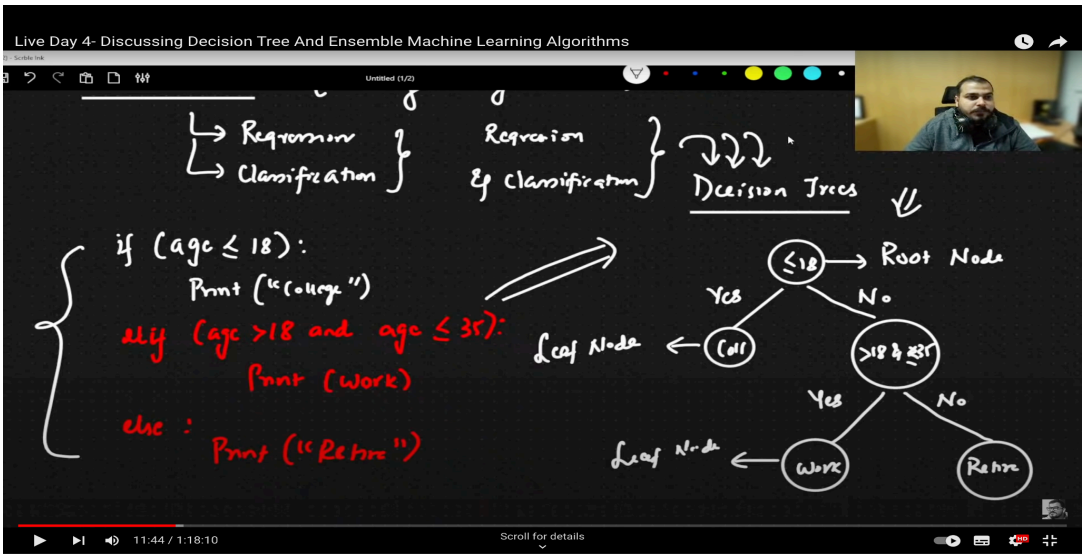
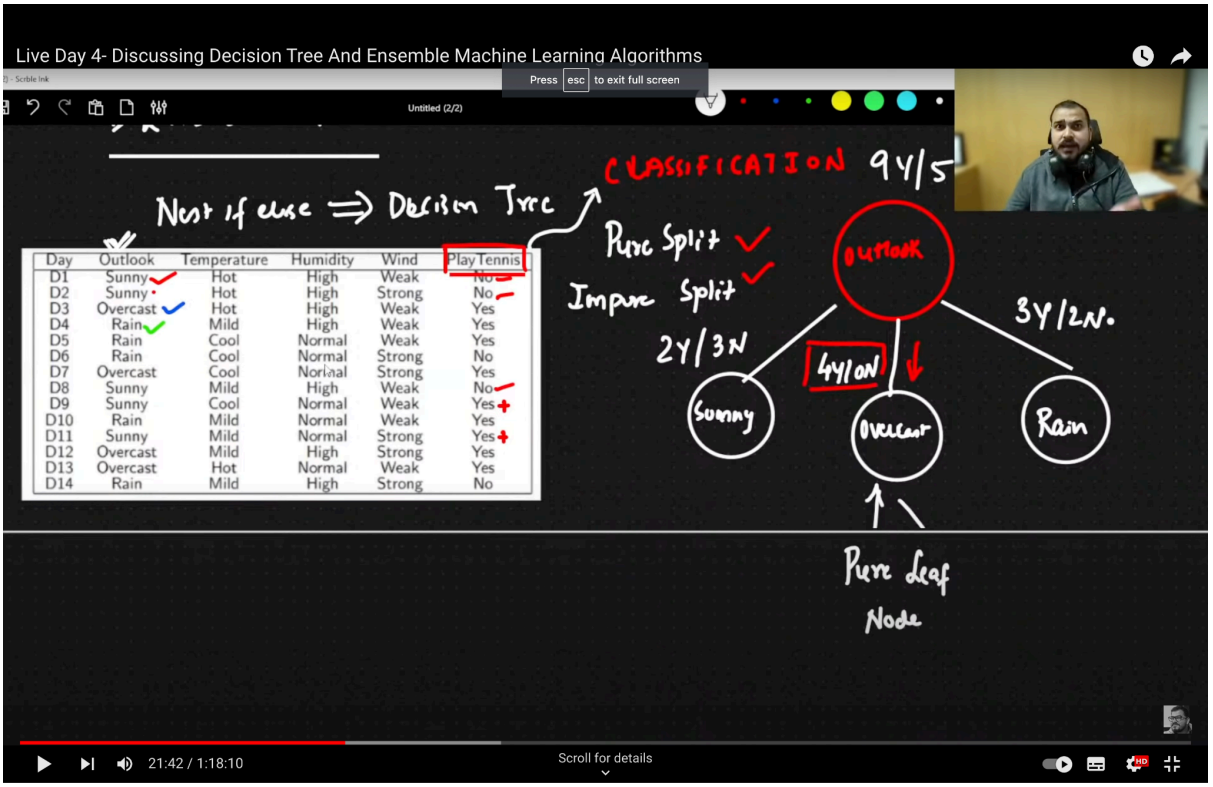
5. Decision Tree Algorithm



