① Lecture: Tree Based Learning.

Plan:-

Introduction to decision trees

- Why are the useful?
- Classification and Regression.

The big question:

How do we learn a decision tree from data?

One problem: There are too many trees that explain the same data. So how do we pick the best one?

Short interlude on information theory.

ID3 (Iterative Dichotomiser 3) for learning Classification trees.

Algorithms for learning regression trees

Random forests, boosting and other methods for improving tree learning.

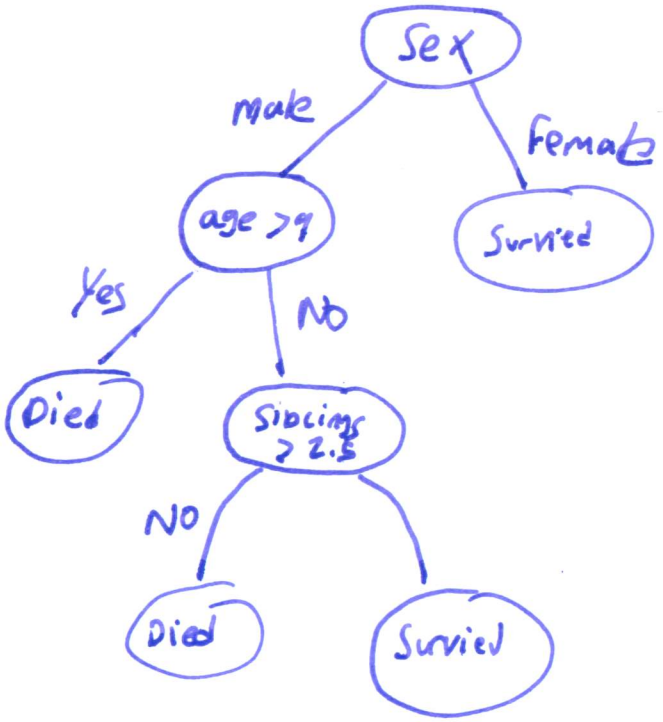
Remember: Regression ? predict a Value

Classification predict a Class.

Your data consists of features or attributes after training you want to implement some function

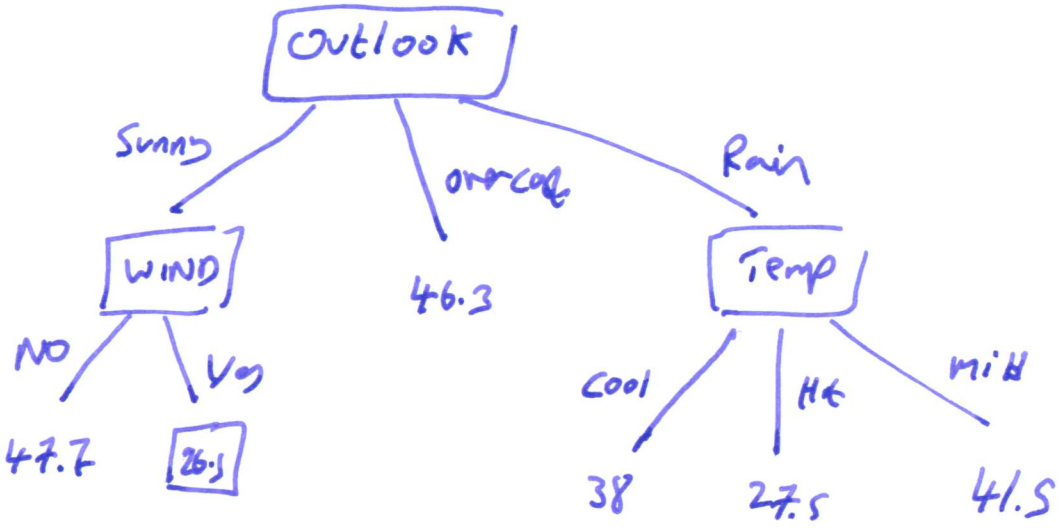
$$f: \text{Features} \longrightarrow \begin{cases} \text{Class} & - \text{ Classification} \\ \text{Value.} & - \text{ Regression} \end{cases}$$

Decision trees The formal definition is not so helpful. It is a rooted directed acyclic graph. The leaf nodes represent classes or values. Given feature values you follow a path from the root to a leaf.



