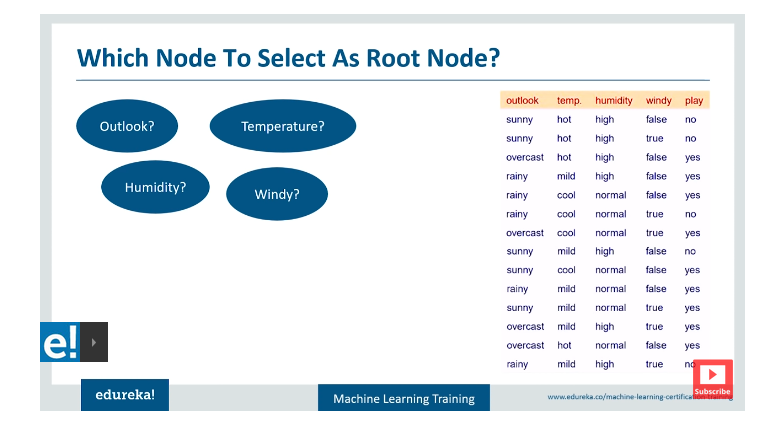
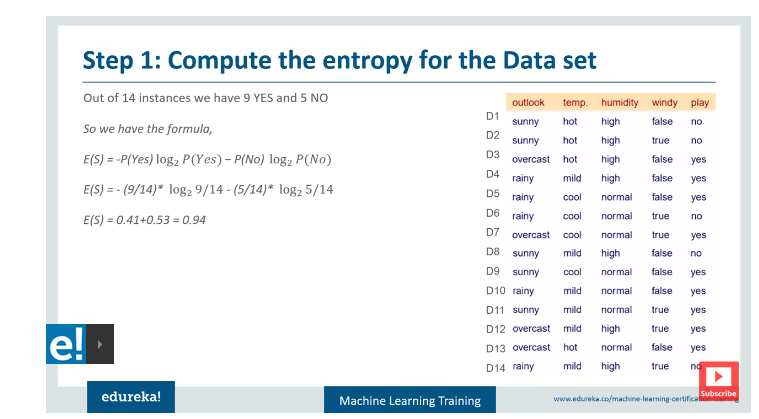
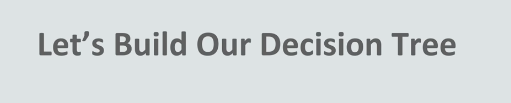
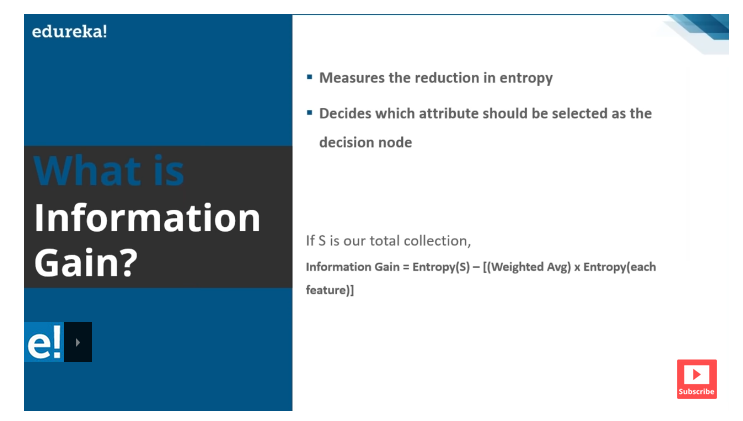
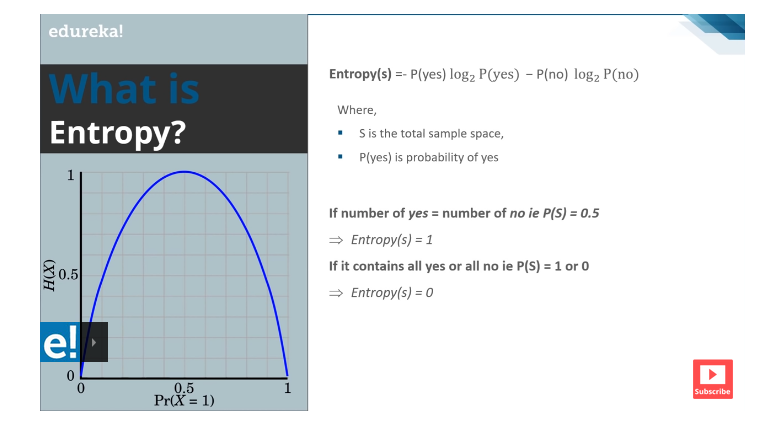
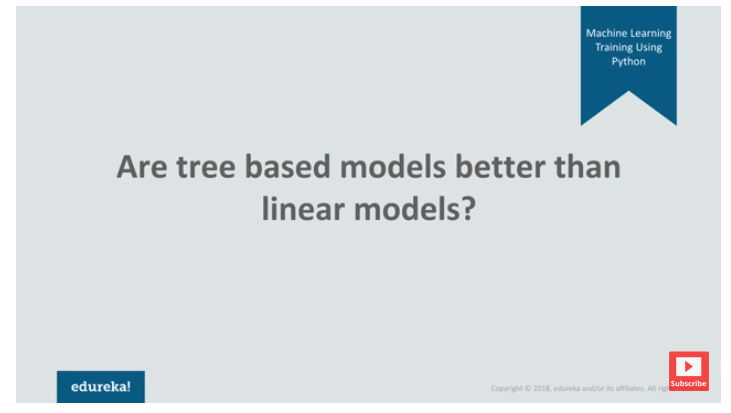
