learn itANALYTICS

Entropy

Entropy is a **measure of disorder or a measure of purity.** The mathematical formula for Entropy is:

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

where

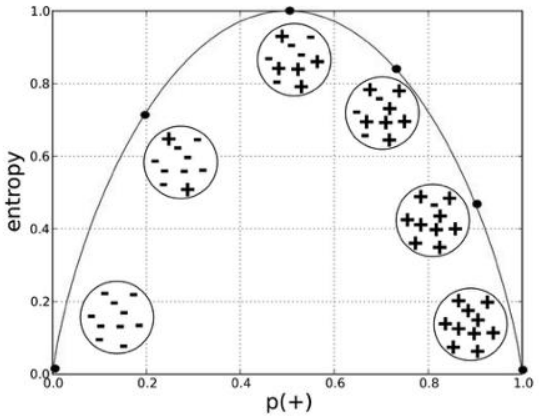
'Pi' is frequentist probability of an element/class 'i' in the data. Let's say there are only two classes, a positive class and a negative class. Therefore 'i' here could be either + or (-). So, if we had a total of 100 data points in the dataset with 30 belonging to the positive class and 70 belonging to the negative class then 'P+' would be 3/10 and 'P-' would be 7/10.

To calculate the entropy of the classes, use the following formula.

$$-\frac{3}{10} \times \log_2\left(\frac{3}{10}\right) - \frac{7}{10} \times \log_2\left(\frac{7}{10}\right) \approx 0.88$$

The entropy is approximately 0.88. This is considered a high entropy or a high level of disorder. This means that there is low level of purity. Entropy is measured between 0 and 1 for dataset with two classes. However, entropy can be greater than 1 if there are three of more classes in the dataset.

For simplicity and clarity, the example will have entropy between 0 and 1.



In the graph, the x-axis measures the proportion of data points belonging to the positive class in each bubble and the y-axis axis measures their respective entropies. Right away, you can see the inverted 'U' shape of the graph. Entropy is lowest at the extremes, when the bubble either contains no positive instances or only positive instances. That is, when the bubble is pure the disorder is 0. Entropy is highest in the middle when the bubble is evenly split between positive and negative instances. Extreme disorder, because there is no majority.

Note that entropy is a measure of disorder or uncertainty, and the goal of machine learning models is to reduce uncertainty. To measure the reduction of the disorder in the target class given the predictors or features is to use the concept of information gain.

Mathematically, information gain is written as:

$$IG(Y, X) = E(Y) - E(Y|X)$$

Information Gain from X on Y

Using the formula, simply subtract the entropy of Y given X from the entropy of just Y to calculate the reduction of uncertainty about Y given an additional piece of information X about Y. The greater the reduction in this uncertainty, the more information is gained about Y from X.

Let's have an example using the contingency table below. This example will illustrate how decision trees use entropy and information gain to decide what feature to split the nodes.

Credit Rating	Liability		
	Normal	High	Total
Excellent	3	1	4
Good	4	2	6
Poor	0	4	4
Total	7	7	14

The target variable is Liability which can take on two values “Normal” and “High” and only one predictor or feature called Credit Rating which can take on values “Excellent”, “Good” and “Poor”.

There are 14 observations. 7 of them belong to the Normal Liability class and 7 belong to High Liability Class. This is an example of an even split. There are 4 observations that have value Excellent for the feature credit rating.

The contingency table is used to calculate the entropy of our target class. Calculation of the entropy of the target class given additional information about the feature or predictor, credit rating. These allow calculation of how much additional information does “Credit Rating” provide for the target variable “Liability”.

$$E(\textit{Liability}) = -\frac{7}{14}\log_2\left(\frac{7}{14}\right) - \frac{7}{14}\log_2\left(\frac{7}{14}\right)$$

$$= -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right)$$

$$= 1$$

The entropy of the target class is 1, at maximum disorder due to the even split between class label “Normal” and “High”.

