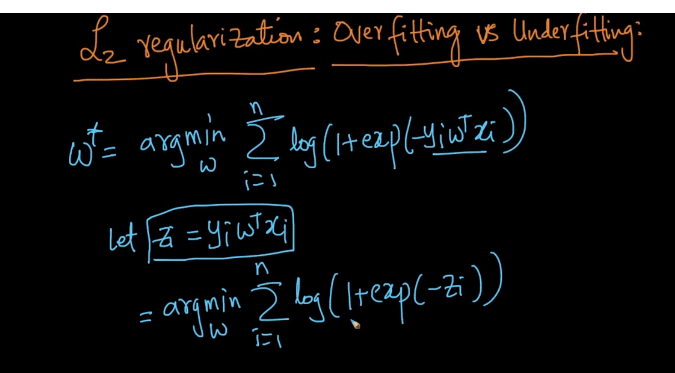
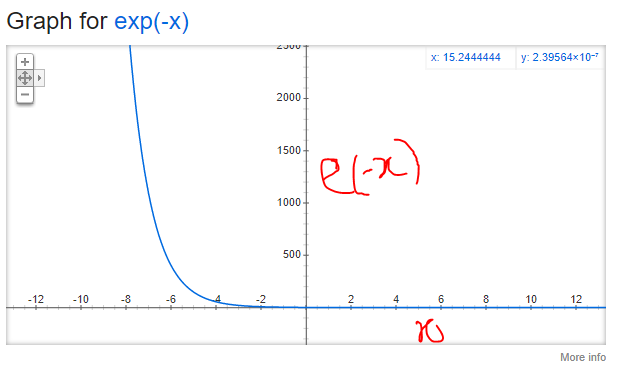
Optimisation problem we have looks like.

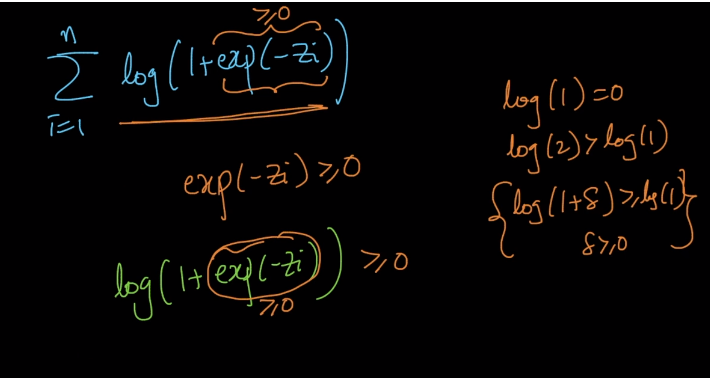


After assuming YiWtXi as Zi it looks like as we can see in above image.

So since we know that log in an monotonically increasing function and we also know that from below image that exp(-x) is always >= 0.



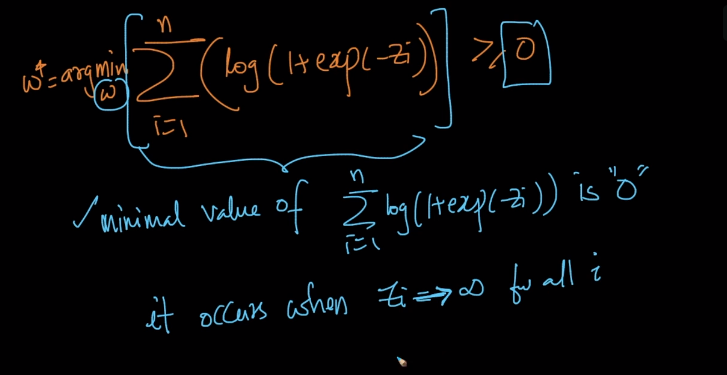
So we can say that exp(Zi) is always >= 0.



