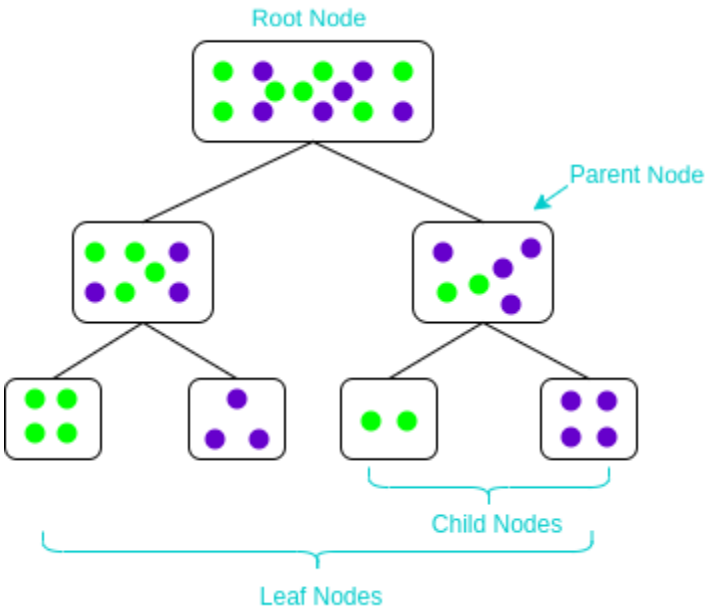
Credits : MachineLearningMastery Blog, AnalyticsVidhya, towardsDataScience, Youtube Channels (Krish Naik/StatQuest) , Udacity, udemy , Quandtree

Decision trees are simple to implement and equally easy to interpret.

- How do you split a decision tree?
- What are the different splitting criteria?
- What is the difference between Gini and Information Gain?

Basic Decision Tree Terminologies

Let's quickly revise the key terminologies related to decision trees which I'll be using throughout the article.



What is Node Splitting in a Decision Tree & Why is it Done?

A decision tree makes decisions by splitting nodes into sub-nodes.

This process is performed multiple times during the training process **until only homogenous nodes are left.**

Node splitting, or simply splitting, is the process of dividing a node into multiple sub-nodes to create relatively pure nodes.

Continuous Target Variable
- ***Reduction in Variance***

Categorical Target Variable
- ***Gini Impurity***
- ***Information Gain***
- ***Chi-Square***

In the upcoming sections, we'll look at each splitting method in detail. Let's start with the first method of splitting – reduction in variance.

Decision Tree Splitting Method #1: Reduction in Variance

Reduction in Variance is a method for splitting the node used when the target variable is continuous, i.e., regression problems. It is so-called because it uses variance as a measure for deciding the feature on which node is split into child nodes.

$$Variance = \frac{\sum (X - \mu)^2}{N}$$

Variance is used for calculating the homogeneity of a node. If a node is entirely homogeneous, then the variance is zero.

Here are the steps to split a decision tree using reduction in variance:

1. For each split, individually calculate the variance of each child node
2. Calculate the variance of each split as the weighted average variance of child nodes
3. Select the split with the lowest variance
4. Perform steps 1-3 until completely homogeneous nodes are achieved

Decision Tree Splitting Method #2: Information Gain

For Categorical target variable

Information Gain is used for splitting the nodes when the target variable is categorical.

$$Information\,Gain = 1 - Entropy$$

$$Entropy = -\sum_{i=1}^{n} p_i \log_2 p_i$$

$$Entropy(S) = -(P_{\oplus} log_2 P_{\oplus} + P_{\ominus} log_2 P_{\ominus}) \quad (1.1)$$

Where $P_{\oplus}$ is the portion of positive examples and $P_{\ominus}$ is the portion of negative examples in S.

Entropy is used for calculating the purity of a node.

Lower the value of entropy, higher is the purity of the node. -> Information gain follows the reverse trend

The entropy of a homogeneous node is zero. -> Information gain is 1

Steps to split a decision tree using Information Gain:

1. For each split, individually calculate the entropy of each child node
2. Calculate the entropy of each split as the weighted average entropy of child nodes
3. Select the split with the lowest entropy or highest information gain
4. Until you achieve homogeneous nodes, repeat steps 1-3

Decision Tree Splitting Method #3: Gini Impurity

Gini Impurity is a method for splitting the nodes when the target variable is categorical. It is the most popular and the easiest way to split a decision tree. The Gini Impurity value is:

$$Gini\ Impurity = 1 - Gini$$

Wait – what is Gini?

