Decision Tree

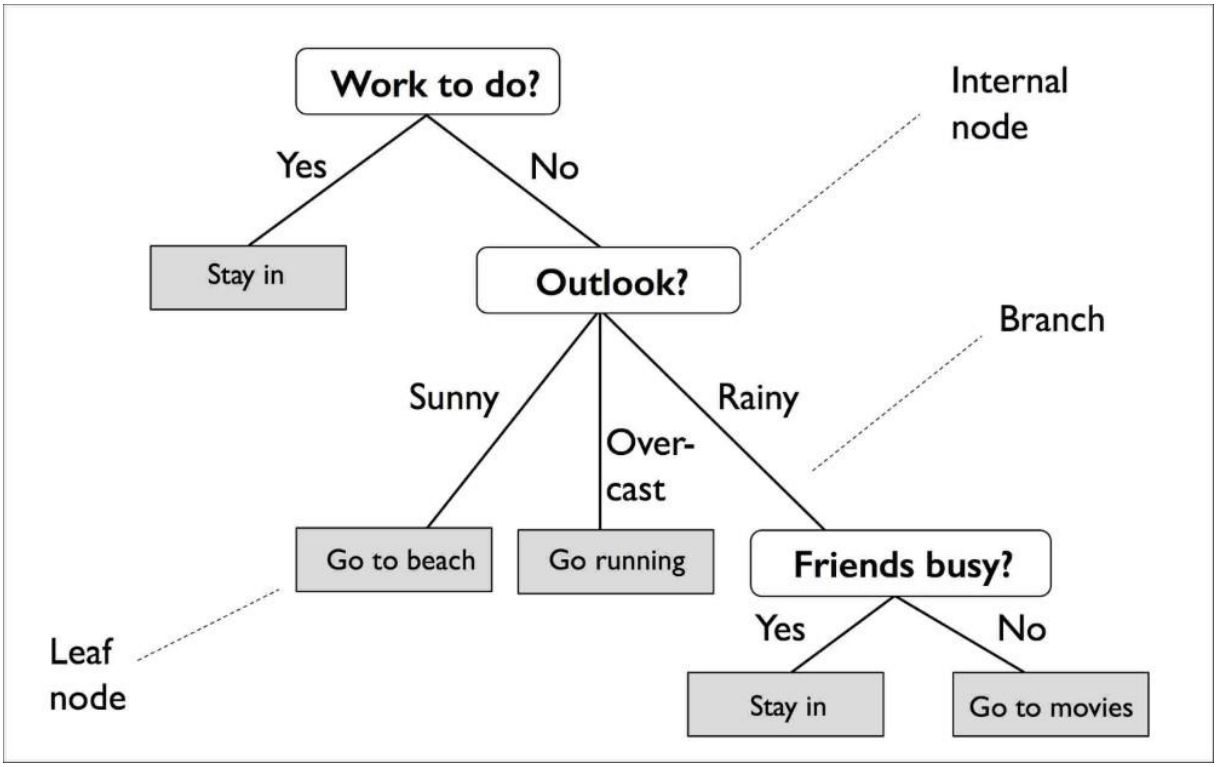
Concept

Decision trees are easy to understand, and they are the basic building block for some of the best models in data science.

The simplest possible decision tree is a tree with 2 branches (Where an answer to a yes-no question could lead to 2 different scenarios “if-then-else decision”). This step of capturing patterns from data is called fitting or training the model. The data used to fit the model is called the training data.

To get a better precision we can add more Decision-Branches at lower level, and the point at the bottom where we make a prediction is called a leaf.

As seen in the example below:



Advantages:

1. simple to understand and interpret.
2. Help determine worst, best, and expected values for different scenarios.
3. Can be combined with other decision techniques.

Disadvantages:

1. relatively inaccurate.
2. Calculations can get very complex, particularly if many values are uncertain and/or if many outcomes are linked.

How does the algorithm work?

The basic idea behind any decision tree algorithm is as follows:

1. Select the best attribute using Attribute Selection Measures (ASM) to split the records.
2. Make that attribute a decision node and breaks the dataset into smaller subsets.
3. Starts tree building by repeating this process recursively for each child until one of the conditions will match:
    - All the tuples belong to the same attribute value.
    - There are no more remaining attributes.
    - There are no more instances.

Attribute Selection Measures

In Decision Tree it is basically the splitting rules since it determines the breakpoints for tuples.

The 2 ASMs that are used be decision tree are:

Information Gain

Its basically used for training dataset by evaluating the information gain for each variable, and selecting the one that maximizes the information gain, which minimizes the entropy and best splits the dataset into groups for effective classification. (so, it done for feature selection, by evaluating the gain of each variable in the context of the target variable).

A larger information gain suggests a lower entropy group or groups of samples, and hence less surprise.

To calculate it (using python):

```
1. # calculate the entropy for a dataset
2. from math import log2
3. # proportion of examples in each class
4. class0 = 10/100
5. class1 = 90/100
6. # calculate entropy
7. entropy = -(class0 * log2(class0) + class1 * log2(class1))
8. # print the result
9. print('entropy: %.3f bits' % entropy)
```

```
1. entropy: 0.469 bits
```

Running the example, we can see that entropy of the dataset for binary classification is less than 1 bit. That is, less than one bit of information is required to encode the class label for an arbitrary example from the dataset.

p.s: An entropy of 0 bits indicates a dataset containing one class; an entropy of 1 or more bits suggests maximum entropy for a balanced dataset (depending on the number of classes), with values in between indicating levels between these extremes.

Information gain provides a way to use entropy to calculate how a change to the dataset impacts the purity of the dataset, e.g., the distribution of classes. A smaller entropy suggests more purity or less surprise.

Gini index

Gini index or Gini impurity measures the degree or probability of a particular variable being wrongly classified when it is randomly chosen. But what is meant by 'impurity'? If all the elements belong to a single class, then it can be called pure. The degree of Gini index varies between 0 and 1, where 0 denotes that all elements belong to a certain class or if there exists only one class, and 1 denotes that the elements are randomly distributed across various classes. A Gini Index of 0.5 denotes equally distributed elements into some classes.

Formula for Gini Index:

$$\text{Gini} = 1 - \sum_{i=1}^{n} (p_i)^2$$

where pi is the probability of an object being classified to a particular class.

While building the decision tree, we would prefer choosing the attribute/feature with the least Gini index as the root node.

DISCLAIMER

I did not come up with any of the information I just collected them from many online sources and videos that are too much to list but mostly used source was
"https://www.datacamp.com/community/tutorials/decision-tree-classification-python"