And since log(1) = 0, this objective function can lead to 0 if exp(-z) leads to 0,

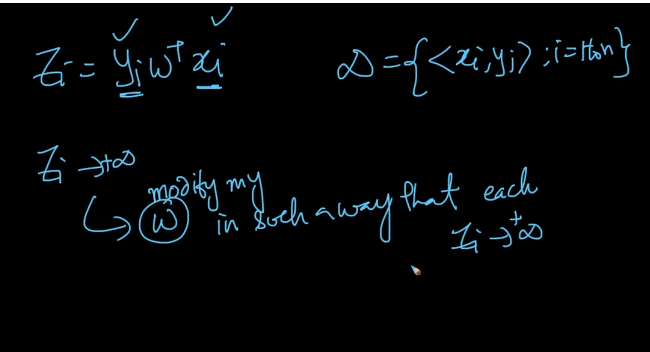
Now when exp(-z) leads to 0, As we can see from graph of exp(-x), whenever x tends to infinity, exp(-x) will become 0.

Now as we can say that objective function can minimum go to zero and this happens when the value for Zi tends to infinity.



Because when Zi is infinity for all i, than exp(-Zi) is nearly equal to 0 and since log of 1 is also equal to 0 so the final value we get is equal to zero.

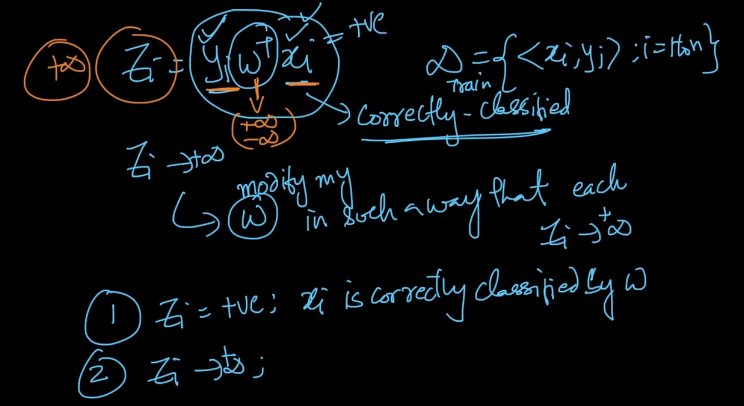
And what our Zi is, it is XiWtYi and since Xi and Yi are our data points so we need to make our W such that it tends to infinity for all i.



So now to make that happen means to make Zi as infinity our Wt will go like –(infinity) or (infinity) depending on the values of Yi and Xi and so even if we do that and we get all the points as correctly classified, even in that case we say that It is an overfitting problem because model is predicting every point correctly even there are outliers, and such situation is called overfitting.

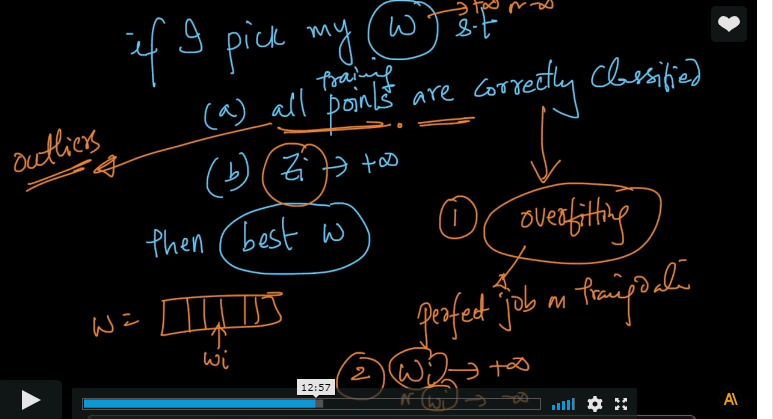
Because this is what overfitting means that if are getting almost 100% accuracy on our training data than we say that it is an overfitting problem.

So we can say that if Zi is positive it means Xi is correctly classified by W.