Gini is the probability of correctly labeling a randomly chosen element if it was randomly labeled according to the distribution of labels in the node. The formula for Gini is:

$$Gini = \sum_{i=1}^{n} p_i^2$$

And Gini Impurity is:

$$Gini\ Impurity = 1 - \sum_{i=1}^{n} p_i^2$$

Lower the Gini Impurity, higher is the homogeneity of the node. The Gini Impurity of a pure node is zero.

Now, you might be thinking we already know about Information Gain then, why do we need Gini Impurity?

Gini Impurity is preferred to Information Gain because it does not contain logarithms which are computationally intensive.

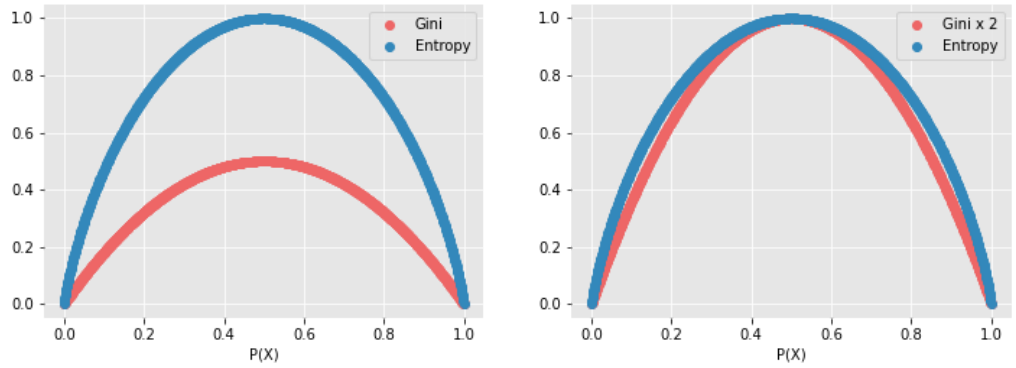
Here are the steps to split a decision tree using Gini Impurity:

1. Similar to what we did in information gain. For each split, individually calculate the Gini Impurity of each child node
2. Calculate the Gini Impurity of each split as the weighted average Gini Impurity of child nodes
3. Select the split with the lowest value of Gini Impurity
4. Until you achieve homogeneous nodes, repeat steps 1-3

Gini vs Entropy

The Gini Index and the Entropy have two main differences:

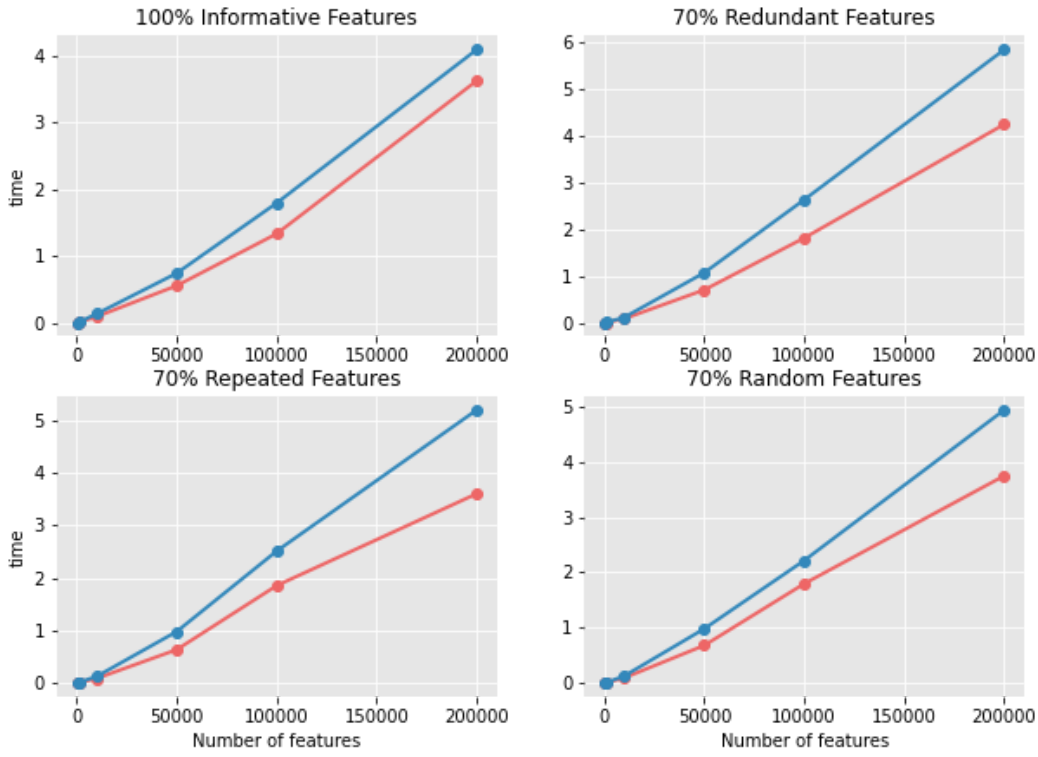
- Gini Index has values inside the interval [0, 0.5] whereas the interval of the Entropy is [0, 1]. In the following figure, both of them are represented. The gini index has also been represented multiplied by two to see concretely the differences between them, which are not very significant.



