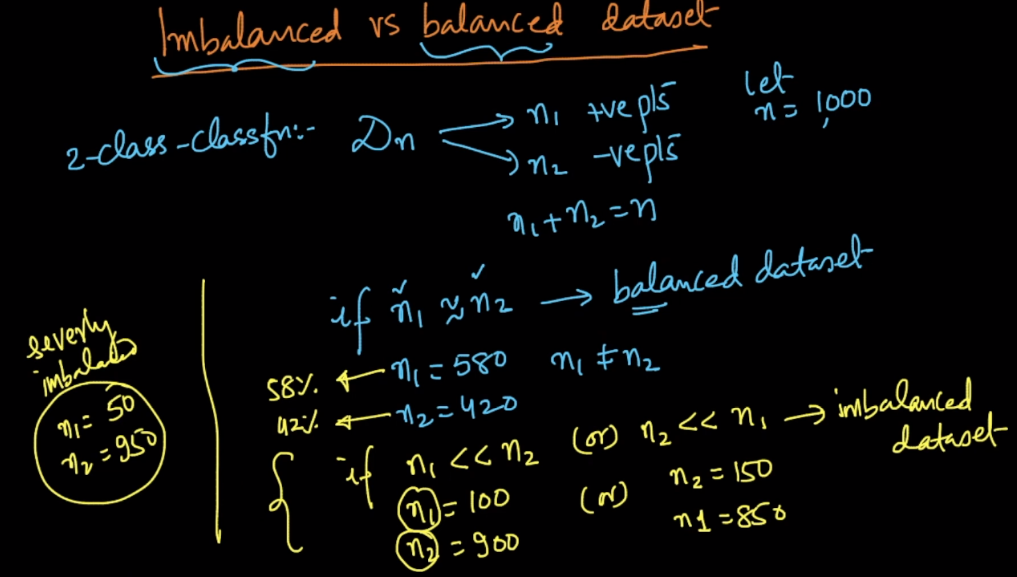
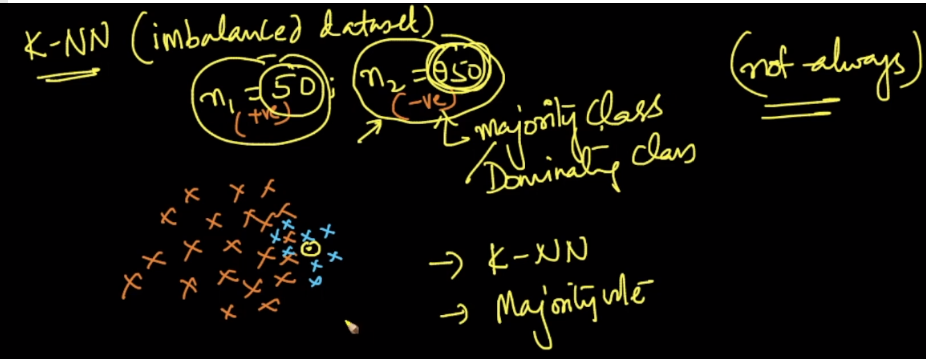
**Balanced dataset:** let’s say there are two classes in data set(+ve and -ve). If no. of points of each class are approximately equal or there is little difference between them, then such dataset is called balanced dataset.

**Imbalanced dataset:** let’s say there are two classes in data set(+ve and -ve). If difference between no. of points of each class is significantly large, then such dataset is called imbalanced dataset. Imbalance dataset is encountered a lot in health dataset.



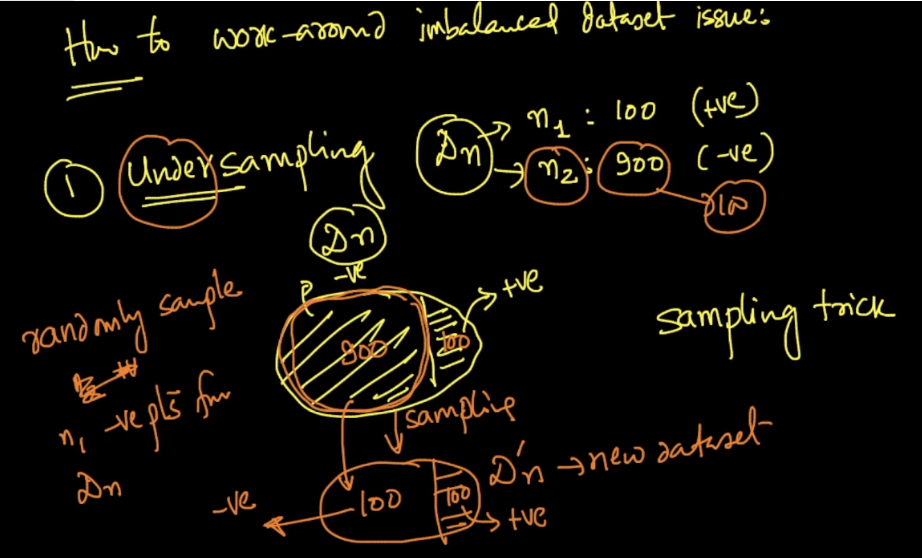
The problem with imbalanced dataset is that modelled will not be trained appropriately to predict output for new points. There are several ways to handle imbalanced dataset which are explained below.