Decision trees for ~~classification~~ Regression

Leaf nodes represent values.



Trying to predict how many hours a week that the US president plays golf.

Advantages of decision trees.

- We can represent non-linear functions

- Small trees are easy to understand and hence give explainable machine learning models.

Explainability :-

Why did the machine learning model refuse me a bank loan?

For the moment we will stick to classification.

Given some data how do we learn a decision tree?

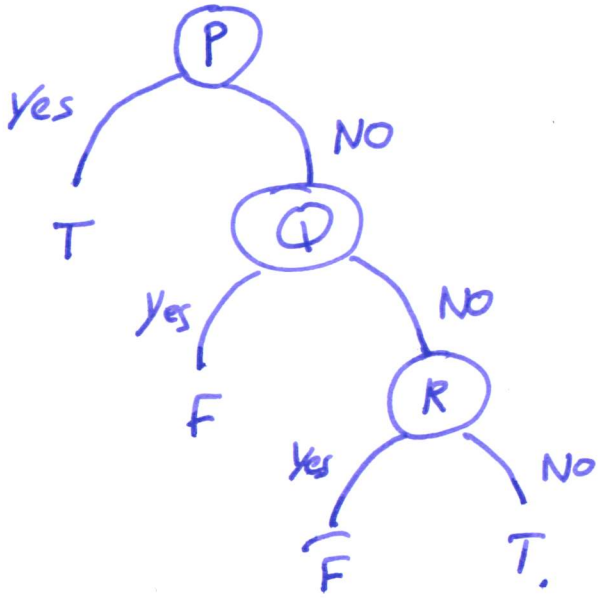
Issues:

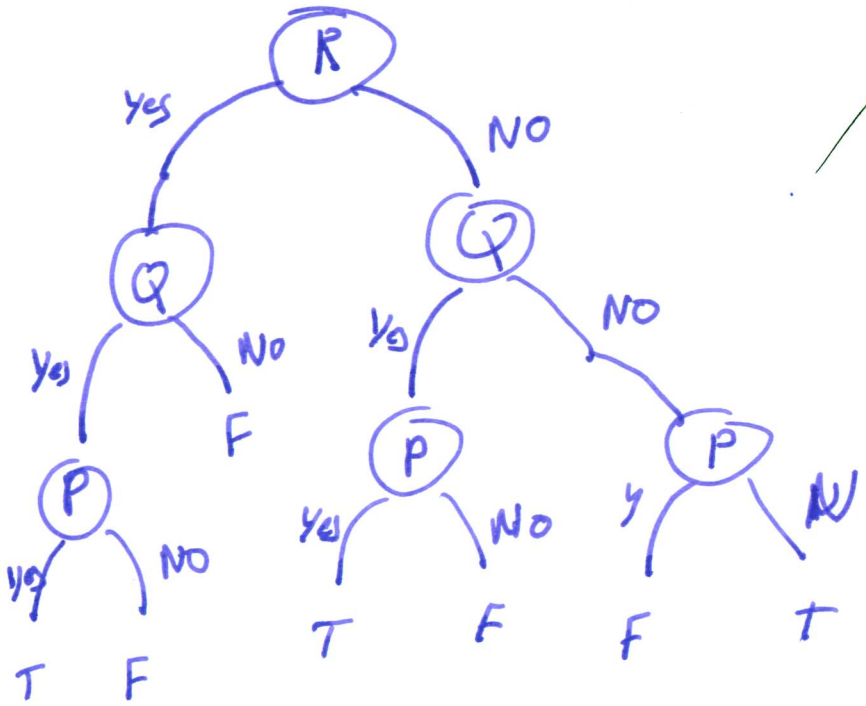
How do we avoid overfitting?

How do we learn small ~~trees~~ trees?

We want the ordering of the nodes to some how represent the importance of the features.

