what is information?

Information from observing the occurrence of an event = the number of bits needed to encode the probability of the event.

If an event has probability $p \to -\log_2(p)$ **bits** are needed.

A coin flip encodes 1 **bit** of information:

$$p = \frac{1}{2}, \text{and thus } -\log_2\left(\frac{1}{2}\right) = 1 \textbf{ bit}$$

For an event with probability 1, 0 bits are needed:

$$p = 1, \text{and thus } -\log_2(1) = 0 \textbf{ bits}$$

Given many events, with probabilities $[\boldsymbol{p_1}, \ldots, \boldsymbol{p_m}]$, the probabilities' mean information is as follows:

$$E_{p\sim[p_1,\ldots,p_m]}I(p) = \sum_{j=1}^{j} p_j I(p_j) = -\sum_{j=1}^{j} p_j \log_2 p_j = \boldsymbol{H([p_1, \ldots, p_m])} \to \textbf{Entropy}$$

Any probabilities that are ~ 0 will have a corresponding **entropy** ~ 0, synonymous to applying the definition of an average. The information is simply weighted by their probabilities of occurrence.

Given the definition above, if only 2 events (**binary**) with probabilities p and $1-p$ exist:

$$H([p, 1-p]) = -p\log_2(p) - (1-p)\log_2(1-p)$$

If the probabilities are $\frac{1}{2}$ and $\frac{1}{2}$:

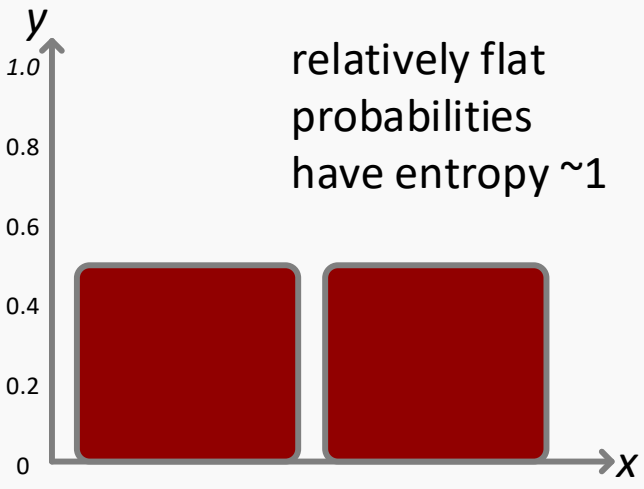
$$H\left(\left[\frac{1}{2}, \frac{1}{2}\right]\right) = -2\frac{1}{2}\log_2\left(\frac{1}{2}\right) = 1$$

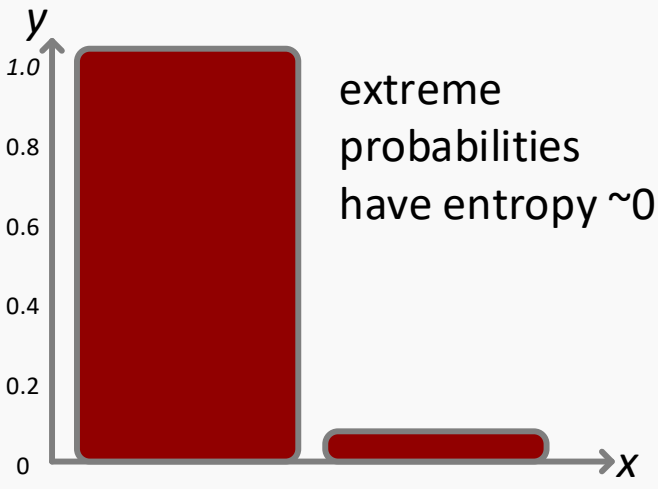
If the probabilities are 0.99 and 0.01:

$$H([0.99, 0.01]) = 0.08 \text{ bits}$$

Entropy Intuition

In relation to probability, higher **entropy** correlated with flat probabilities (uniform) that are spread out. **Rare events** conversely result in lower **entropy**:





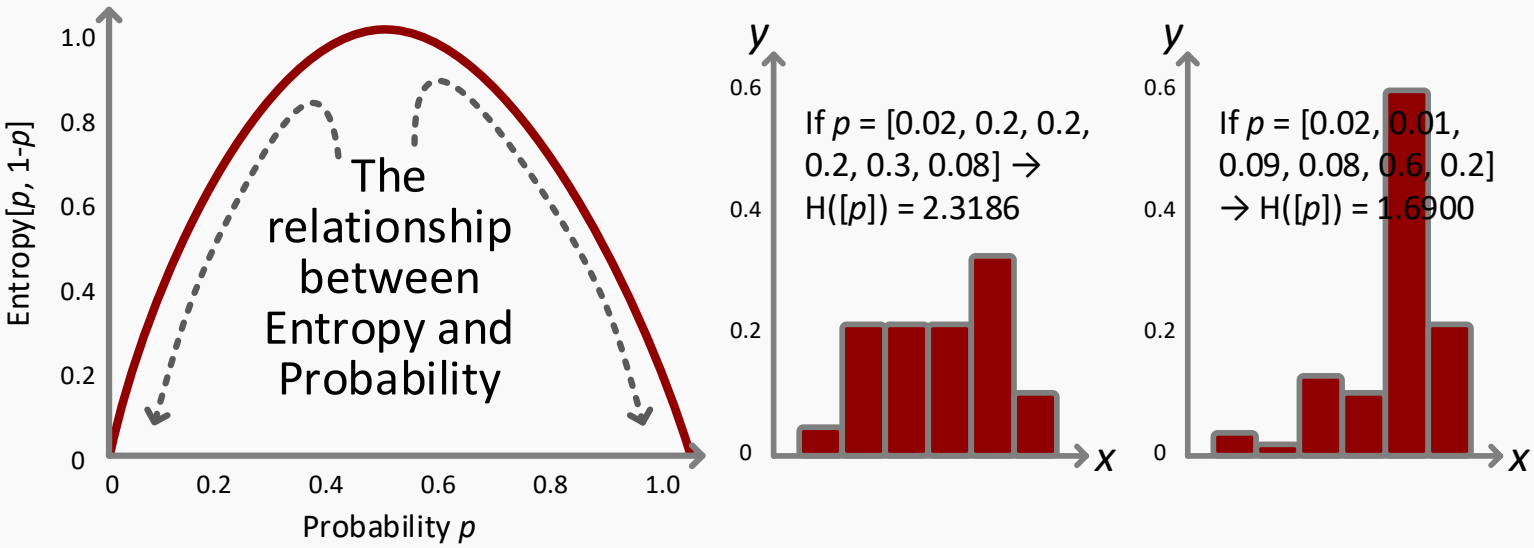
If the probabilities are $\frac{1}{2}$ and $\frac{1}{2}$:

$$H\left(\left[\frac{1}{2}, \frac{1}{2}\right]\right) = -2\frac{1}{2}\log_2\left(\frac{1}{2}\right) = 1$$

If the probabilities are 0.99 and 0.01:

$$H([0.99, 0.01]) = 0.08 \text{ bits}$$

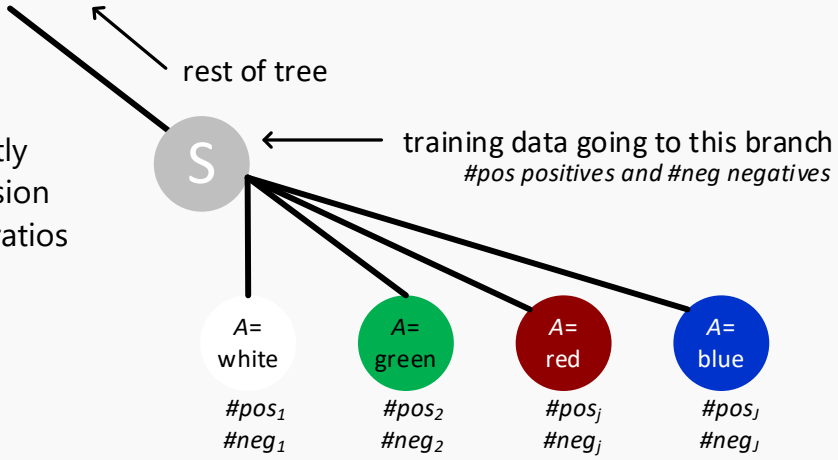
For example:



splitting criteria for decision trees ⚠ information gain

Given the above definition of **information**, with **entropy** being the average of **information**, the definition of **information gain** is as follows:

Support Vector Machines begin by addressing the question of which decision boundary to select when each point is correctly classified. The right illustration splits the decision on a **color** variable. The positive to negative ratios can be computed for each branch:



The training probabilities for branch$_j$ are:

$$\left[\frac{\#\text{pos}_j}{\#\text{pos}_j + \#\text{neg}_j}, \frac{\#\text{neg}_j}{\#\text{pos}_j + \#\text{neg}_j}\right]$$

... noting that $\frac{[0.08, 0.99] \rightarrow \text{is good}}{[0.50, 0.50] \rightarrow \text{is bad}}$

Extreme probabilities at either end of the spectrum (near **0** and **1)** have strong predictive power; the predicted labels will essentially be known for new observations in the dataset. Flat probabilities are poor predictors.

Thus, **Information Gain** for a particular subset of Data $\boldsymbol{S}$, in a particular branch of variable $\boldsymbol{A}$ is the original **entropy** before the branch minus the entropy after the branch:

$$\begin{aligned}\text{Gain}(\boldsymbol{S},\boldsymbol{A}) &= \text{expected reduction in entropy due to branching on attribute } \boldsymbol{A}\\ &= \text{original entropy} - \text{entropy after branching}\\ &= H\left(\left[\frac{\#\text{pos}}{\#\text{pos}+\#\text{neg}},\frac{\#\text{neg}}{\#\text{pos}+\#\text{neg}}\right]\right) - \sum_{j=1}^{j}\frac{\#\text{pos}_j+\#\text{neg}_j}{\#\text{pos}+\#\text{neg}}H\left[\frac{\#pos_j}{\#pos_j+\#neg_j},\frac{\#neg_j}{\#pos_j+\#neg_j}\right]\end{aligned}$$

For example, the variable $\boldsymbol{A}$ is split with half of the data being positive and half of the data being negative (**Entropy** – 1). The entropy after splitting will be much lower, proving information gain to be substantial:

$$\text{Gain}(\boldsymbol{S},\boldsymbol{A}) = H\left(\left[\frac{1}{2},\frac{1}{2}\right]\right) - \left[\frac{2}{12}H([0,1]) + \frac{4}{12}H([1,0]) + \frac{6}{12}H\left(\left[\frac{2}{6},\frac{4}{6}\right]\right)\right] \approx 0.541 \textbf{ bits}$$

A less successful example can be seen with variable **B**:

$$\text{Gain}(\boldsymbol{S},\boldsymbol{B}) = 1 - \left[\frac{2}{12}H\left(\left[\frac{1}{2},\frac{1}{2}\right]\right) + \frac{2}{12}H\left(\left[\frac{1}{2},\frac{1}{2}\right]\right) + \frac{4}{12}H\left(\left[\frac{2}{4},\frac{2}{4}\right]\right) + \frac{4}{12}H\left(\left[\frac{2}{4},\frac{2}{4}\right]\right)\right] \approx 0 \textbf{ bits}$$

Standard Procedure for Building Decision Trees

- Begin at the top of the Tree
- Grow the Tree by "**splitting**" the features (**C4.5** Algorithm displayed above, **CART** alternatively uses Gini Index) one by one. Determine where to split by examining how "impure" the node is.
- Assign leaf nodes the majority vote in the leaf
- Return back through the Tree to **prune** leaves and reduce overfitting of the model to the data.
  - The elimination of subtrees that do not have a high cost-to-value for the model
  - CART uses the "minimal cost complexity" pruning method

Decision Trees are generally used for **classification** but do exist in ways for **regression**.

Decision Tree Advantages

- Handles nonlinearities due to being logical models
- Decision Trees are generally interpretable (especially CART)
- Greedy models starting from the top and not regressing backwards to reevaluate, thus fast
- Can easily handle imbalanced data by reweighting the points

Decision Trees Disadvantages

- Not very interpretable without trading off in accuracy (complexity increases model accuracy)
- Greedy models will be fast but will trade off in accuracy
- Decision Trees are a heuristic algorithm, thus lacking computational elegance considering that they are not optimized for any specific purpose or application
- Tend to perform poorly on imbalanced data, even after adjusting the parameters.