Gini Index and Entropy

- Computationally, entropy is more complex since it makes use of **logarithms** and consequently, the calculation of the Gini Index will be faster.

In the following graphs, the x-axis is the number of samples of the dataset and the y-axis is the training time.



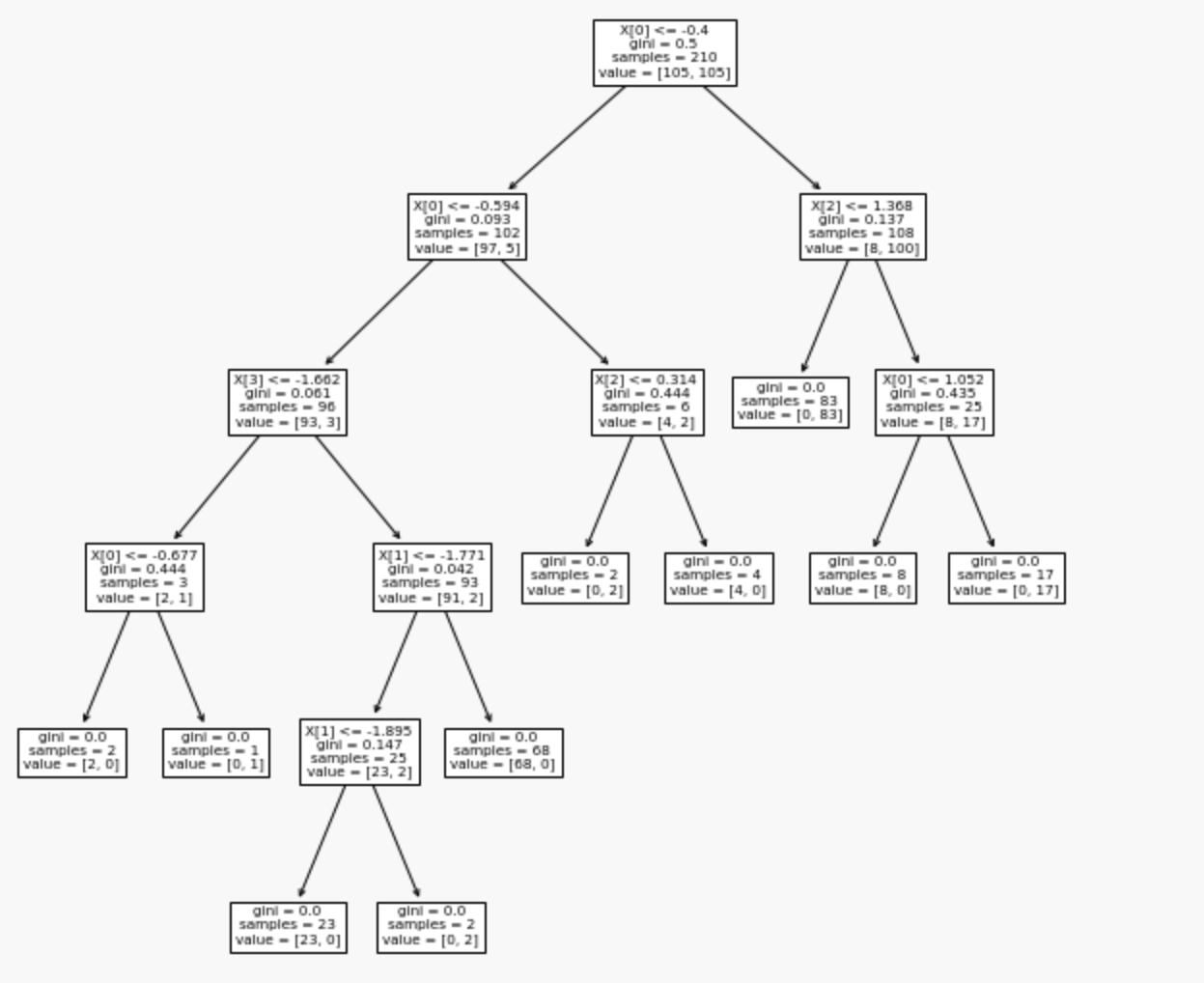
Training time when using the Entropy criterion is much higher.
Redundant features are the most affected ones, making the training time 50% longer than when using informative features.
The differences in training time are more noticeable in larger datasets.