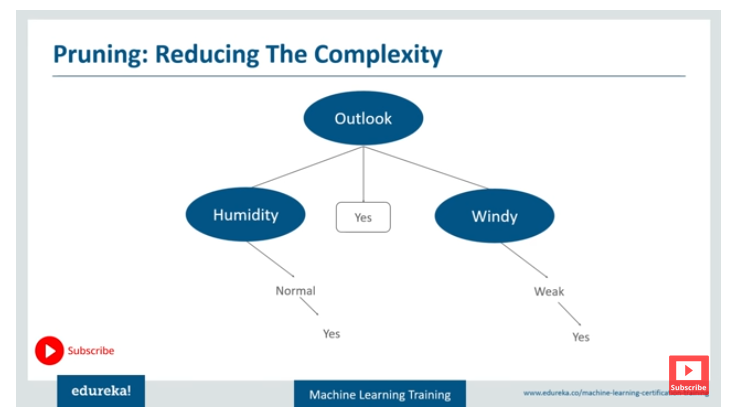
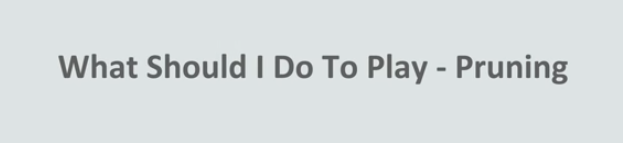
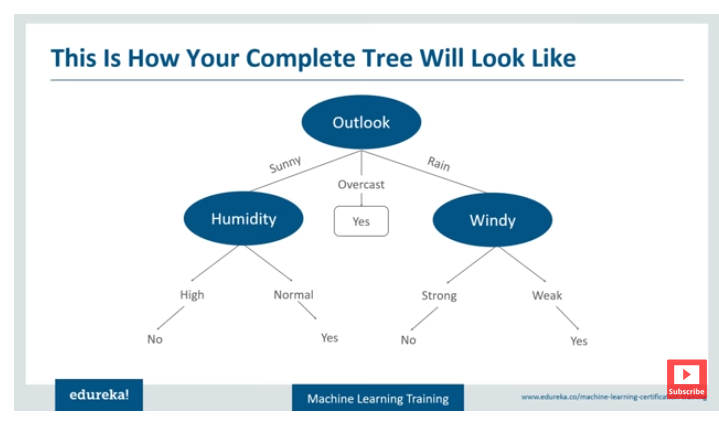
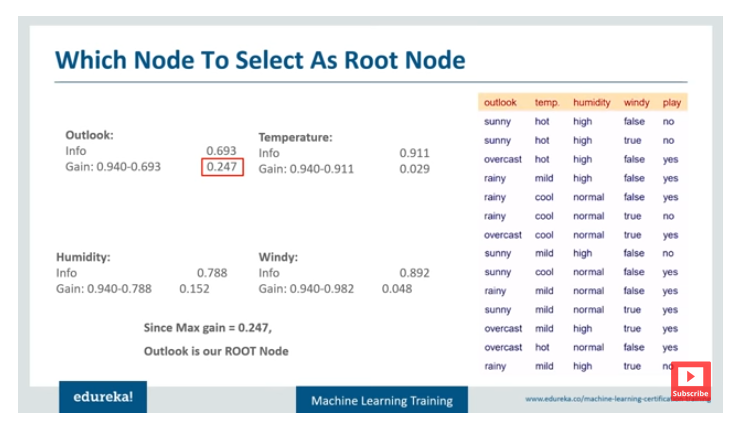
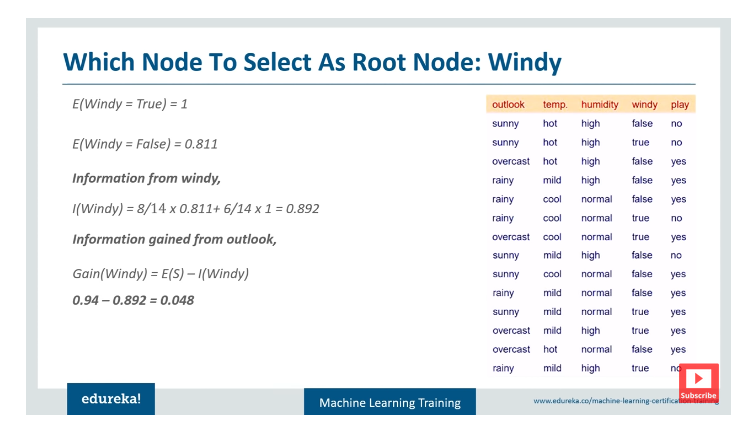