Decision Tree for Classification:



- **Pure Split** means we have either **yes** or **no,** in other words 4 yes or 4 no , One sided results.

- We will go ahead to add features and keep splitting until we get pure leaf node.

There are two questions that,

First and foremost, How do we calculate purify and come to know that this is pure split.

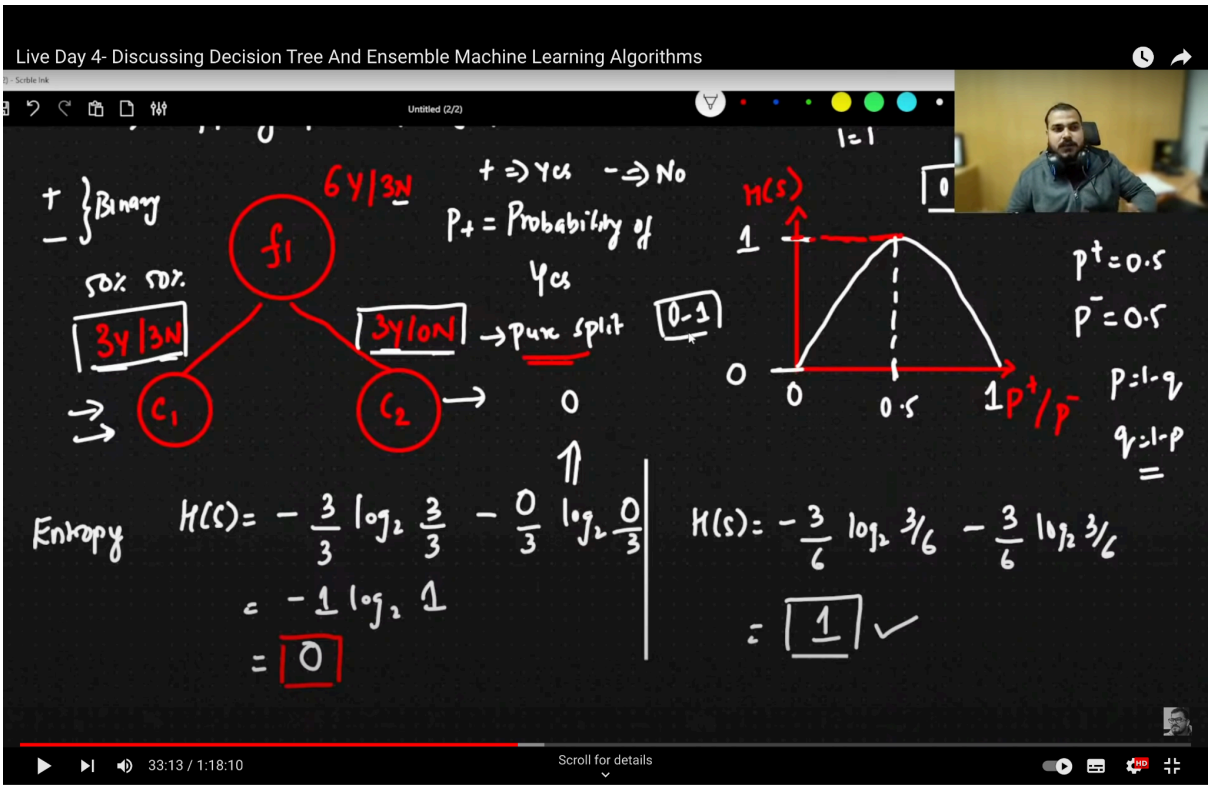
For this, we use two things 1) **Entropy** and 2) **Ginni Coefficient**

