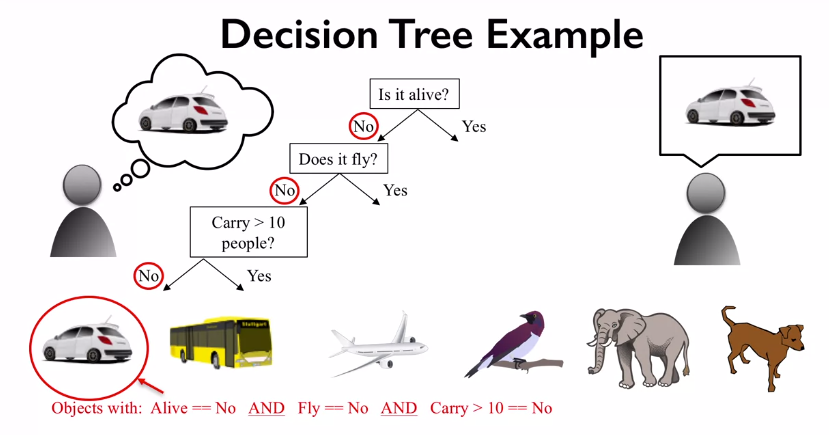
**Decision Trees:**

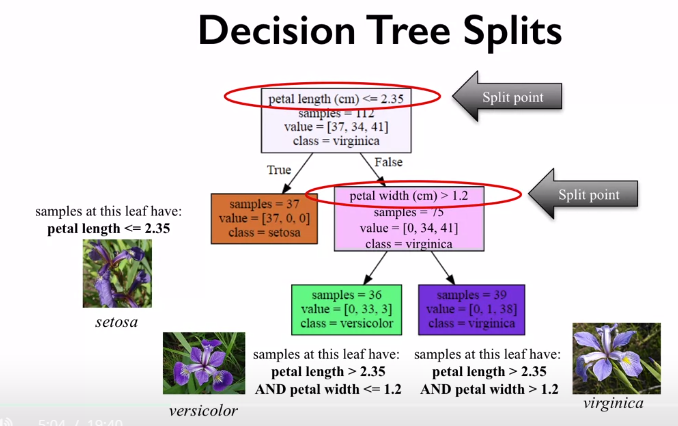
A supervised ML method that can be used for both **regressions** and **classification**. Decision trees are often used as an **exploratory** way to understand what features influence your dataset.



The topmost decision is known as the **Root node**, the arrows are known as the **branches**, the second to the second last decisions are known as **nodes**, and the final outcome is known as the **leaf node.**

Rather than try to figure out these nodes manually there are supervised algorithms that can determine the order of the nodes to most efficiently determine the category of the data. Decision trees can work with both True or False decision as well as numerical decision e.g. “is its > 10”.

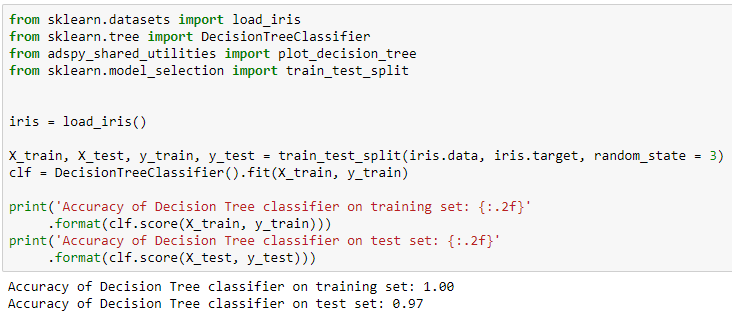
**The decision tree is built by trying to find questions that best classify the data using the fewest steps.**



**Splits** are known as the decision used to form a decision, an **informative split** is when a split results in an accurate separation of the labelled data, e.g. above the split of “petal length <= 2.35” => True, results in all of the Setosa flowers being correctly classified.

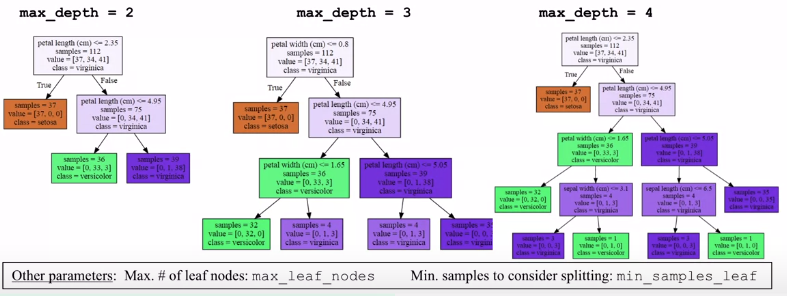
A way to ensure that **splits** are always **informative** is to use **information game**.

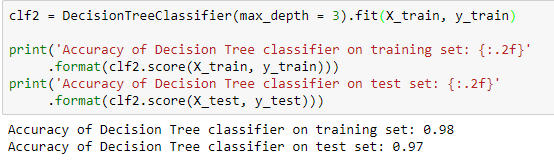
Trees whose leaf nodes have all the same target value are called **Pure leaf nodes**. For the above figure that would be the Setosa node.