The next step is to calculate the entropy of the target variable Liability given additional information about credit score. For this, 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.

$$E(Liability \mid CR = Excellent) = -\frac{3}{4}\log_2\left(\frac{3}{4}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right) \approx 0.811$$

$$E(Liability \mid CR = Good) = -\frac{4}{6}\log_2\left(\frac{4}{6}\right) - \frac{2}{6}\log_2\left(\frac{2}{6}\right) \approx 0.918$$

$$E(Liability \mid CR = Poor) = -0\log_2(0) - \frac{4}{4}\log_2\left(\frac{4}{4}\right) = 0$$

Weighted Average:

$$E(Liability \mid CR) = \frac{4}{14} \times 0.811 + \frac{6}{14} \times 0.918 + \frac{4}{14} \times 0$$

$$= 0.625$$

Based on the result of calculation, the entropy for the target class is 0.625. The Information Gain on Liability from Credit Rating to see how informative the feature is calculated as:

Inf ormation Gain:

$$IG(Liability, CR) = E(Liability) - E(Liability \mid CR)$$

$$= 1 - 0.625$$

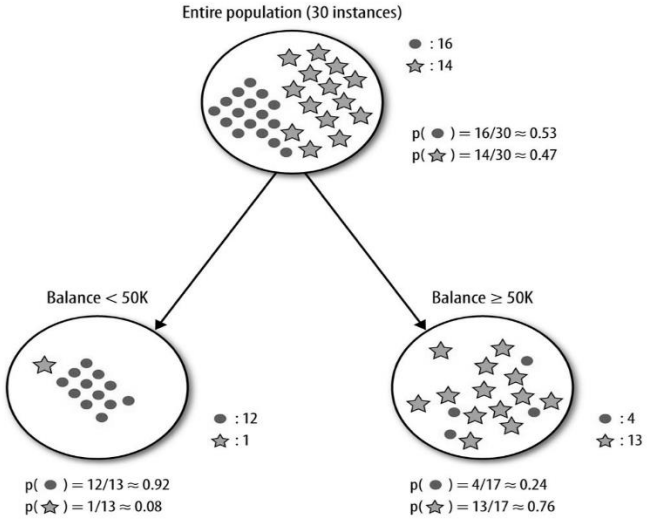
$$= 0.375$$

Knowing the Credit Rating helps reduce the uncertainty of the target class. This is what a good feature does. It provides information about the target variable. This is the reason why decision trees use entropy and information gain to determine which feature to split the nodes on to get closer to predicting the target variable with each split and also to determine when to stop splitting the tree. You may look for a good article about hyperparameter like max depth for more advanced methods.

Example: Decision Tree

Consider an example on building a decision tree to predict whether a loan given to a person would result in a write-off or not. The dataset consists of 30 instances, 16 belong to the write-off class and the other 14 belong to the non-write-off class. The features are “Balance” that can take on two values -> “< 50K” or “>50K” and “Residence” that can take on three values -> “OWN”, “RENT” or “OTHER”.

This example will show how a decision tree algorithm will decide what attribute to split on first and what feature provides more information, or reduces more uncertainty about our target class out of the two using the concepts of entropy and information gain.



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 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.

Calculate the entropy for the parent node and see how much uncertainty the tree can reduce by splitting on Balance,

$$E(Parent) = -\frac{16}{30}\log_2\left(\frac{16}{30}\right) - \frac{14}{30}\log_2\left(\frac{14}{30}\right) \approx 0.99$$

$$E(Balance < 50K) = -\frac{12}{13}\log_2\left(\frac{12}{13}\right) - \frac{1}{13}\log_2\left(\frac{1}{13}\right) \approx 0.39$$

$$E(Balance > 50K) = -\frac{4}{17}\log_2\left(\frac{4}{17}\right) - \frac{13}{17}\log_2\left(\frac{13}{17}\right) \approx 0.79$$