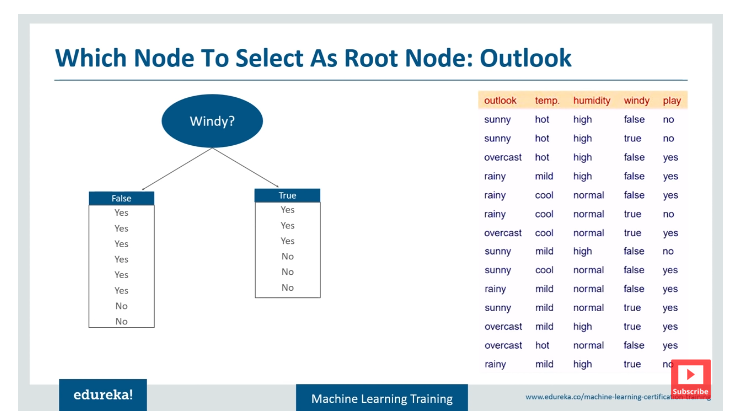
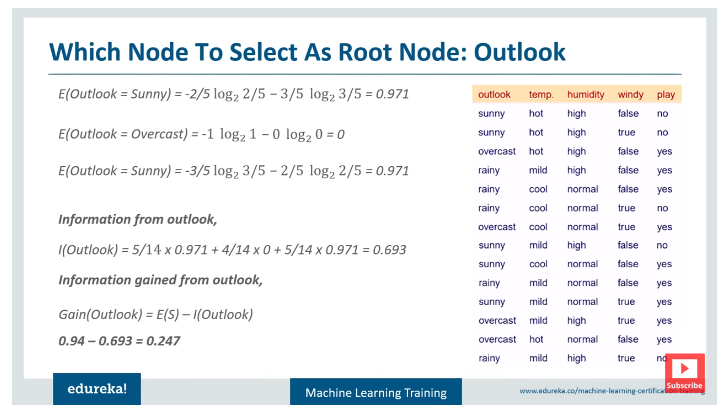
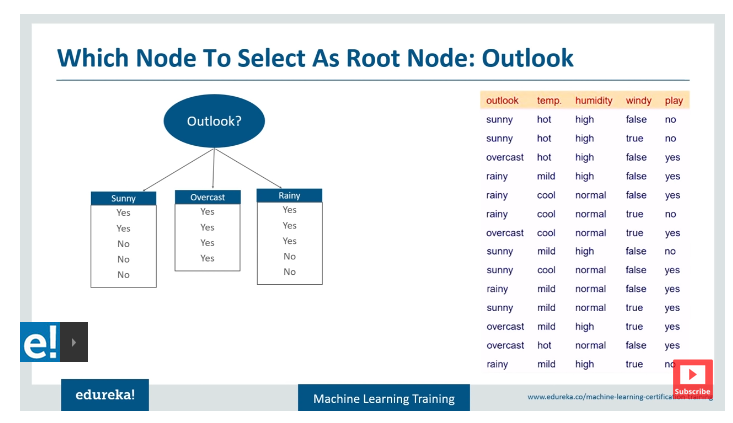
Decision tree can be used for regression as well as for classification but more popular for classification

