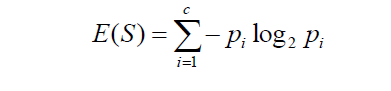
Entropy Calculation

<https://www.google.com/search?client=firefox-b-e&q=Entropy+in+statistics#fpstate=ive&vld=cid:84bb2e52,vid:YtebGVx-Fxw>

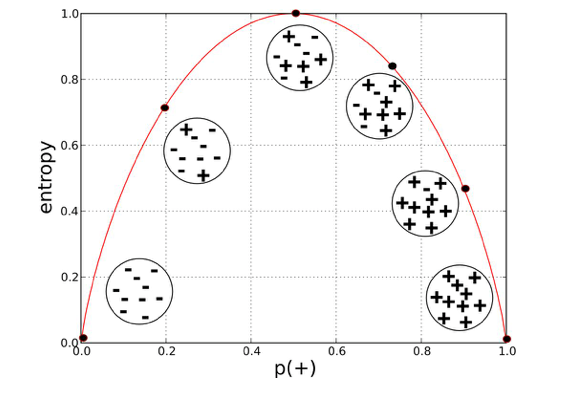
Entropy is measure of impurity of data. Entropy for Pure sample should be zero and also zero for class whose probability is 0.

The Mathematical formula for Entropy is as follows -

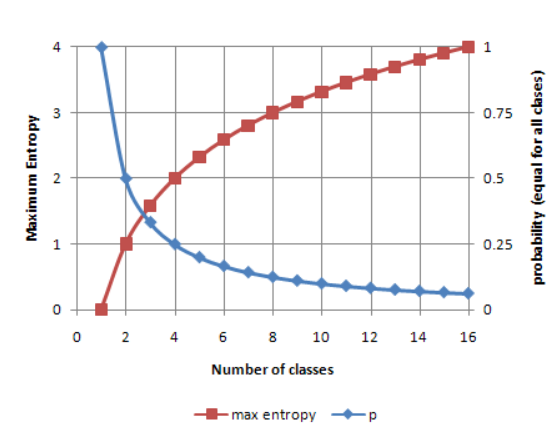


Negative sign as we had log(1/P) and log rules becomes -log(P)

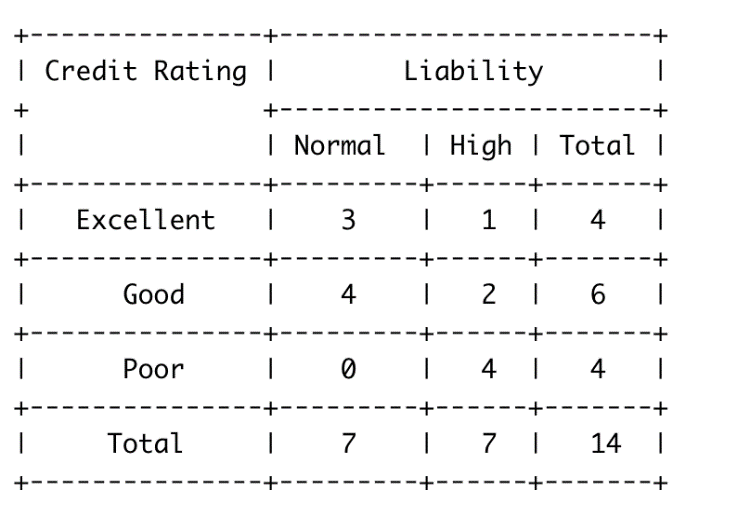
The Entropy for data have two classes only (+, -)