Secondly, How we select the features?

For this, we use **Information Gain**

Now, Let's jump on a **Entropy,**

$$H(S) = -P_{+} log_2 P_{+} - -P_{-} log_2 P_{-}$$

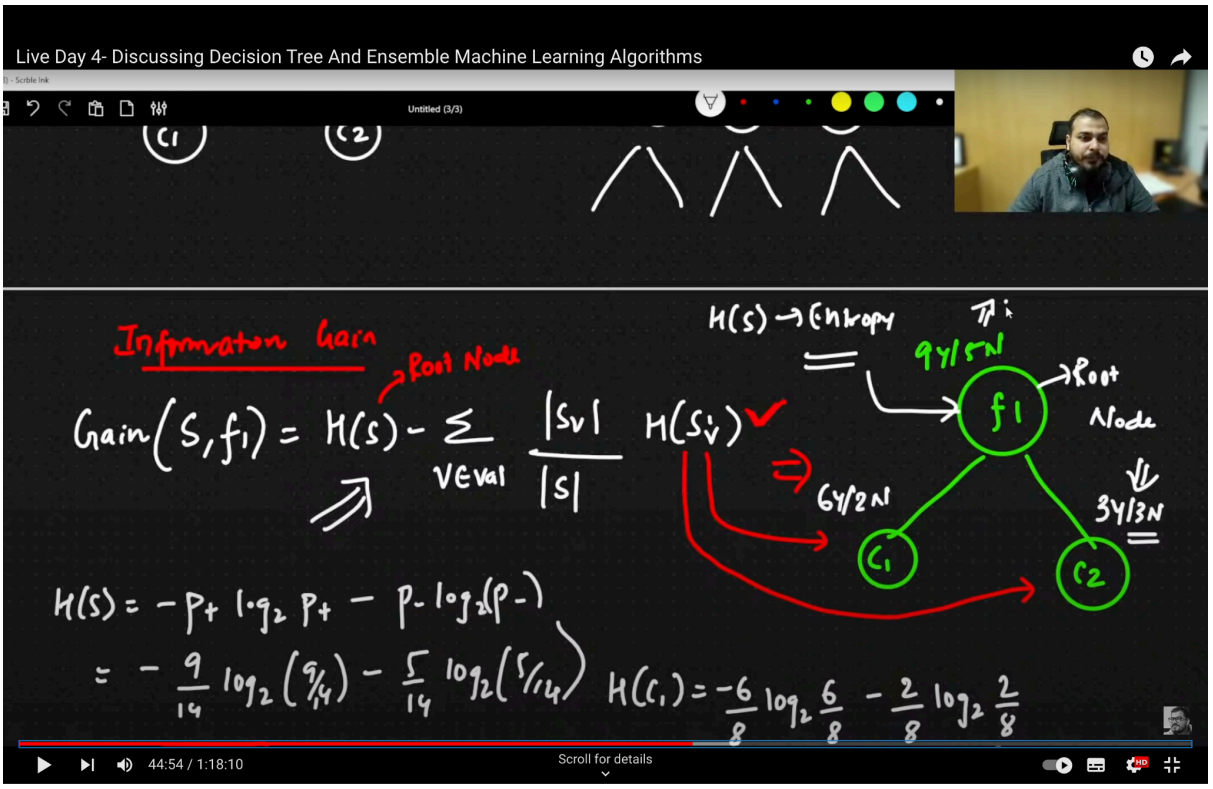


If probability is 0.5 then our Entropy is going to be 1. So, now we will go further and keep adding features and splitting until we get pure leaf node.

If H(s) is **0**, it is a **Pure Split ,while** If H(s) is 1, it is a **impure split** , also our h(s) always between 0 and 1.