**Target is to find smaller tree that fit the training data having Small number of nodes and depth**

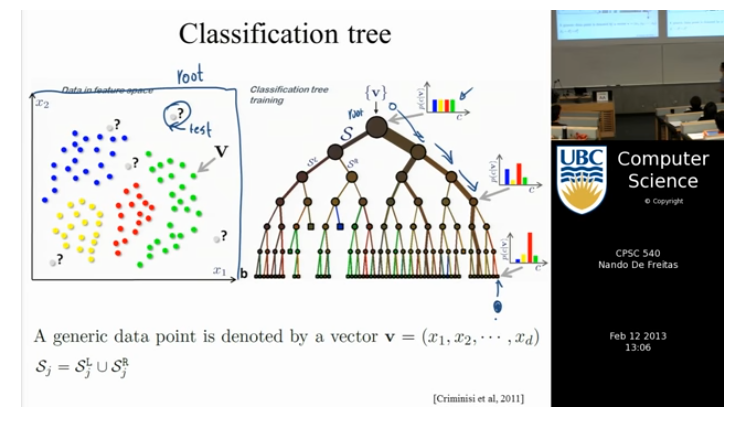
**Uses Greedy Algorithm to create decision tree**

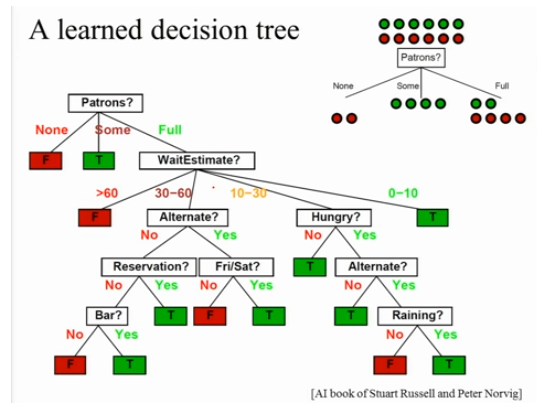
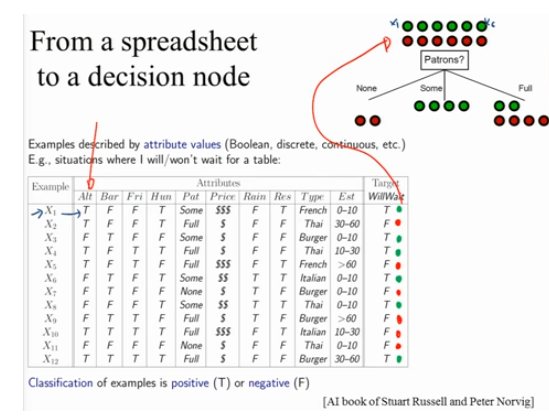






If the relationship between dependent and independent variables is approx linear then go for Linear if relationship is complex then tree model will do better.



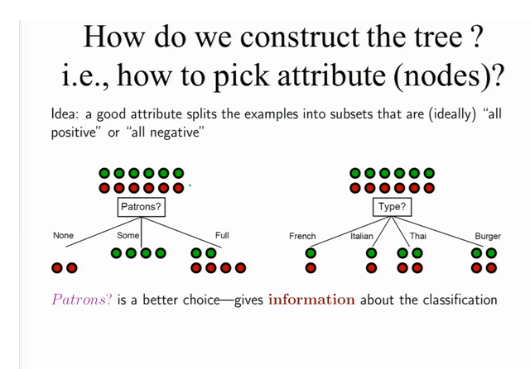
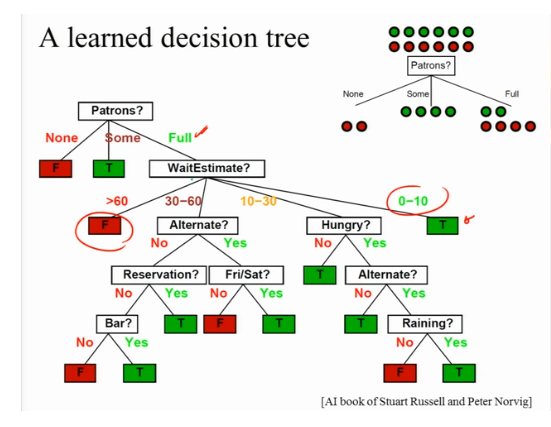


