REVISITING ML

Today I will be starting with Decision Tree
Before that lets just revise what is bias and Variance
Variance vs Bias:

///////
Bias

- **Bias** is when a model makes consistent mistakes.
- Think of it as a model that is too simple to understand the real pattern.
- **High Bias** Example: Trying to fit a straight line to data that curves.
    - The model can't capture the curve, so it performs poorly both on training data and new data.
    - This is called **underfitting**.

Variance

- **Variance** is when a model changes a lot with small changes in the training data.
- Think of it as a model that is too sensitive to the specific details of the training data.
- **High Variance** Example: Trying to fit a complex curve to data that is actually linear.
    - The model captures too much detail, including noise, and performs well on training data but poorly on new data.
    - This is called **overfitting**.

The Bias-Variance Tradeoff

- **Tradeoff**: You need to balance bias and variance to get a good model.
    - A simple model (high bias) might miss patterns (underfit).
    - A complex model (high variance) might capture noise (overfit).
- **Goal**: Find a model that is just right"not too simple, not too complex.

Visual Example

- Imagine trying to hit the bullseye on a dartboard:
    - **High Bias**: All darts land far from the bullseye in the same spot.
    - **High Variance**: Darts are all over the place, some near the bullseye, some far.
    - **Low Bias and Low Variance**: Darts land close to the bullseye in a tight cluster.

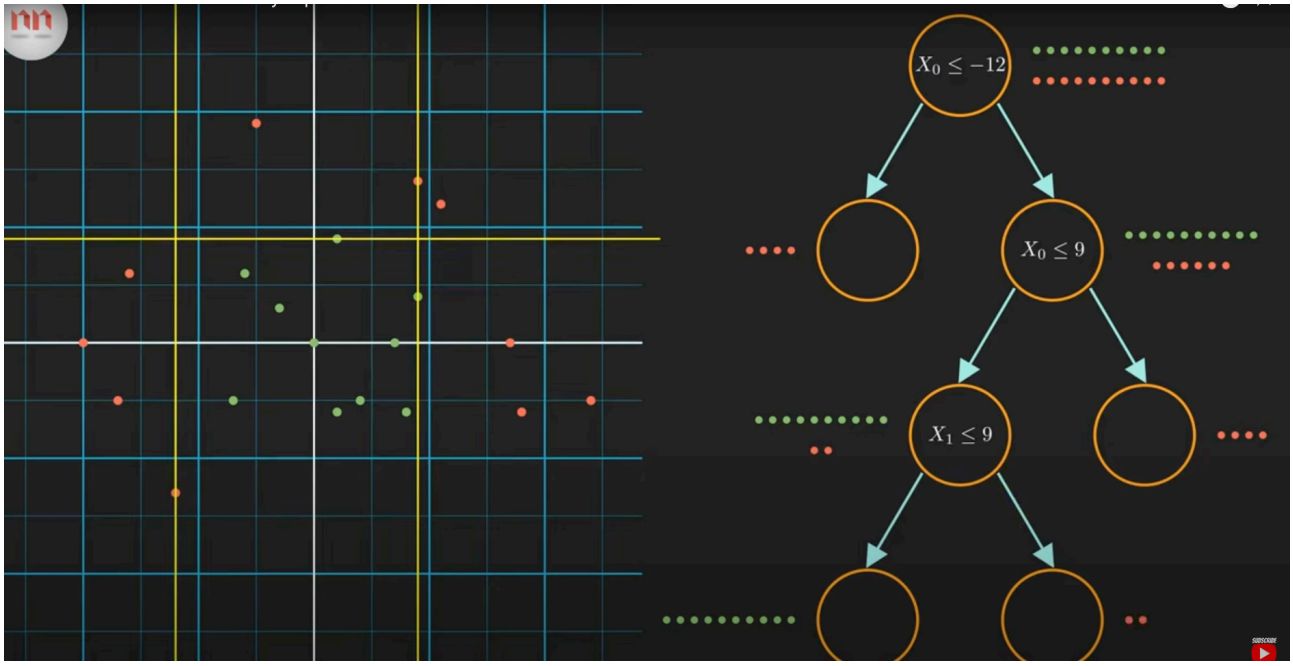
///////

Lets start with decision tree

Decision trees are a type of model used for both classification and regression. Trees answer sequential questions which send us down a certain route of the tree given the answer. The model behaves with "if this than that" conditions ultimately yielding a specific result.