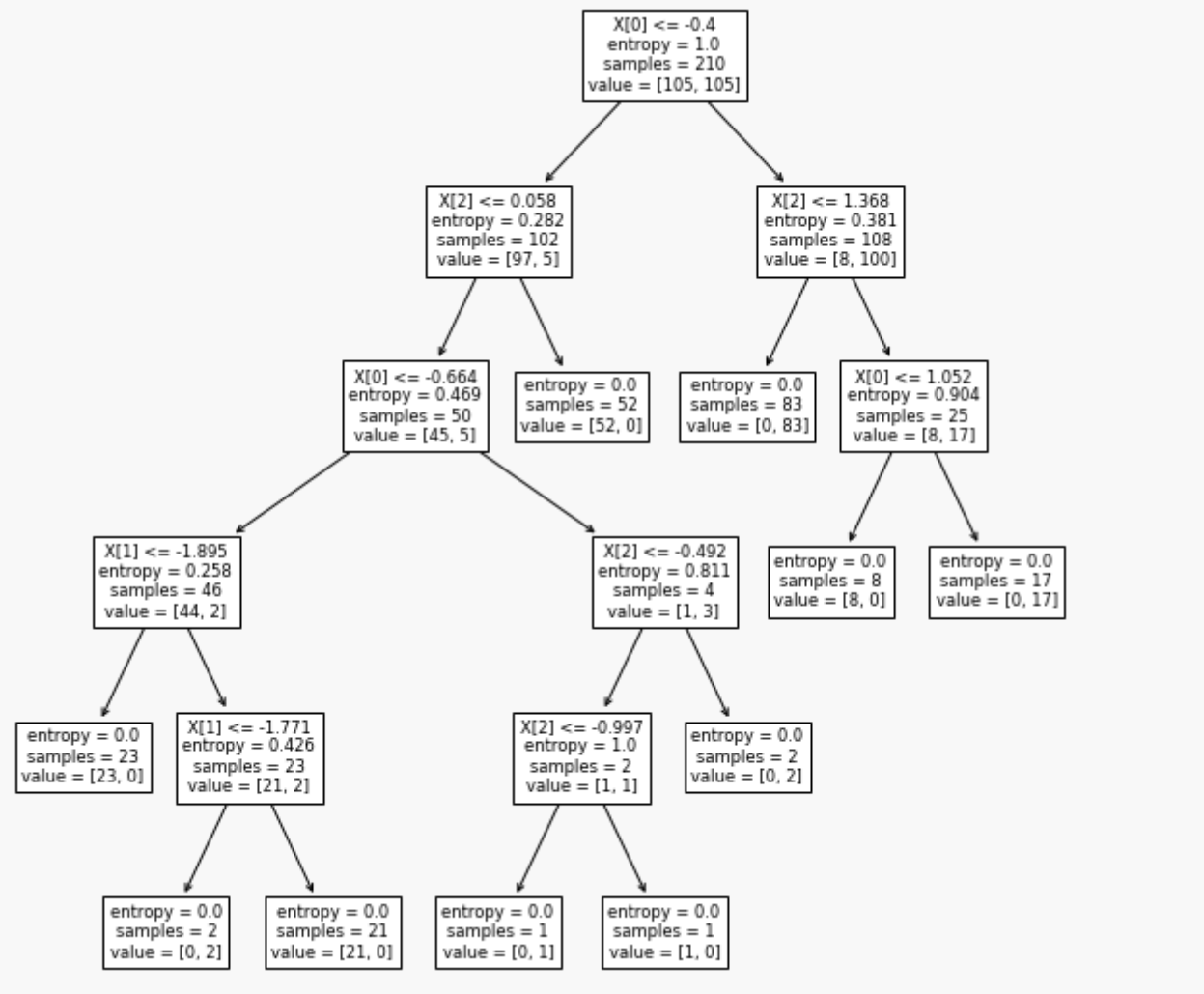
Results

Besides, we are also going to compare the obtained results with both criteria. For that purpose, we are going to use the datasets used in the training time analysis, specifically the ones with 1000 samples. Furthermore, **cross-validation** has also been used with k=3. The following table shows the obtained results, these being the **F-Score**.

	Dataset 1	Dataset 2	Dataset 3	Dataset 4
Gini	0.8619 ± 0.0088	0.8148 ± 0.0030	0.9350 ± 0.0146	0.8481± 0.0029
Entropy	0.8659 ± 0.0205	0.8119 ± 0.0223	0.9390 ± 0.0146	0.8583 ± 0.0067

the results are very similar, being the ones where the entropy criterion is used slightly better.
Finally, if we compare the structure of the trees, we can see that they are different.
Although the trees are not equal, the obtained result is practically identical.


X[0] <= -0.4
gini = 0.5
samples = 210
value = [105, 105]
X[0] <= -0.594
gini = 0.093
samples = 102
value = [97, 5]
X[2] <= 1.368
gini = 0.137
samples = 108
value = [8, 100]
X[3] <= -1.662
gini = 0.061
samples = 96
value = [93, 3]
X[2] <= 0.314
gini = 0.444
samples = 6
value = [4, 2]
gini = 0.0
samples = 83
value = [0, 83]
X[0] <= 1.052
gini = 0.435
samples = 25
value = [8, 17]
X[0] <= -0.677
gini = 0.444
samples = 3
value = [2, 1]
X[1] <= -1.771
gini = 0.042
samples = 93
value = [91, 2]
gini = 0.0