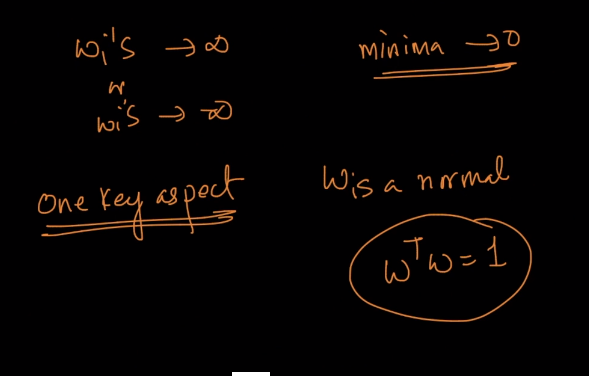


So if I pick my W such that all the points are correctly classified and Zi tends to infinity(Which means that our W will tend to (infinity) or (-infinity)

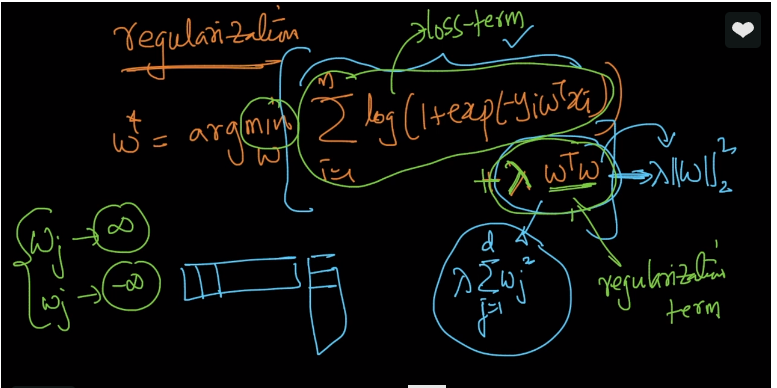
So it has a problem that the value of Wi goes really high and it can get affected by outlier.



In all this theory we have missed a fact that W is an normal vector which means Wt\*W = 1.



So to avoid this problems of infinity we will modify our equation with something called as Regularization.

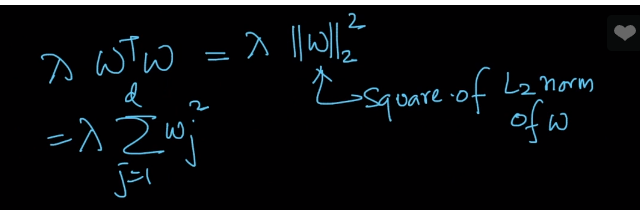


So to understand regularisation suppose we have a ruler in a city which doing everything as he wishes so we introduce some person against which wont allow him to do as his wishes but will settle the things to some sweet extent.

So the Regularization term we are introducing her is that person which wont allow our expression to choose W to (infinity) or (-infinity).

So what Wt\*W is, it is equal to square(||W||) and if the equation chooses W as infi or -infi so this term will go to infinity which is not and minimum value we needed to find in objective function.

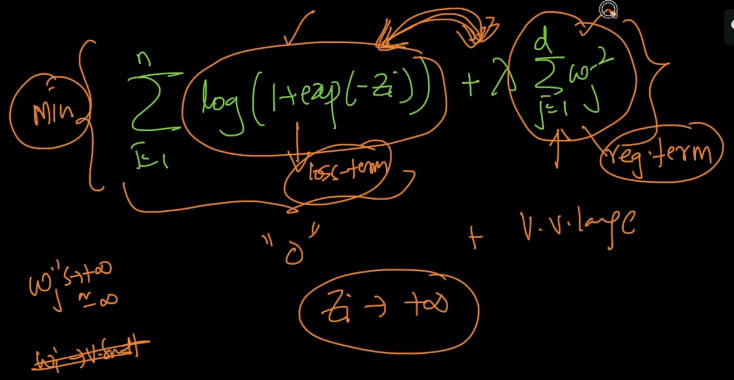
So by adding regularization term we are not allowing our equation not to choose W as infi or -infi.



So in the below image the first part of whole equation is said to be as Loss-term and we are adding reg. term to the loss- term.

By choosing W as (+,-) infi loss term will go to zero but our reg. term will be very very large and we cant allow that, because we are finding minimum value.

And so it wont allow loss term to choose W as infi.