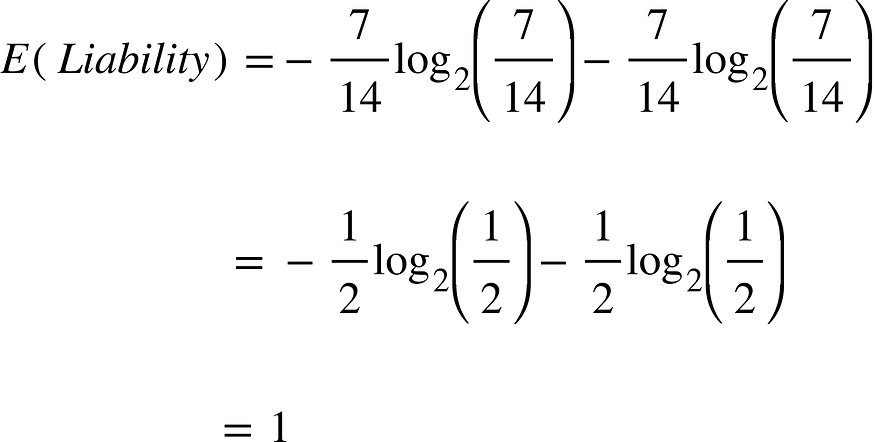
When number of class increases Total Entropy value also increases, if we have 4 classes entropy will be 2. And so on. As all classes have equal probability, the individual probability decreases when class increases.



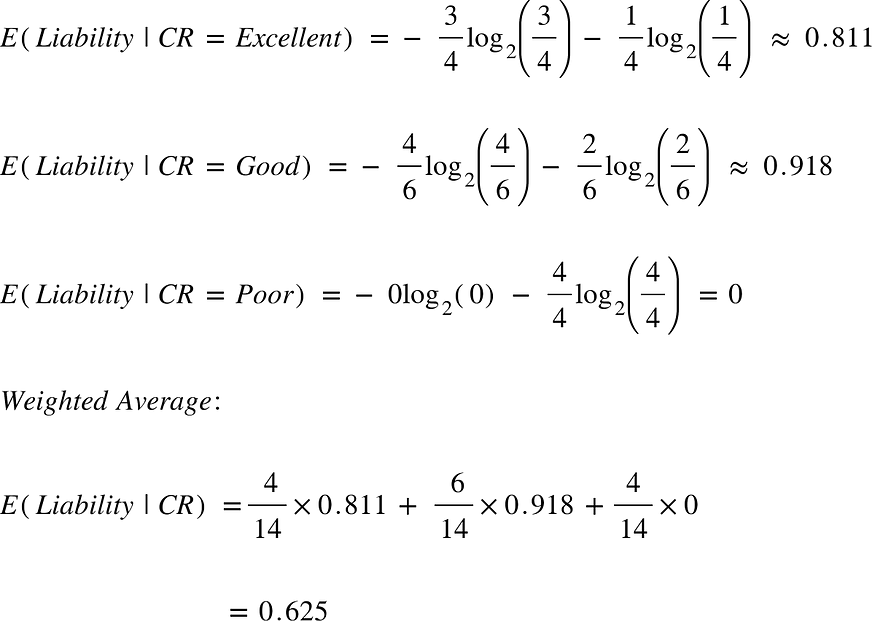
**Example: Contingency Table**



Target variable is Liability and credit Rating is input



The entropy of our target variable is 1, at maximum disorder due to the even split between class label “Normal” and “High”. Our next step is to calculate the entropy of our target variable Liability given additional information about credit score. For this we will calculate the entropy for Liability for each value of Credit Score and add them using a weighted average of the proportion of observations that end up in each value. Why we use a weighted average will become clearer when we discuss this in the context of decision trees.