Let's solve our Second questions that selecting features,

How can we select feature to start the splitting and go further for that we use **information gain.**



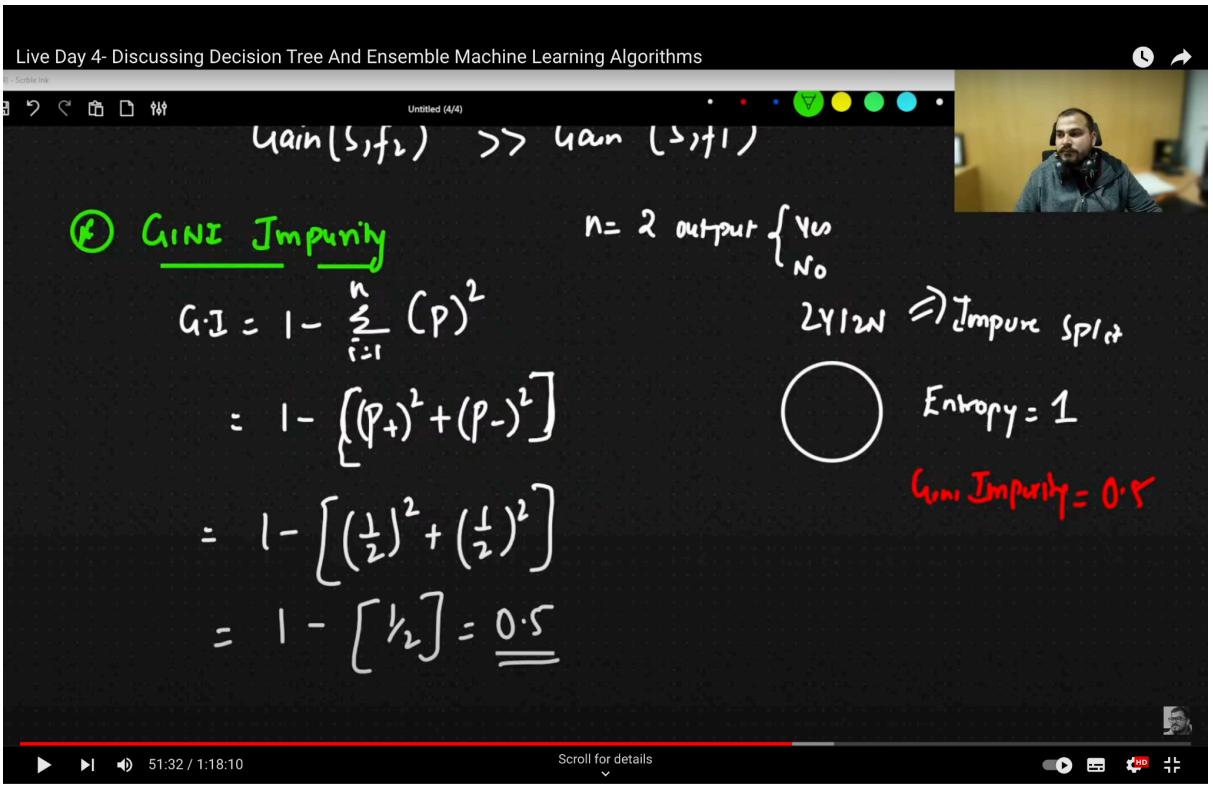
$$\textbf{Gain(Sample, feature1) = H(s) - } \sum_{V \in value} \frac{|s_v|}{|s|} H(s_v^{'})$$

Here, $|Sv|$ is the total of category 1 as shown in above image C1 is 8 and c2 is 6,

Now, $|S|$ total number of sample as shown in above total sample is 14.