samples = 2
value = [0, 2]
gini = 0.0
samples = 4
value = [4, 0]
gini = 0.0
samples = 8
value = [8, 0]
gini = 0.0
samples = 17
value = [0, 17]
gini = 0.0
samples = 2
value = [2, 0]
gini = 0.0
samples = 1
value = [0, 1]
X[1] <= -1.895
gini = 0.147
samples = 25
value = [23, 2]
gini = 0.0
samples = 68
value = [68, 0]
gini = 0.0
samples = 23
value = [23, 0]
gini = 0.0
samples = 2
value = [0, 2]


X[0] <= -0.4
entropy = 1.0
samples = 210
value = [105, 105]
X[2] <= 0.058
entropy = 0.282
samples = 102
value = [97, 5]
X[2] <= 1.368
entropy = 0.381
samples = 108
value = [8, 100]
X[0] <= -0.664
entropy = 0.469
samples = 50
value = [45, 5]
entropy = 0.0
samples = 52
value = [52, 0]
entropy = 0.0
samples = 83
value = [0, 83]
X[0] <= 1.052
entropy = 0.904
samples = 25
value = [8, 17]
X[1] <= -1.895
entropy = 0.258
samples = 46
value = [44, 2]
X[2] <= -0.492
entropy = 0.811
samples = 4
value = [1, 3]
entropy = 0.0
samples = 8
value = [8, 0]
entropy = 0.0
samples = 17
value = [0, 17]
entropy = 0.0
samples = 23
value = [23, 0]
X[1] <= -1.771
entropy = 0.426
samples = 23
value = [21, 2]
X[2] <= -0.997
entropy = 1.0
samples = 2
value = [1, 1]
entropy = 0.0
samples = 2
value = [0, 2]
entropy = 0.0
samples = 2
value = [0, 2]
entropy = 0.0
samples = 21
value = [21, 0]
entropy = 0.0
samples = 1
value = [0, 1]
entropy = 0.0
samples = 1
value = [1, 0]