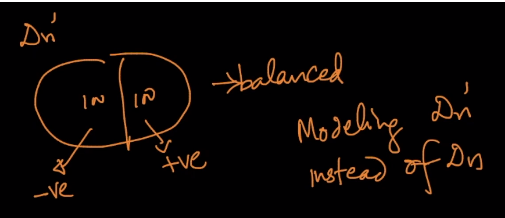


**1) Under Sampling:**

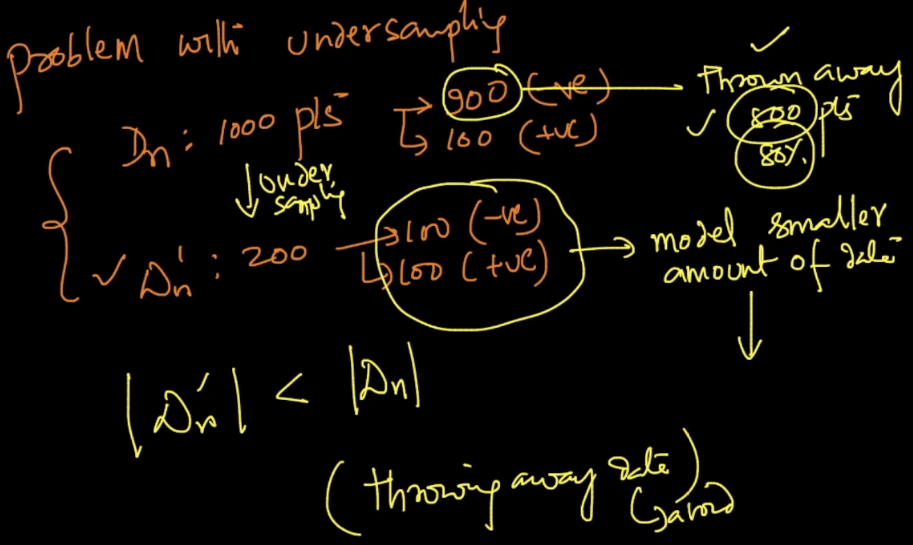
Let’s say you have 100 +ve points and 900 -ve points, then sampling in under sampling you will take only 100 samples randomly from 900 -ve points, and all the points from +ve points.

In this way you have 100-100 samples of each category.





**Drawback of under sampling:** Since in ML, the more data you have the more it will be good for a model, but in under sampling you are throwing a large portion of data around(70-80%), which contains a lot of information, and your model would not be able to learn from all the data. Hence model will not perform good.