Let's say our gain(s,f1) = 0.48 and gain(s,f2) = 0.51,now which features should we consider in order to start the splitting ? Here we will use gain(s,f2) for start the splitting.

Let's discuss **Ginni Impurity/ coefficient/ index**



In Gini Impurity, when we have impure split it will be 0.5, whereas in Entropy it will be 1.

Now, when we can use **Ginni** or **entropy**. Let's consider we have more than 100 features . in that scenario , we must use **Ginni because it has a simple math fromula**, while when we have less features we use entropy as it has **Wrost time complexity and** we use log function.

Decision tree Regresser

Let's consider feature 1 ,we use for splitting then we take the **mean** of the output then either we use **MSE** or **MAE, then** we move to the leaf node where we take sample from output ,follow the same process as we do.

The difference between Decision tree **Classifier** and **Regresser** is that we use ginni and entropy in Classifier, while we use either **MSE** and **MAE** in Regresser**.**

Decision tree always leads to overfitting ,so in order to prevent overfitting we use two techniques 1) **Post Pruning and** 2) **Pre pruning**

1) **Post Pruning**

   Let's say we got 80% yes at some point , should we go further or not ? Answer is **NO, we** will cut the tree from that branch and that is called Post pruing.

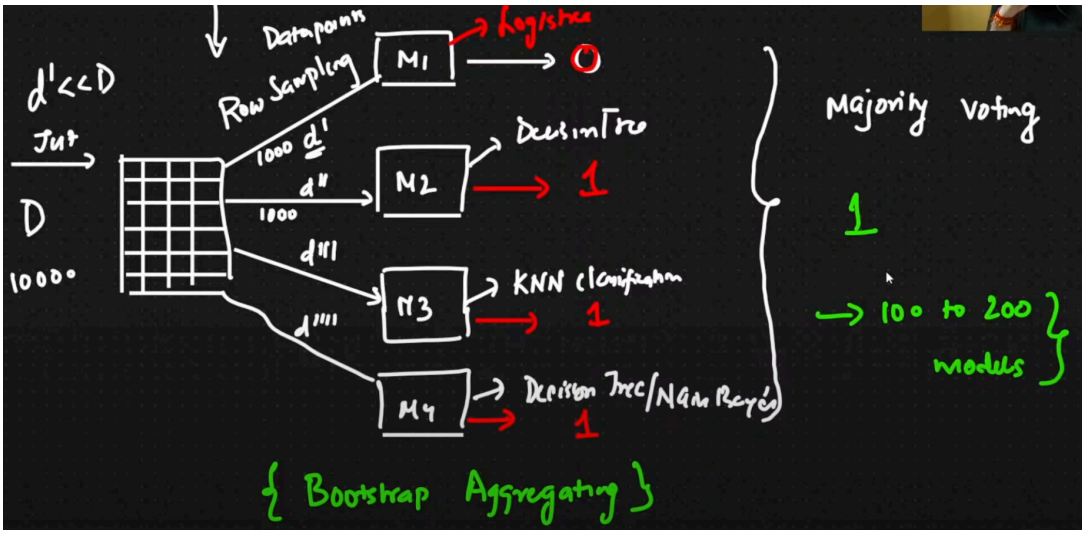
2) **Pre pruning**

   Here, we use some hyperparameter such as max_depth, max_leaf with the help of **GridSearchCv**

6. Ensemble Techniques

There are two methods in ensemble techniques **1) Bagging** and **2) Boosting**

To begin with the **Bagging,** let's say we have 4 models m1,m2,m3,m4 and give raw sampling , here some raws may be similar in models ,We get output 0 in M1 and 1 in others models after that we aggregated all outputs and take the final output which is our majority output here it's **1.** In rare scenario,it will happened that there is no majority in output ,however datasets have more features so this scene will not urge in most cases.This is the scenario for classification, while **in regression** we take the **mean of output of each model** and thats the final output.



In the **Boosting,** There is a sequential of models and our sample data pass through all models and in the initial stage this all models are weak learner, So when we combine all the models it becomes strong learner. For Example, we have a question and there are model 1 that is physics professor, model 2 that is Chemistry professor and model 3 that is maths professor , now that problem pass through all models and in the end we got our output in the efficient way.

In **Bagging,** we use Random Forest Classifier and Random Forest Regresser, **while** in **Boosting** we use **AdaBoost, Gradient Boost, and XGboost (**also known as **Extream gradient boost).**

1) **Random Forest Classifier and regresser:**

Generally, when we use default decision tree (without hypertuning) it may lead us to the overfitting and it means **Low bias** and **high variance. Now,** our target is to convert **High variance** into **Low Variance.** It possible with the help of random forest classifier.

What random forest do ?

Let's say we have 4 models m1,m2,m3,m4 and these all models are actually decision trees.

If we combine in the form of bootstrap aggregator this high variance will convert into low variance. There will be many decision trees and with majority of voting output high variance converted into low variance.

Now, How it works?

Random forest classifier takes raw and features sampling from dataset and give to model 1 for training likewise it does with model 2 ,3, 4 and so on. Here, not only raw data but also features can be repeated in models and thats how particular models will be got expertise to predict something. We give some test data to random forest and it majority of voting will be considered as output.

Now there are two questions **first, whether Is normalization necessary in Random forest or Not?**

Answer is that **No** because Decision tree do splitting and if we minimize our data that split won't be that much important.

Second ,whether normalization or standardlization is required in KNN?

Answer is **Yes** as we use Two things 1) Euclidean distance 2) Manhattan Distance. We have to use these for making computation or distance easy.

Third question, Is random forest is impacted by outlinears?

Answer is the it is less impacted by outlinears as it get average of multiple decision trees.

Now, the difference between random forest classifier and random forest regresser is that in classifier we take the majority of votes, while in regresser we take the average of all the decision tree's output.

Boosting

2) **AdaBoost:**

Let's say we have four features f1,f2,f3,f4 ,and output feature's value yes and no. In adaboost, first we define weights and initially for all input records we provide equal weights.**So,How do we provide it?** Let's say we have 7 inputs and sum of all equal weights should be between 0 and 1, so here we take weights 1/7 for all 7 inputs as sum of all 7 inputs will be 1.

After that, We take any features through information gain and entropy or ginni. Now, let's say we got feature 1 is higher, so we start making nodes for decision tree. The depth of this decision tree will be only **one, so** we call it **stumps, and** this is a weak learner.

We provide all inputs record to feature 1 in order to check how many input records are correct, and we got one input raw wrong. We are going to calculate the total error this particular model made. We have one wrong and its weight is 1/7.

This was our first step, now in our second step we calculate the performance of stump. In order to do that we use formula for it which is,

$$= \frac{1}{2} Log_e\left(\frac{1-TE}{TE}\right)$$

We have Total Error(TE) is 1/7 so final answer will be **0.895.**

Now, our third step is to update all weights and we have new sample weight from stump. If i want to update the sample weight , first i have to update weights of all correct records. When we update it ,weight for all correct records should reduce, while for wrong records it will increase. **Why?** If wrong records weights are increased, those records will go further to weak learner.

So, we change weights for **correct records**, there is a formula for it which is

$$Weights * e^{-PS}$$, PS means Performance Stump

Final value weight for correct records is **1/7 * e^-0.895** which is **0.05,** So new weights for all correct records are 0.05

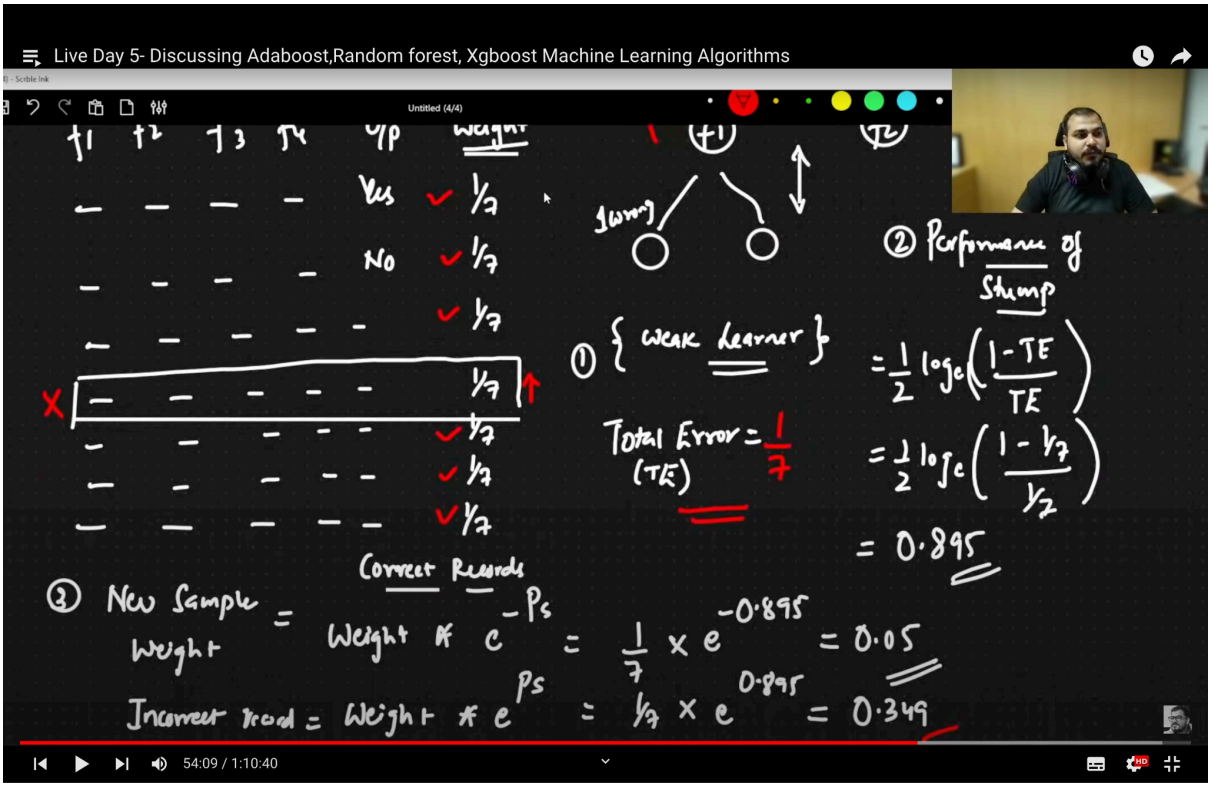
Weight change for **Wrong records**, formula is ,

$$Weights * e^{PS}$$,

Final value weight for wrong records is **1/7 * e^0.895** which is **0.349,** So new weights for all wrong records are 0.349.

We considered 7 records so as we had 6 correct records and 1 wrong records and if we do summation we got 0.649 which is definitely not equal to 1, **So what we do?**

We will normallize our new Weights, we will divide all new weights with their summation which is 0.649 and then we sum all normalized weights which will be nearly or eqal to 1.

