**Log-Probabilities for numerical stability:**

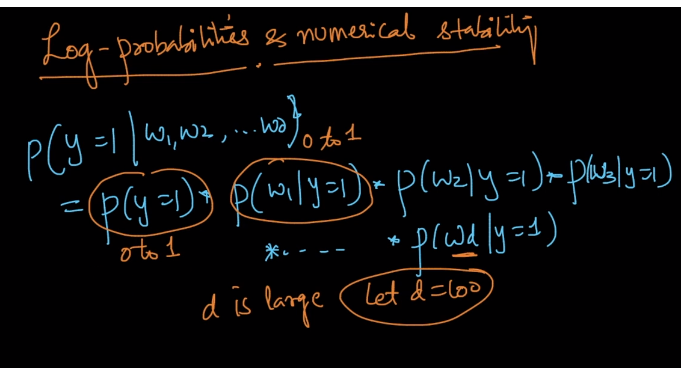
There in one more problem other than Laplace Smoothing ,

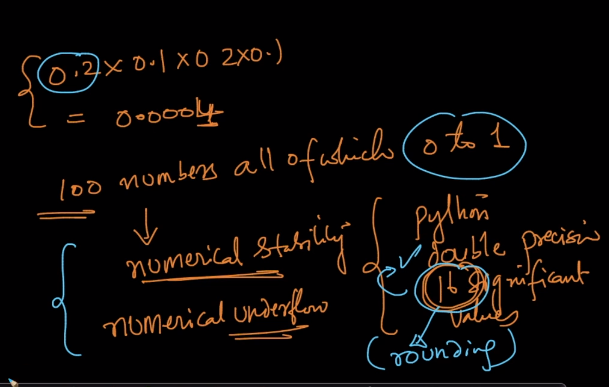
When we are multiplying probabilities of d likelihoods and one class label where each value is between 0—1 and suppose if d is large ,lets say 100

It will result in really a small value which we wont even be able to store in python properly because there is double float which has 16 maximum digits and since we are multiplying 100 decimal values

It will result in really small and to store the value it will start rounding the value and make possible errors.

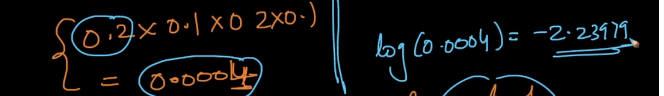
This phenomenon of multiplying 100’s of number all between 0-1 we can run into a problem of “**Numerical Stability”**





When python starts rounding number which is very very small so to store it in some manner is called **numerical underflow**.

The solution to this is using log for final multiplication value.

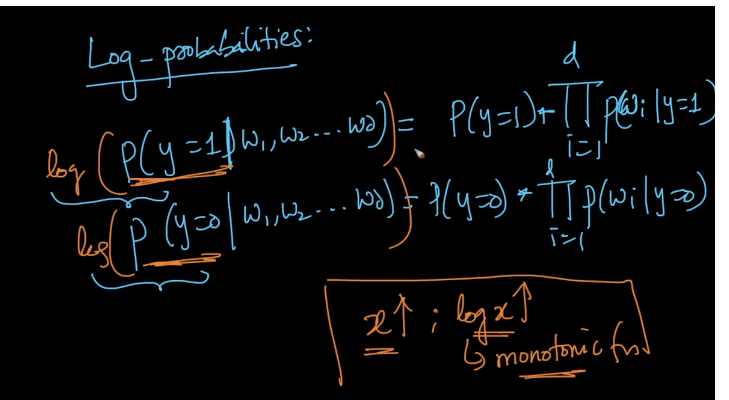


It is better to implement log than to play with such small values.

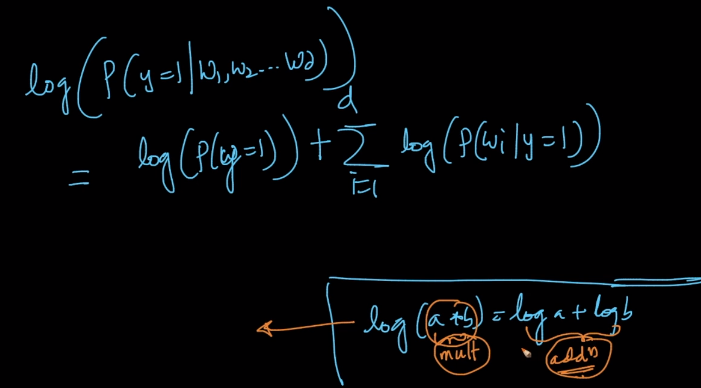
And this probabilities are called “**Log Probabilities”.**

So now instead of comparing P(X= 1|q) to P(X=0|q) we will compare

Log(P(X=1|q)) to log(P(X=0|q)) and since log is monotonic function which means if x increase log(x) also increase it is totally correct.



Now by doing this it will transform the formula to Naïve Bayes algorithm



i.e., it will convert all multiplication into addition.