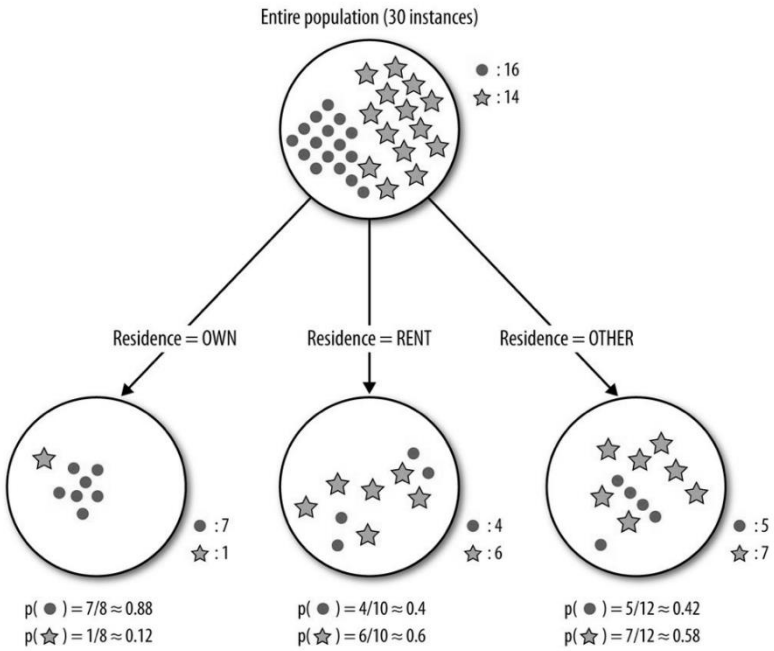
Weighted Average of entropy for each node:

$$E(Balance) = \frac{13}{30} \times 0.39 + \frac{17}{30} \times 0.79$$
$$= 0.62$$

Information Gain:

$$IG(Parent, Balance) = E(Parent) - E(Balance)$$
$$= 0.99 - 0.62$$
$$= \mathbf{0.37}$$

Splitting on feature, “Balance” leads to an information gain of 0.37 on the target class. You may do the same to feature, “Residence”.



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 get 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.

Calculate the entropy after the split to compute the information gain from “Residence”.

$$E(Residence = OWN) = -\frac{7}{8}\log_2\left(\frac{7}{8}\right) - \frac{1}{8}\log_2\left(\frac{1}{8}\right) \approx 0.54$$

$$E(Residence = RENT) = -\frac{4}{10}\log_2\left(\frac{4}{10}\right) - \frac{6}{10}\log_2\left(\frac{6}{10}\right) \approx 0.97$$

$$E(Residence = OTHER) = -\frac{5}{12}\log_2\left(\frac{5}{12}\right) - \frac{7}{12}\log_2\left(\frac{7}{12}\right) \approx 0.98$$

Weighted Average of entropies for each node:

$$E(Residence) = \frac{8}{30} \times 0.54 + \frac{10}{30} \times 0.97 + \frac{12}{30} \times 0.98 = 0.86$$

Information Gain:

$$\begin{aligned} IG(Parent, Residence) &= E(Parent) - E(Residence) \\ &= 0.99 - 0.86 \\ &= \mathbf{0.13} \end{aligned}$$

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 its purity when you calculate the total entropy from splitting on Residence. This is important if you are looking for overall informative power of a feature and you don't want the results to be skewed by a rare value in a feature.

By itself the feature, Balance provides more information about the target class than Residence. It reduces more disorder in the target class. A decision tree algorithm uses the result to make the first split on the data using Balance. From here on, the decision tree algorithm would use this process at every split to decide what feature it is going to split on next. In a real-world scenario, with more than two features, the first split is made on the most informative feature and then at every split the information gain for each additional feature needs to be recomputed because it would not be the same as the information gain from each feature by itself. The entropy and information gain would have to be calculated after one or more splits have already been made which would change the results. A decision tree would repeat this process as it grows deeper and deeper till either it reaches a pre-defined depth or no additional split can result in a higher information gain beyond a certain threshold which can also usually be specified as a hyper-parameter.

Reference

Entropy: How Decision Trees Make Decisions. Towards Data Science