We would be seeing this with an example , let us suppose we have these points

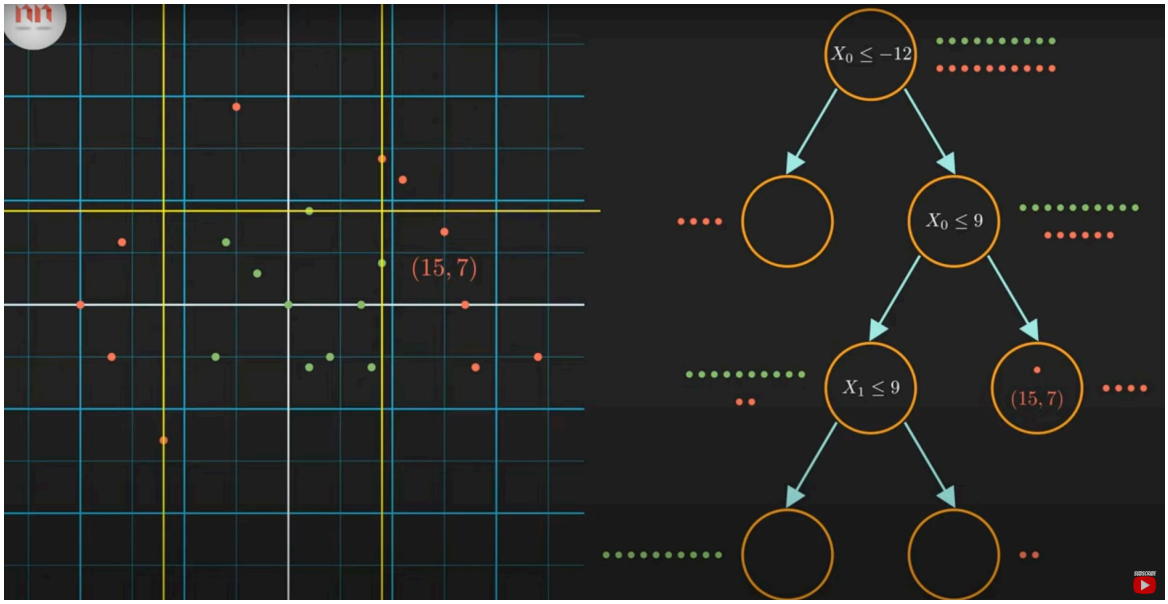
We made a decision trees on certain conditions as you can see on right (points that matches



condition go to right on every step) and on basis of these conditions we kept on dividing the graph until we saw some similarities as on the children node we could see that we bifurcated all the green and red dots.

So if we are given some random point to predict if it would have red or green dot we could do it easily.

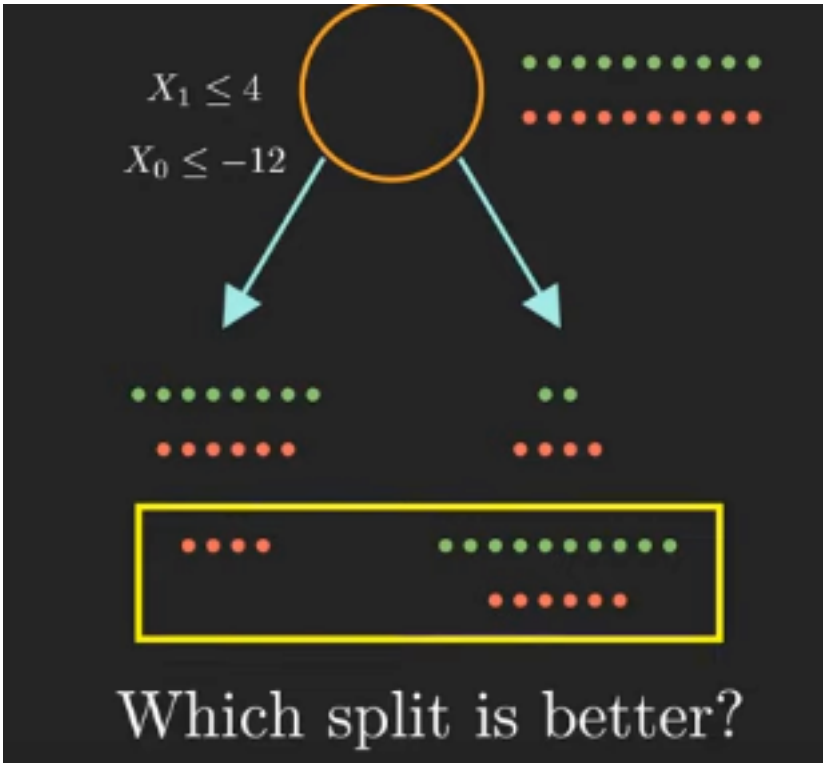
As you can see the (15,7) based on the conditions in the tree comes out to be on right most child node where there are all reds so it would most likely have a red dot while classifying.