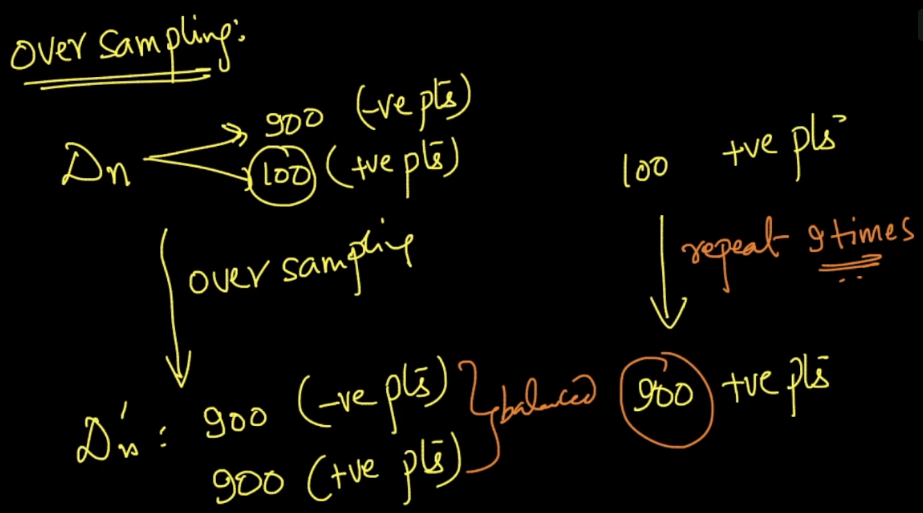


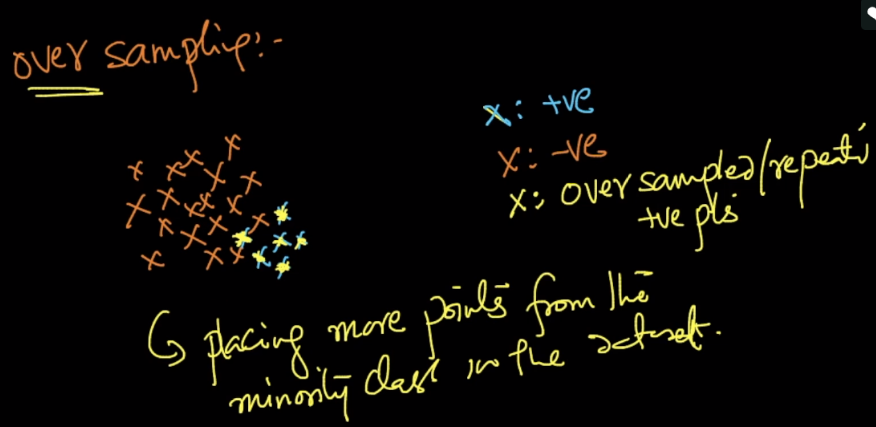
**2) Over Sampling:**

Another method to handle imbalanced data is over sampling, which is as follows:

Suppose you have 900, -ve points and 100, +ve points. So what we do is repeat each +ve points 9 times, therefore we will get 900 +ve points now.

Gemotrically we are placing 9 point on above of each +ve points.

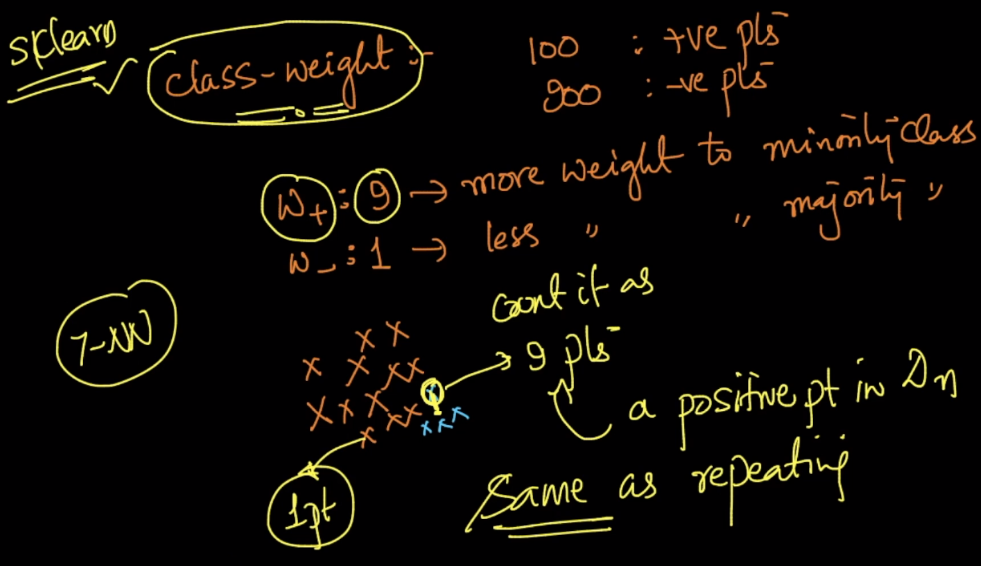




There is also another way of doing over sampling that instead of repeating each point 9 times, we’ll assign a weight according to the ratio we have, to each +ve points.

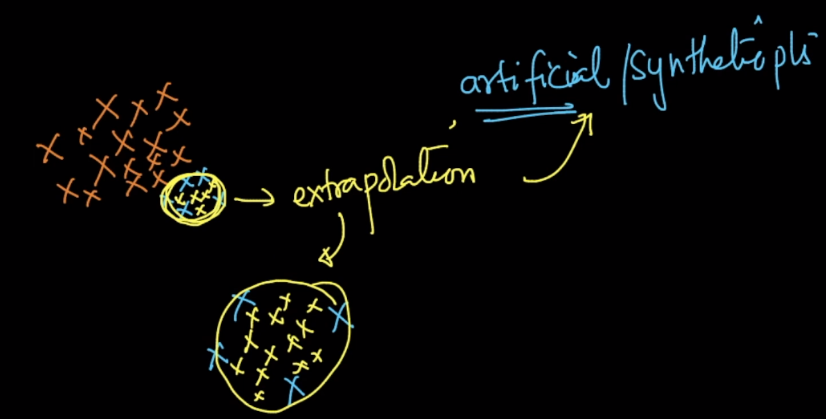
Since there are 100 +ve, and 900 -ve points. Therefore the ration is 1:9, that means we assign a weight of 9 to each +ve point and weight of 1 to each -ve point.