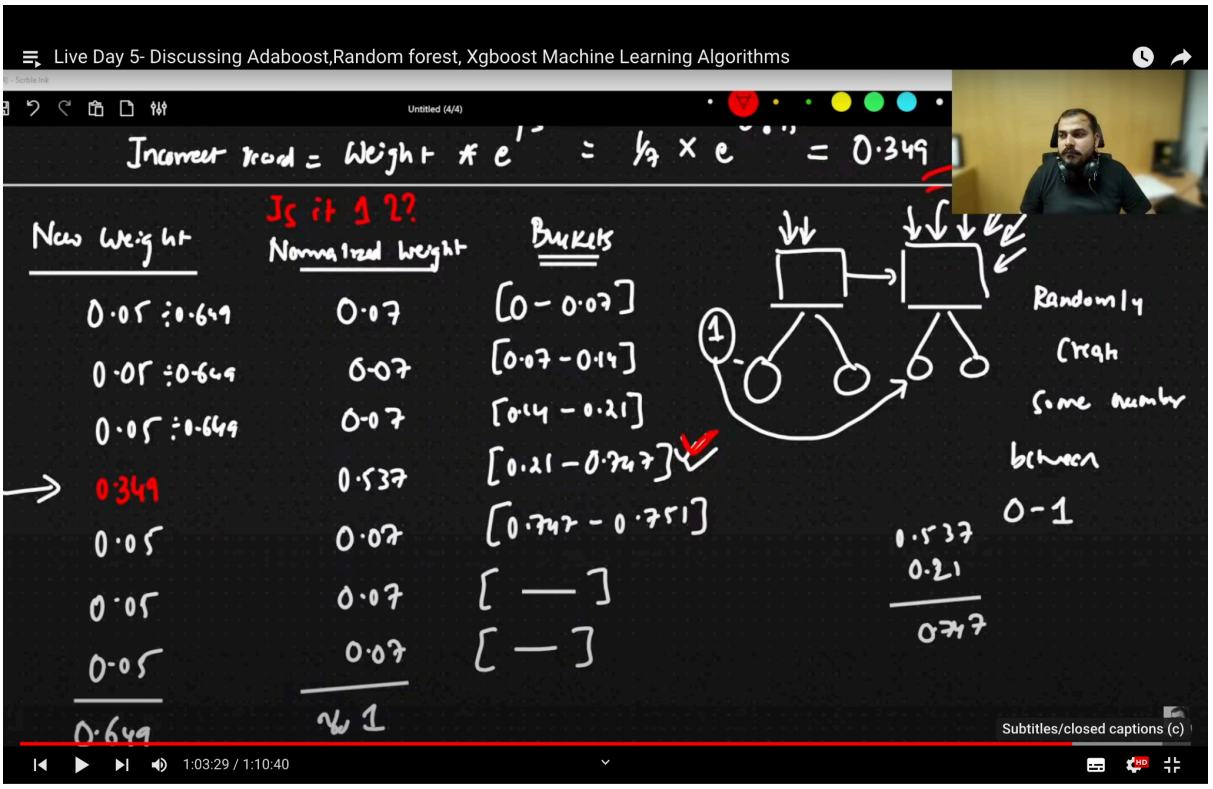


After Normalization, we will creat buckets, then we got output of our stumps(model 1 or first decision tree), now we creat or add new model in sequence as Adaboost is boosting technique.

In order to creat new sequential model, we will provide some specific raws along with those wrong raws from previous models. Usually, in new model ,those data will go which were wrong in previous model and thats possible by creating random numbers between 0 and 1 which will be done by models.

Let's see about **Buckets,**

In Buckets, we start from 0 and go further adding normalized weights like shown in below image,



Likewise, we will generate more weak learners or models that can be **more than 100.**That's why it has **more Time complexity**.

Each models will give output that could be either **yes/No** or any **continuous values**. As usual **for classification** we take majority of output, whereas **for regresser** we will take an mean of all sequential model's output.