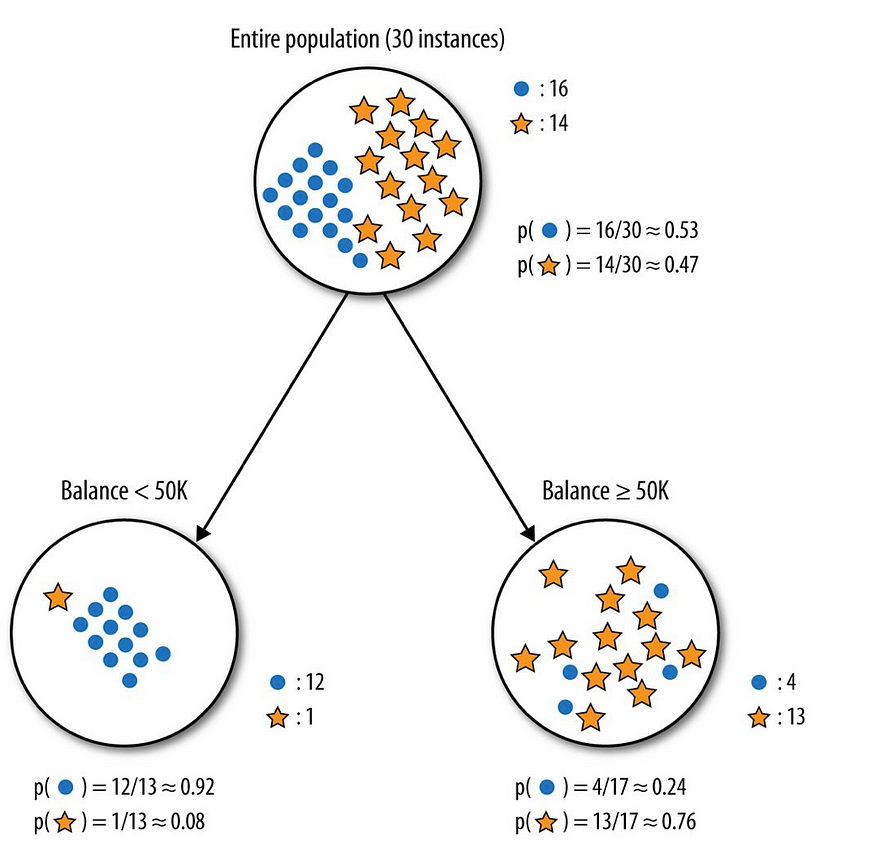


We got the entropy for our target variable given the feature Credit Rating. Now we can compute the Information Gain on Liability from Credit Rating to see how informative this feature is.

In Decision Tree

Consider an example where we are building a decision tree to predict whether a loan given to a person would result in a write-off or not. Our entire population consists of 30 instances. 16 belong to the write-off class and the other 14 belong to the non-write-off class. We have two features, namely “Balance” that can take on two values -> “< 50K” or “>50K” and “Residence” that can take on three values -> “OWN”, “RENT” or “OTHER”. I’m going to show you how a decision tree algorithm would decide what attribute to split on first and what feature provides more information, or reduces more uncertainty about our target variable out of the two using the concepts of Entropy and Information Gain.