OUR MAIN OBJECTIVE IS TO GET PURE LEAF NODES

NOW QUESTION ARISES HOW DO WE GET OPTIMAL SPLITS AS GIVEN IN THE EXAMPLE LIKE HOW DID WE CAME UP WITH THE VALUES (-12, 9) etc?

Lets start at the root node we would compare to random values just for example take it



(image a)

Based on this picture we could see that second split is better as out of two one of the leaf node is pure unlike split one having no pure nodes.

Machine can do this with Information theory like the split that maximises information gain is chosen.

Like talking about this case it has equal green and red dots, imagine predicting class of randomly picked point only half of time we would be correct so it is highly impure and uncertain.
Way to quantify this is to use entropy

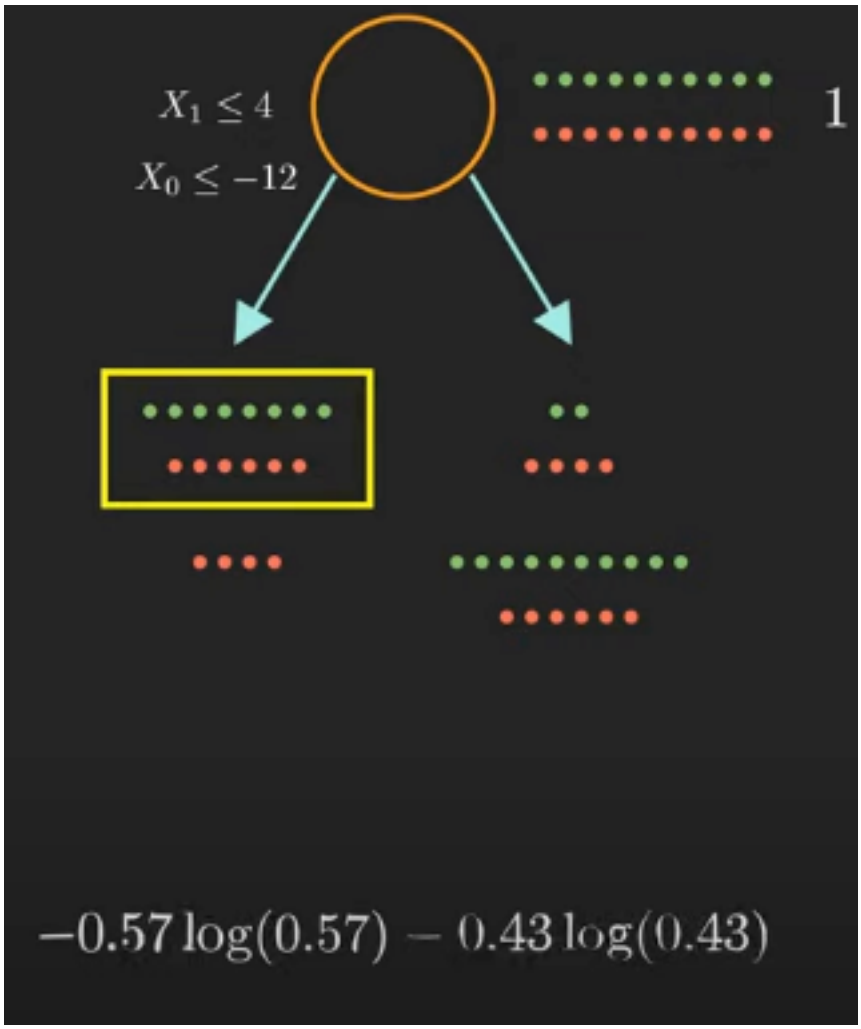
ENTROPY IS THE MEASURE OF INFORMATION CONTAINED IN THE STATE

Entropy high— — — — —> MOST UNCERTAIN

$$Entropy = \sum - \ p_i \ \log(p_i)$$

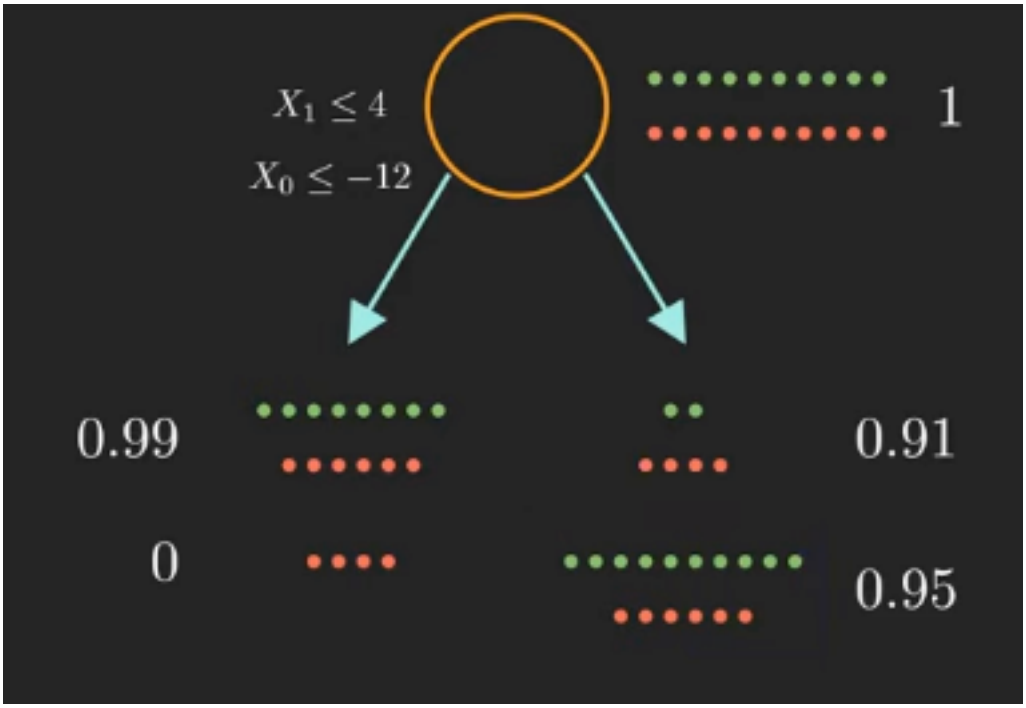
p_i = probability of class i

Based on this formula we can calculate entropy of every split from image a*



0.57 is probability of green and 0.43 is probability of getting red dot.
REMEMBER entropy can never exceed 1.

Calculating entropies of every split:



See as we predicted the purest node showed minimum entropy (0)
Now we need to subtract combine entropy of child from that of the parents node.

$$\text{IG} = E(parent) - \sum w_i \; E(child_i)$$

We have also taken the weights for the child node, thats nothing but relative size of child to that of parents.

Let's calculate the INFORMATION GAIN from the entropies

$$\text{IG} = E(parent) - \sum w_i \; E(child_i)$$

$$\text{IG}_1 = 1 - \frac{14}{20} \times .99 - \frac{6}{20} \times .91 = 0.034$$

$$\text{IG}_2 = 1 - \frac{4}{20} \times 0 - \frac{16}{20} \times .95 = 0.24$$

We can see the second split gives greater Information Gain so we would choose the second one.

$$IG2 > G1$$

Now we have just compared 2 splits to understand the working but in reality the model takes in account every possible split to maximise the information gain and find the optimum split.

DECISION TREE IS A GREEDY ALGO it selects current best split and does not backtrack and change the previous state. So all following splits would depend on current one and doesn't guarantee the optimal split.