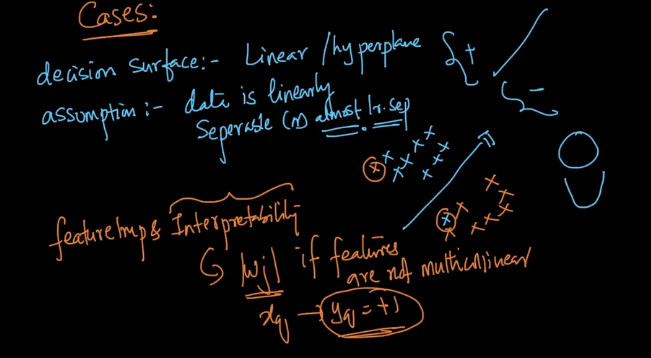
So now take the real world cases to understand LR in better fashion.

The decision surface in LR is an linear or hyperplace because its assumption is that the data is Linearly separable or almost linearly separable.

For feature importance we have got |Wj| and by using it we can check feature importance and interpretability.



What we do in case of **imbalanced data**: its upsampling or downsampling as we learnt in previous chapters.

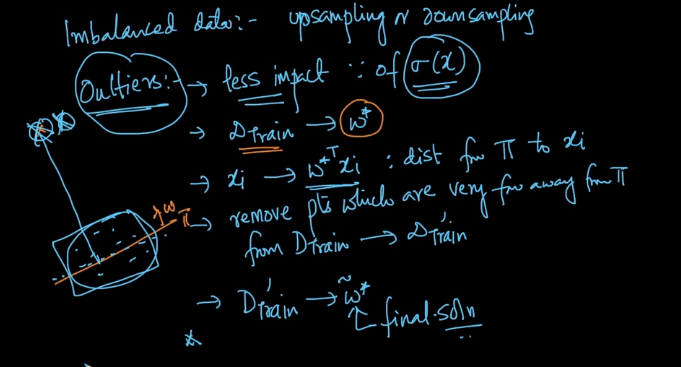
What to do in case of **outliers?**

LR is very less impacted by outliers because of sigmoid function and after that if we face issue with outliers so we can do is:

WE find W\* using whole Dtrain and then using that W we find distance from hyperplane

And remove the points that are much far from our cluster and then again find the W\* using the new Dtrain.

And then we consider it as a final solution.



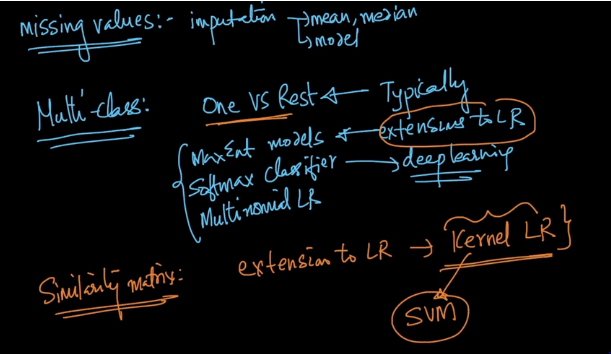
What to do in case of **missing values**?

Imputations like mean, median, mode.

What to do in case of **Multi-class**?

So in case of multi-class we always have One vs Rest and other than that we have methods like softmax classifier or multinomial LR(both are same)

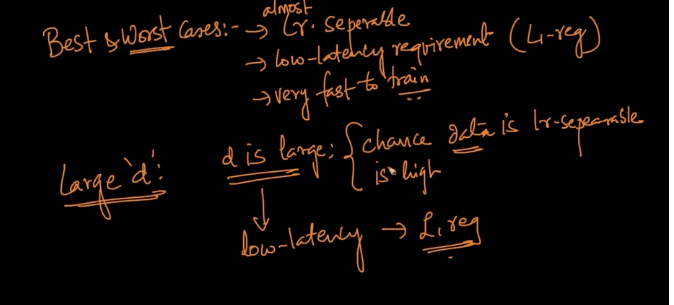
And we will learn it when we learn Deep learning.



The worst and best cases for LR is :

Data is almost linearly seperable and in case of low latency requirement it is v.good and very fast to train

But when d is very large then the performance for the model is affected and hence we need to use L1 reg. to overcome this problem



This are all the cases with LR.

Additional video for How Imbalanced data affects LR?

Let’s take example where we have 20 +Ve and 3 –ve points,

Now for plane 1 we get max value as 18.4, NOTE: plane 1 is correctly separating +Ve and –ve points.

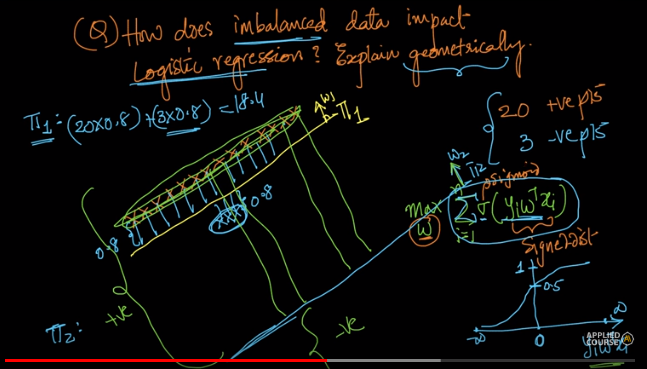
And for plane 2, it’s classifying all the +Ve points correctly and –ve points incorrectly because plane is too far from points,

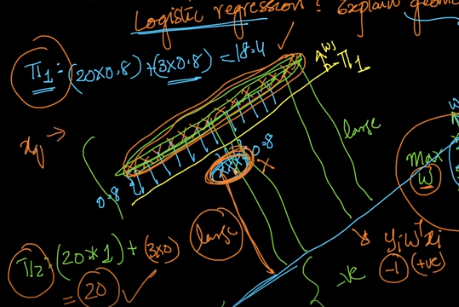
So, because of sigmoid function for all +Ve we get 1 and for all –ve we get 0, which results in output of model as 20.

Comparing results of both plane, plane 2 is max than plane 1 so we took plane 2, even though it’s misclassifying –ve points.

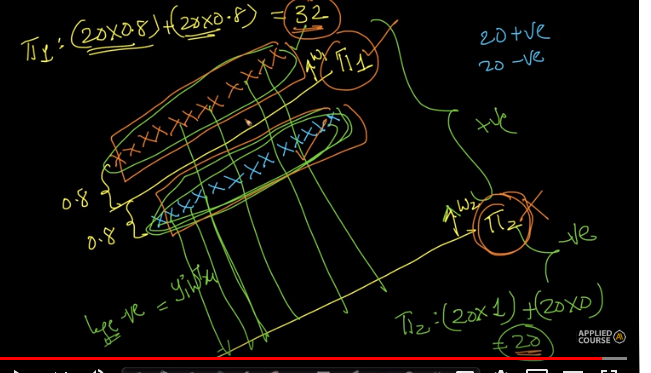
**What’s happening?** : Since we have small –ve points, so in order to maximize objective fun, plane tries to go as far as possible from +ve.

And because of this it would lead to predict any point as +Ve.





Example where we have balanced data.



Comments:

