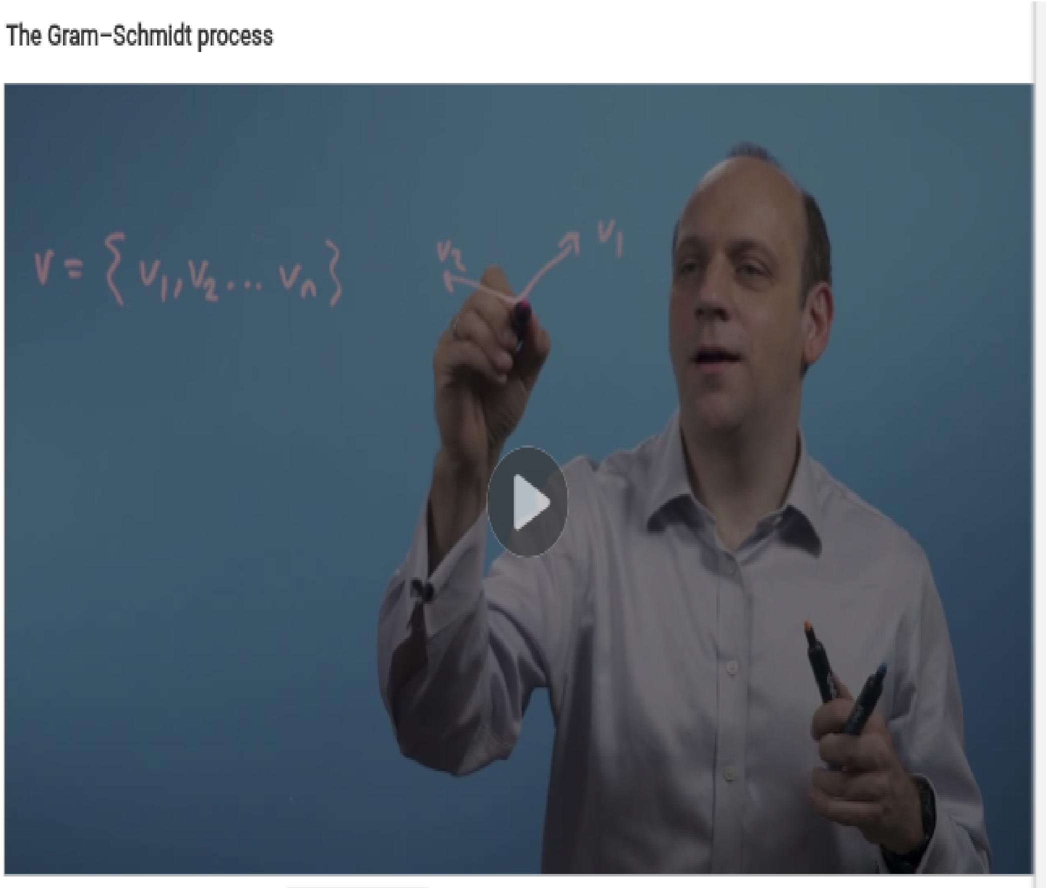
DAILY ASSESSMENT FORMAT

Date:	**16/07/2020**	**Name:**	Nichenametla Bhargavi
Course:	Courseera	**USN:**	**4AL17EC061**
Topic:	**Machine Learning using Linear Algebra**	**Semester & Section:**	**6th Sem A Sec**
Github Repository:	Bhargavi_Nichenametla		

SESSION
Image of the session



Report – Report can be typed or hand written for up to two pages.

Einstein summation convention and the symmetry of the dot product:

Now, there's an important other way to write matrix transformations down. It's called the Einstein's Summation Convention. And that writes down what the actual operations are on the elements of a matrix , which is useful when you're coding or programming. It also lets us see something neat about the dot product that I want to show you. And it lets us deal with non-square matrices. When we started, we said that multiplying a matrix by a vector or with another matrix is a process of taking every element in each row in turn, multiplied with corresponding element in each column in the other matrix, and adding them all up and putting them in place. So, let's write that down just to make that concrete. So I'm going to write down a matrix A here, and I'm going to give it elements.

A's an n by n matrix. I'm going to give it elements a11, a21, all the way down to an1. And then a12, all the way across to a1n. And then I'll have a22 here all the way across, all the way down until I fill it all in and I've got ann down here. So the first suffix on this matrix, first suffix on all of these elements in the matrix is the row number, and the second one is the column number. Now, if I want to multiply A by another matrix B, and that's also going to be an n by n matrix, and that will have elements b11, b12 across to b1n, and down to bn1 and across to bnn, dot, dot, dot, dot, dot, dot, dot, dot, dot.

If I multiply these together, I'm going to get another matrix, which I'll call AB, and then what I'm going to do is I'm going to take a row of A multiplied by the elements of a column of B and put those in the corresponding place. So let's do an example. So if I want an element, let's say ab, element two, three. I'm going to get that by taking row two of A, multiply by column three of B. So I'm going to take row two of A, that's going to be a21, a22, and all the others up to a2n, and I'm going to multiply it by column three of B. So that's b13, b23, all the way to bn3. And I'm going to add all those up. And I'll have a dot, dot, dot in between. So that's going to be this element, row two, column three of AB. Now, in Einstein's convention, what you do, is you say, well okay, this is the sum over some elements j of aij, bjk. So if I add these up over all the possible j's, I'm going to get a11, b11 plus a12, b21, and so on, and so on, and that's for i and k as well.

I'm going to then go around all the possible i's and k's. So, what Einstein then says, well okay, if I've got a repeated index, I won't bother with the sum and I'll just write that down as being aij, bjk. And that's equal to this the product abik. So abik is equal to ai1, b1k, plus ai2, b2k, plus ai3, b3k and so on and so on, until you've done all the possible j's, and then you do that for all the possible i's and k's, and that will give you your whole matrix for AB, for the product. Now, this is quite nice. If you are coding, you just run three loops over i, j and k, and then use an accumulator on the j's here to find the elements of the product matrix AB. So the summation convention gives you a quick way of coding up these sorts of operations. Now, we haven't talked about this so far but now we can see it. There's no reason, so long as the matrices have the same number of entries in j, then we can multiply them together even if they're not the same shape. So we can multiply a two by three matrix, something with two rows and three columns. So one, two, three, one, two, three, by a three by four matrix, three there and four there. So it's got one, two, three, four times. And when I multiply these together, I'm going to go that row times that column.

I've got the same number of j's in each case, so and then I'm going to be able to do that for all of the possible columns, so I'm going to get something with four columns. And I'm going to be able to do that for the two rows here. I'm going to be able to do that row times that one, is going to get a two by four matrix out. So it's going to have one, two, three, four, one, two, three, four. So I can multiply together these non-square matrices if I want to, and I'll get, in the general case, some other non-square matrix. I'm going to have the number of rows of the one on the left and the number of columns of the one on the right.

Orthogonal matrices:

* Now in data science what we're really saying here is that wherever possible, we want to use an orthonormal basis vector set when we transform our data.

* That is, we want our transformation matrix to be an orthogonal matrix. That means the inverse is easy to compute.

* It means the transmission is reversible because it doesn't collapse space. It means that the projection is just the dot product.

* Lots of things are nice and pleasant, and easy.

* If I arrange the basis vectors in right order, then the determinant is one, and that's an easy way to check and if they aren't just exchange a pair of them and actually then they will be determinant one rather than the minus one.