Decision tree very well handle categorical data, discrete data (like range ex. wait time 10 -15 mins age etc), categorical data, unordered data( like type of restaurant).

Decision tree are build in Greedy fashion.(top down approach).

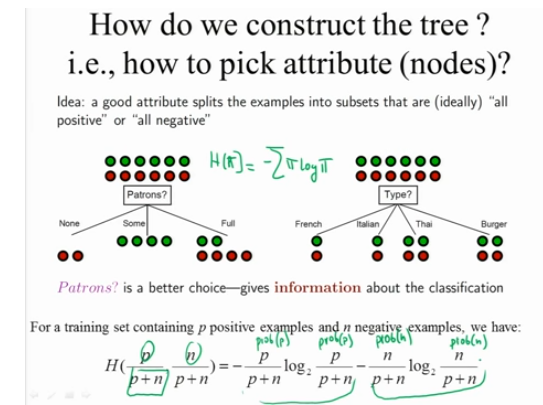
There are several of possibilities to build decision trees on the basis of selection of nodes.

That is where Random forest comes in to picture to take average of all the decision tress build.

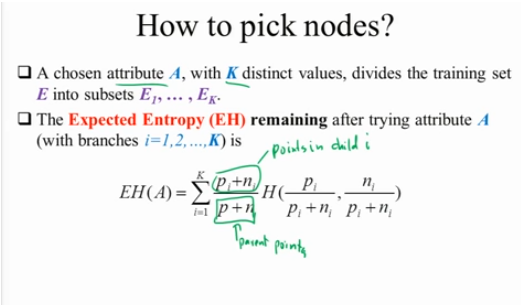


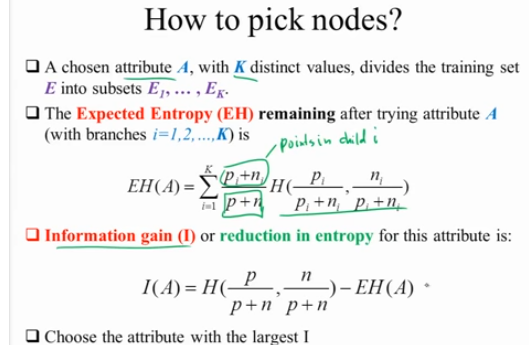
Entropy is measure of information or disturbance.

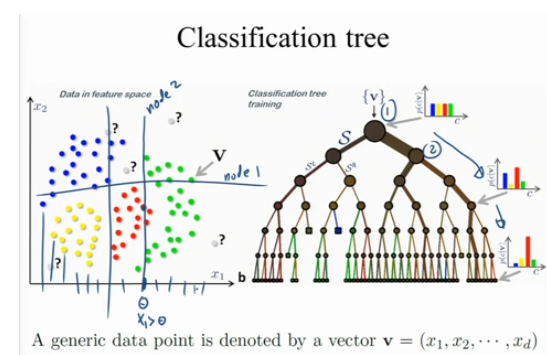
For binary classification , there will be only two outcomes so entropy will be.