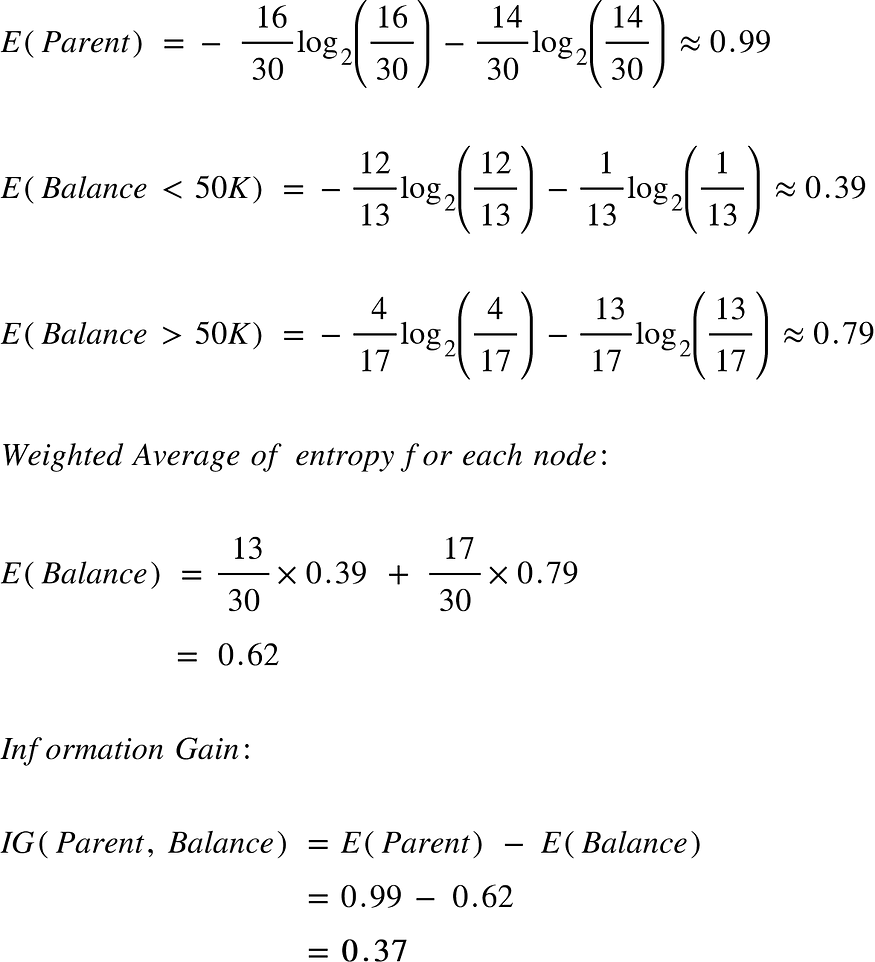
**Feature 1: Balance**



Provost, Foster; Fawcett, Tom. Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking

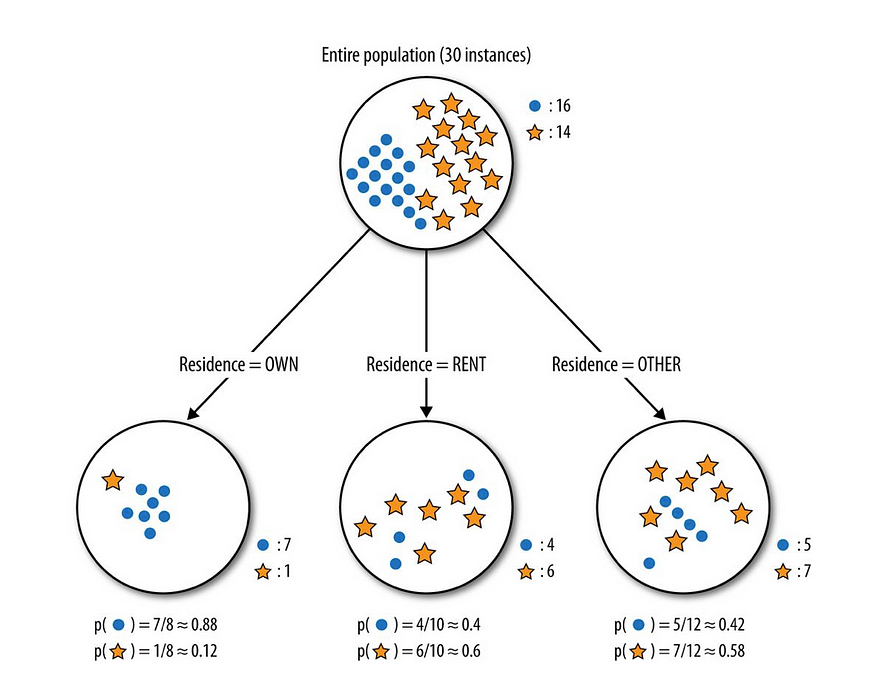
The dots are the data points with class right-off and the stars are the non-write-offs. Splitting the parent node on attribute balance gives us 2 child nodes. The left node gets 13 of the total observations with 12/13 ( 0.92 probability) observations from the write-off class and only 1/13( 0.08 probability) observations from the non-write of class. The right node gets 17 of the total observation with 13/17( 0.76 probability) observations from the non-write-off class and 4/17 ( 0.24 probability) from the write-off class.