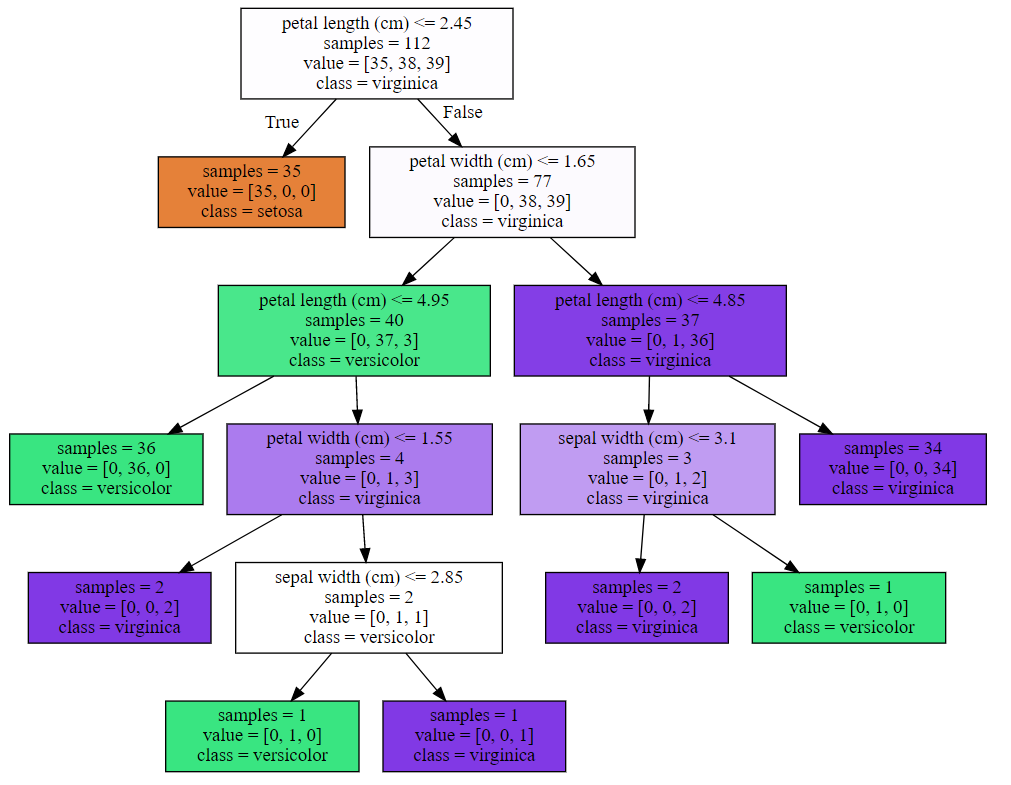
We can see that the training data shows a perfect score, this is because all of the leaf nodes are **pure** as the decision tree has kept adding leaf nodes until they become pure. This is an example of **over training**; this is a problem with decision trees. The above decision tree is overly complex and essentially memorises the training data and might not generalize well with new data.

One strategy to prevent overfitting it to use the argument “max­­\_dept” in the training stage. This is known as **pruning.** Another method is to allow the decision tree to go to a depth that has all pure leaf nodes and then reduce the depth once we know the maximum depth, this is known as **post-pruning**.



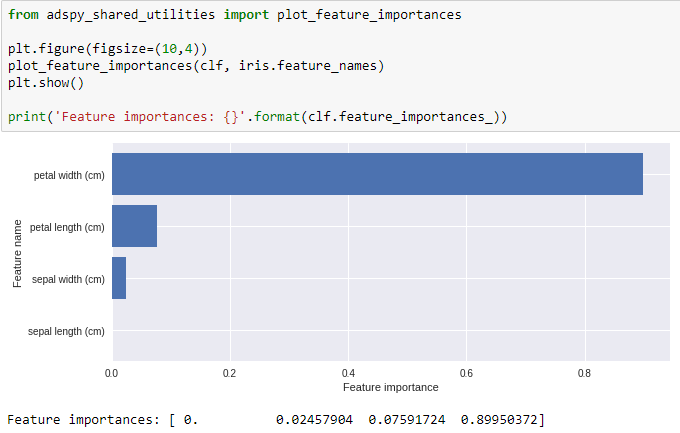


The above code shows that when the decision tree is set to a max depth of 3 it has a slightly reduced training score but the same test score. Below is an example of an **unpruned** decision tree of the iris data set used above.



**Feature Importance Calculations:**

What this does is indicate how important that feature is to the overall prediction accuracy, this is typically a number between 0-1. A value of 0 means that the feature has no relevance to the prediction, whereas a value of 1 would perfectly predict the target value. These numbers are often normalized so that the sum of all the feature importance’s equals 1.



If the feature shows a low importance that doesn’t necessarily mean that the feature is not important for prediction, it also doesn’t indicate more complex relationships between features. Having high importance can show quick insights into individual relationship when trying to predict outcomes.

It important to know that the feature importance is heavily impacted by the test train split and therefore to get a really good insight into the individual features many feature importance plots need to be done (possibly use CV for this).