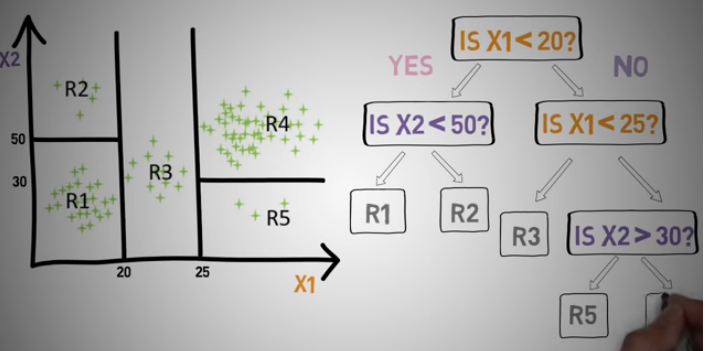


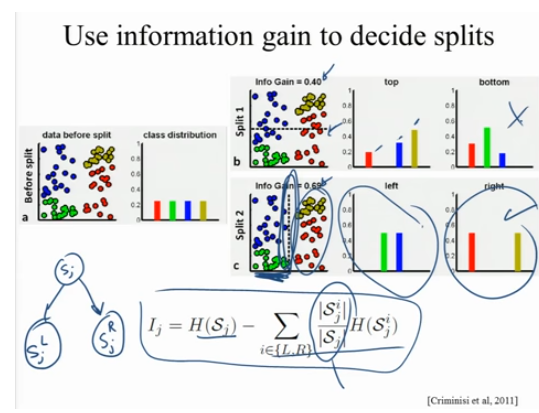
For more them two classes we change the formula accordingly.





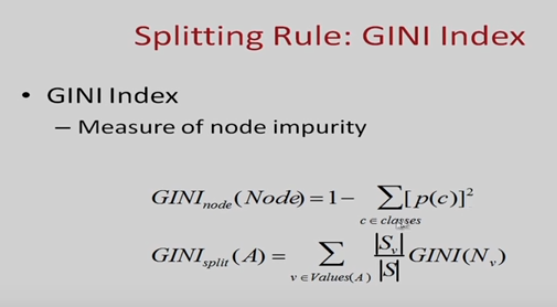




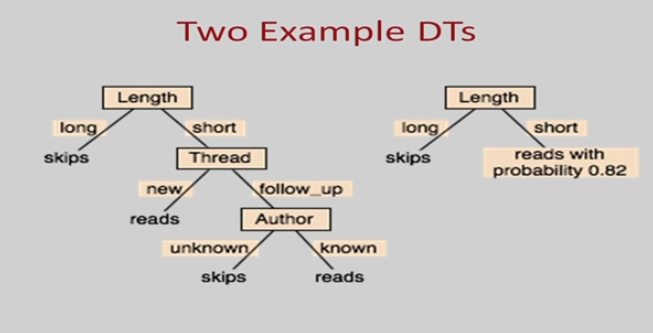


We will chose which dimension to choose if there are hundreds of variables.

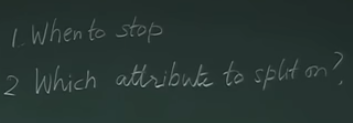
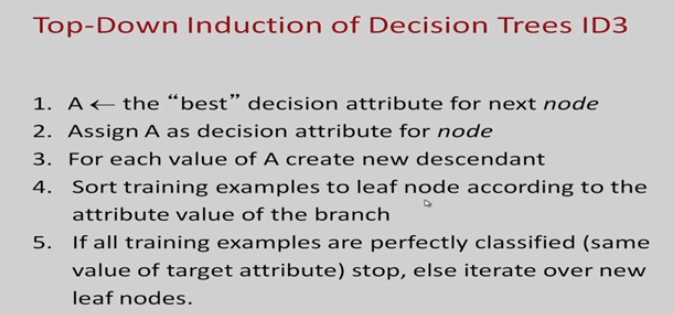
Other then information gain we can use another measures to choose attribute for splitting as gini index :



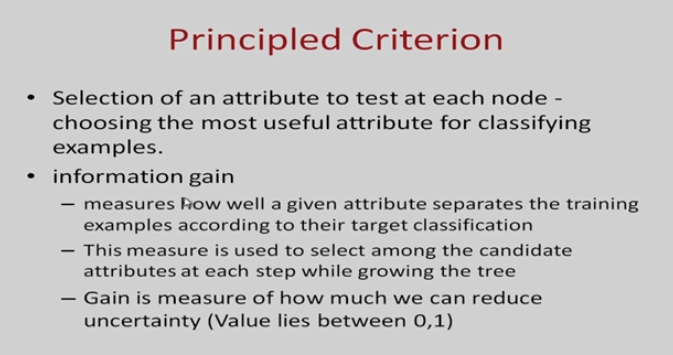
==================================================

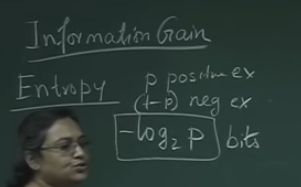


We have the choice the grow the tree when there are more than one possible values or stop if one possible value probability is higher than others as in above example.