Let’s calculate the entropy for the parent node and see how much uncertainty the tree can reduce by splitting on Balance.



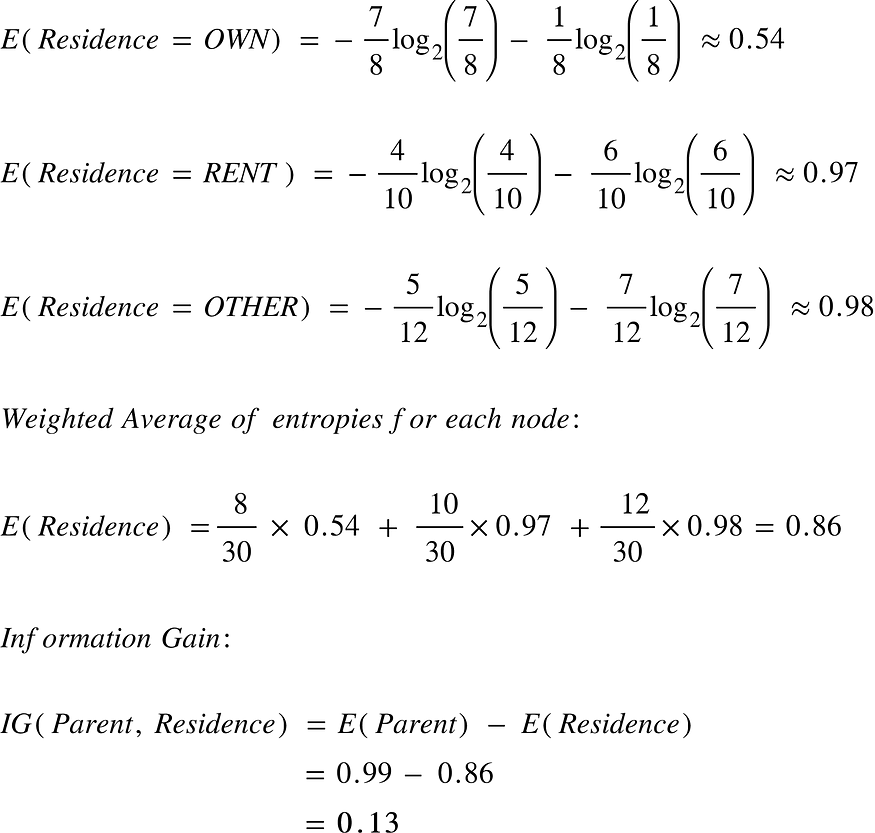
Splitting on feature ,“Balance” leads to an information gain of 0.37 on our target variable. Let’s do the same thing for feature, “Residence” to see how it compares.

**Feature 2: Residence**



Provost, Foster; Fawcett, Tom. Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking

Splitting the tree on Residence gives us 3 child nodes. The left child node gets 8 of the total observations with 7/8 (0.88 probability) observations from the write-off class and only 1/8 (0.12 probability) observations from the non-write-off class. The middle child nodes gets 10 of the total observations with 4/10 (0.4 probability) observations of the write-off class and 6/10( 0.6 probability) observations from the non-write-off class. The right child node gets 12 of the total observations with 5/12 ( 0.42 probability) observations from the write-off class and 7/12 ( 0.58 ) observations from the non-write-off class. We already know the entropy for the parent node. We simply need to calculate the entropy after the split to compute the information gain from “Residence”



The information gain from feature, Balance is almost 3 times more than the information gain from Residence! If you go back and take a look at the graphs you can see that the child nodes from splitting on Balance do seem purer than those of Residence. However the left most node for residence is also very pure but this is where the weighted averages come in play. Even though that node is very pure, it has the least amount of the total observations and a result contributes a small portion of it’s purity when we calculate the total entropy from splitting on Residence. This is important because we’re looking for overall informative power of a feature and we don’t want our results to be skewed by a rare value in a feature.