So whenever we query a point in K-NN and whenever it encounters +ve points as nearest, that one +ve point would be considered as 9 +ve points. So we are just creating a situation like repetition.



**3) Extrapolation/Artificial points:**

The another way we can use to handle imbalanced dataset is we can create artificial points in the region of +ve points, as given in below image, there are only 4 +ve points, but we have created extra artificial +ve points.



There is one more problem with imbalanced dataset that even if don’t do undersampling or oversampling, we could get high accuracy. **How ?**

Suppose we’ve 100 +ve and 900 -ve points, and we do train test split in 7:3 ratio.

Therefore train data have 70+ve and 630 -ve

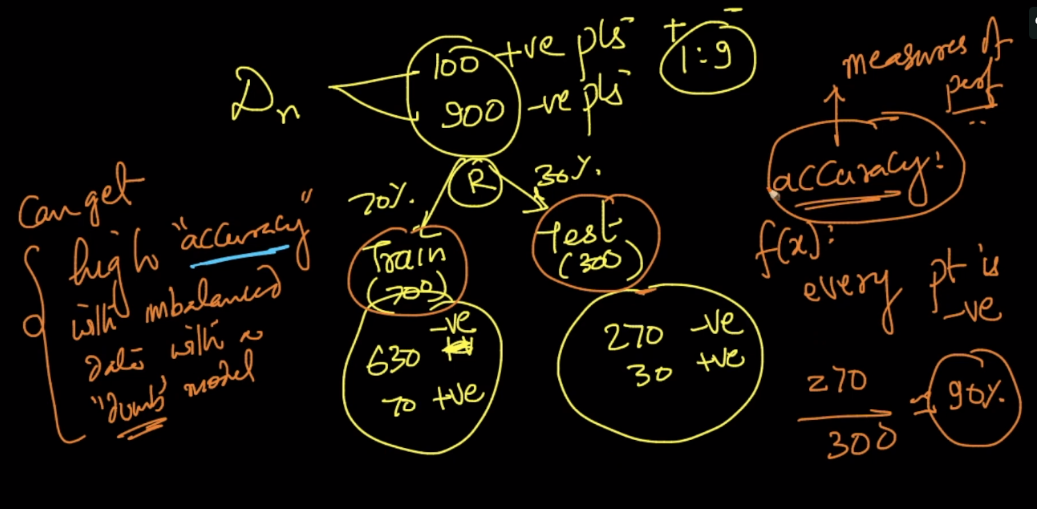
Test data have 30 +Ve and 270 -ve.

So even if our model always give prediction as -ve points, since there are large no of -ve points, then also we would 90% accuracy on test as:

It would predict 270 correct prediction out of 300 (30 + 270) test data, that is 90% accurcay.

So for imbalanced data even a dumb model would give such accuracy.

**Therefore we must check for balanced or imbalance data before applying them to any classifier model. Prefer Oversampling most over undersampling, by doing this you would not loose any information.**



**Comments:**

* For multi level classification how these sampling is being performed

For a multi-class classification, the algorithm internally uses One-vs-Rest approach that internally creates binary classifiers for each class. We need to balance the dataset manually if we are going with Oversampling/undersampling, whereas if we are going with balancing the class weights, then if we use **class\_weight='balanced'**parameter while building the model, it internally balanced the points for each binary classifier.

* If we do oversampling we are increasing the datapoints which again results in slow processing, which may require high computing end i.e a computer with 32Gb or 64 gb of RAM which is again expensive. Is there any other way?

You can go with balancing the dataset by balancing according to the class weights. This can be done by setting the parameter class\_weight='balanced'

* Over-sampling or undersampling is performed only when we have an imbalanced dataset.

In practise, if we are working on a problem where the dataset is huge and the system memory/RAM isinsufficient to handle, then we have to go for undersampling. Whereas in case if there are no memory constraints, then we can easily balance the dataset using over-sampling. But in general, oversampling creates replication of points from the minority class which coudn't yield best results but it is better when compared to leaving the dataset in an imbalanced state. The best way is to handle this by balancing using the class weights