If all the input training set has single value at any node then stop or split the node on any attribute to go deeper in to decision tree.





In information theory if optimum length code assigned is –logp base 2 when there are positive examples and (1-p) negative examples.

Entropy:

https://www.khanacademy.org/computing/computer-science/informationtheory/moderninfotheory/v/information-entropy

**Information entropy** is the [average](https://en.wikipedia.org/wiki/Expected_value) rate at which [information](https://en.wikipedia.org/wiki/Information) is produced by a [stochastic](https://en.wikipedia.org/wiki/Stochastic) source of data.

The measure of information entropy associated with each possible data value is the negative [logarithm](https://en.wikipedia.org/wiki/Logarithm) of the [probability mass function](https://en.wikipedia.org/wiki/Probability_mass_function) for the value:

S = − ∑ i P i log ⁡ P i S=-\sum \_{i}P\_{i}\log {P\_{i}} . [[1]](https://en.wikipedia.org/wiki/Entropy_(information_theory)#cite_note-pathriaBook-1)

When the data source has a lower-probability value (i.e., when a low-probability event occurs), the event carries more "information" ("surprisal") than when the source data has a higher-probability value

The logarithm of the probability distribution is useful as a measure of entropy because it is additive for independent sources. For instance, the entropy of a fair coin toss is 1 bit, and the entropy of *m* tosses is *m* bits. In a straightforward representation, log2(*n*) bits are needed to represent a variable that can take one of *n* values if *n* is a power of 2. If these values are equally probable, the entropy (in bits) is equal to this number

Consider tossing a coin with known, not necessarily fair, probabilities of coming up heads or tails; this can be modelled as a [Bernoulli process](https://en.wikipedia.org/wiki/Bernoulli_process).

The entropy of the unknown result of the next toss of the coin is maximized if the coin is fair (that is, if heads and tails both have equal probability 1/2). This is the situation of maximum uncertainty as it is most difficult to predict the outcome of the next toss; the result of each toss of the coin delivers one full [bit](https://en.wikipedia.org/wiki/Bit) of information. This is because

H ( X ) = − ∑ i = 1 n P ( x i ) log b ⁡ P ( x i ) = − ∑ i = 1 2 1 2 log 2 ⁡ 1 2 = − ∑ i = 1 2 1 2 ⋅ ( − 1 ) = 1 {\displaystyle {\begin{aligned}\mathrm {H} (X)&=-\sum \_{i=1}^{n}{\mathrm {P} (x\_{i})\log \_{b}\mathrm {P} (x\_{i})}\\&=-\sum \_{i=1}^{2}{{\frac {1}{2}}\log \_{2}{\frac {1}{2}}}\\&=-\sum \_{i=1}^{2}{{\frac {1}{2}}\cdot (-1)}=1\end{aligned}}}

However, if we know the coin is not fair, but comes up heads or tails with probabilities *p* and *q*, where *p* ≠ *q*, then there is less uncertainty. Every time it is tossed, one side is more likely to come up than the other. The reduced uncertainty is quantified in a lower entropy: on average each toss of the coin delivers less than one full [bit](https://en.wikipedia.org/wiki/Bit) of information. For example, if *p*=0.7, then

H ( X ) = − p log 2 ⁡ ( p ) − q log 2 ⁡ ( q ) = − 0.7 log 2 ⁡ ( 0.7 ) − 0.3 log 2 ⁡ ( 0.3 ) ≈ − 0.7 ⋅ ( − 0.515 ) − 0.3 ⋅ ( − 1.737 ) = 0.8816 < 1 {\displaystyle {\begin{aligned}\mathrm {H} (X)&=-p\log \_{2}(p)-q\log \_{2}(q)\\&=-0.7\log \_{2}(0.7)-0.3\log \_{2}(0.3)\\&\approx -0.7\cdot (-0.515)-0.3\cdot (-1.737)\\&=0.8816<1\end{aligned}}}

The extreme case is that of a double-headed coin that never comes up tails, or a double-tailed coin that never results in a head. Then there is no uncertainty. The entropy is zero: each toss of the coin delivers no new information as the outcome of each coin toss is always certain.