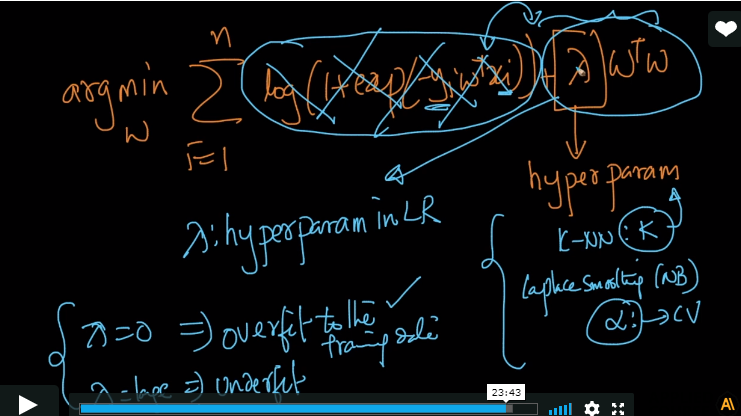
SO (lambda) here in our Reg. term is an hyperparam which handles the whole reg. term.

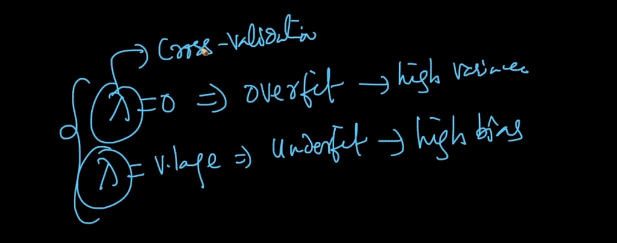
As we had K in KNN and (Alpha) in Laplace Smoothing.

SO we will calculate (Lambda) in same manner as we calculated other hyperparams that is by using CV.

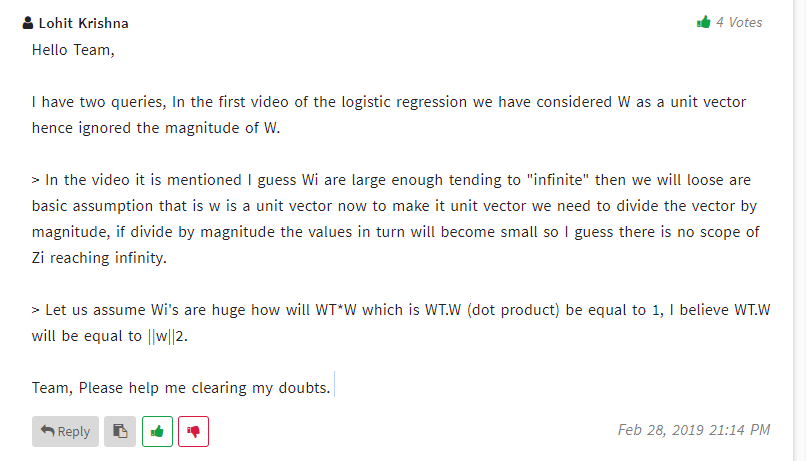
So when the value of lambda is 0 so Reg. term goes to zero and we again reach to the problem of Overfitting

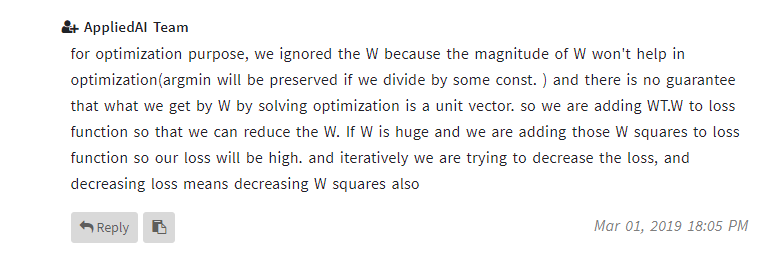
and when our lambda is very high it diminishes the impact of Loss term, because now our objective function will only try to minimize the regularization term as it’s very high, and it does not give importance to loss term in which our training data is present, in such case our model gets trained without any information of training data and since will no longer be using our training data to find W and which is problem of Under fitting and hence high bias.





Comments:





**Doubt:**

