Binary classification problems - 2 outputs: true/false

Entropy (disorder, impurity, uncertainty, unpredictability, bits of information)

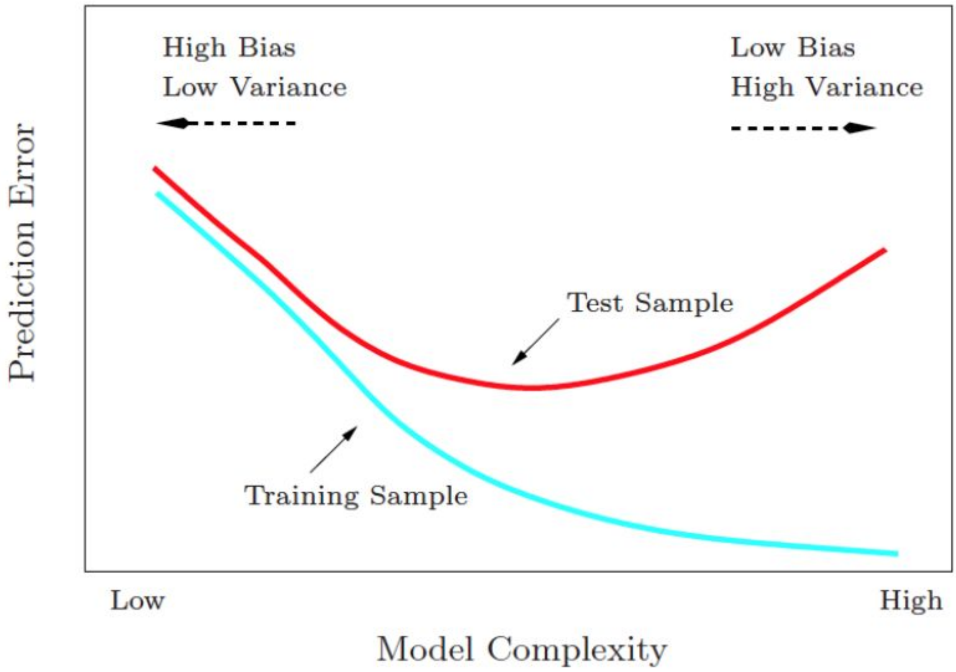
$$Ent = \sum_i - p_i log_2(p_i)$$

If pure/order, only one class, Ent = 0,
if complete uniform impurity, 50/50, Ent = 1 (in the case of binary classification, since 1 bit conveys 2 possibilities)
Fair Coin: 0.5 heads, 0.5 tails -> Ent = -0.5log(0.5)-0.5log(0.5) = -1log(0.5) = -1 (-1) = 1 bit
Skewed coin: 0.6 heads, 0.4 tails -> Ent = -0.6log(0.6)-0.4log(0.4) < 1

Information Gain
Which attribute maximize reduction of Entropy?

$$\text{Gain} = \underbrace{\text{Ent}(S)}_{\text{before}} - \underbrace{\sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|}}_{\substack{\text{weighted}\\ \text{sum}}} \underbrace{\text{Ent}(S_v)}_{\text{after}}$$

Bias and **variance**
Error due to Bias: the difference between the average (expected) prediction of our model and the correct value.
Error due to Variance: the variability of a model prediction for a given data point between different realizations of the model.



Overfitting

Good result on training data, but generalise poorly

Occurs with:

- Non-representative examples
- Noisy examples
- Too complex model

What can be done about it?

- Choose simpler model and accept some errors

Occam's principle:

- The simplest explanation compatible with data tend to be the right one

Split into training and validation sets

- Training sets tend to have errors, but the validation set is unlikely to have the same errors
- The validation set provide a safety check towards overfitting for the training set
- The validation set can't be too small

Reduced error pruning

One of the simplest forms of pruning is reduced error pruning. Starting at the leaves, each node is replaced with its most popular class. If the prediction accuracy is not affected then the change is kept. While somewhat naive, reduced error pruning has the advantage of **simplicity and speed**.

- Split data into training and validation set
- While further pruning not harmful
    - Evaluate impact on validation set of pruning each node and those below
    - Remove the one that improves validation set the most

- Consider each node for pruning
- Pruning = removing the subtree at that node, make it a leaf and assign the most common class at that node
- A node is removed if the resulting tree performs no worse then the original on the validation set - removes coincidences and errors
- Nodes are removed iteratively choosing the node whose removal most increases the decision tree accuracy on the graph
- Pruning continues until further pruning is harmful
- uses training, validation & test sets - effective approach if a large amount of data is available