Even for the boolean case the order you consider the features or variables gives you different size/shape trees.





These trees (should unless I made a mistake) represent the same function.

Facts (Complexity theory)

- For boolean functions you can get an exponential blow up in the size for different orders.

- It it is NP-hard to find an ordering that gives the smallest tree.

Implications for machine learning

- We have to find heuristics that find small trees.

  Small trees give good easy to understand explanations.

  Small trees also avoid overfitting.

We will look at one approach using information theory.

Information theory

Revision $\log_2 x = y \iff 2^y = x$

$\log_2 4 = 2 \quad (2^2 = 4) \qquad \log_2 1 = 0 \quad 2^0 = 1.$

In computer science we often measure capacity on a logarithmic scale

2G is twice the size of 1G

Even though the number ~~bits~~ ~~grow~~ states (possible 0-1 patterns grows exponentially)

2^{1024} 2^{2048} not twice the size.

Information theory is a measure of information that takes into account the probability of events.

Don't forget

$$\log_b(x) = \frac{\log(x)}{\log(b)}$$

If I flip an unfair coin with probability 0.9 H and 0.1 T then knowing the coin came up heads is not so much information.

If the probabilities are 0.5 H & 0.5 T then knowing that it is heads is much more information.

H is a function that takes probability distributions and returns the amount of information.

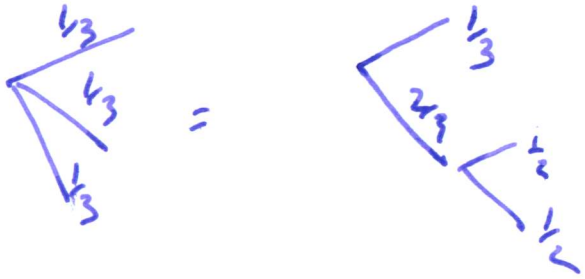
We want certain property for example

~~H(P1,P2)~~

$$H\left(\tfrac{1}{2}, \tfrac{1}{2}\right) \leq H\left(\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}\right) \leq H\left(\tfrac{1}{4}, \tfrac{1}{4}, \tfrac{1}{4}, \tfrac{1}{4}\right)$$

&

$$H\left(\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}\right) = H\left(\tfrac{1}{3}, \tfrac{2}{3}\right) + \tfrac{2}{3} H\left(\tfrac{1}{2}, \tfrac{1}{2}\right) \cdot \tfrac{2}{3}$$



Mathematically the only sensible measure is

$$H(P_1, \ldots, P_n) = -\sum_{i=1}^{n} P_i \cdot \log_2 P_i$$

It does not matter which base, but Convention we take log to the base 2.

If we have a fair coin

$$H(\tfrac{1}{2}, \tfrac{1}{2}) = -\tfrac{1}{2}\log(1/2) - \tfrac{1}{2}\log(\tfrac{1}{2})$$

$$= -\left(\log(\tfrac{1}{2})\right) = -(\log 1 - \log 2)$$

$$= -(0 - 1) = 1.$$

We can now calculate the entropy of sets of observations

Suppose we have 47 people (Y) 23 own cars, 24 don't own cars

$$H(Y) = -\frac{23}{47}\log_2\frac{23}{47} - \frac{24}{47}\log_2\frac{24}{47} = 0.9997$$

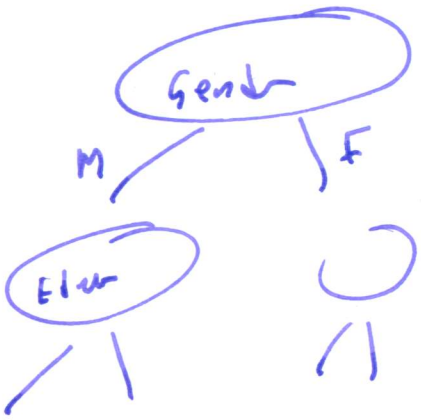
When we are building decision trees we are looking out what to split on.

Suppose our data contains the attributes

Gender = Male, Female
(12 own) (11 own)

Education = University, Other
(20 own) (20 21 own)

We need to build a tree.

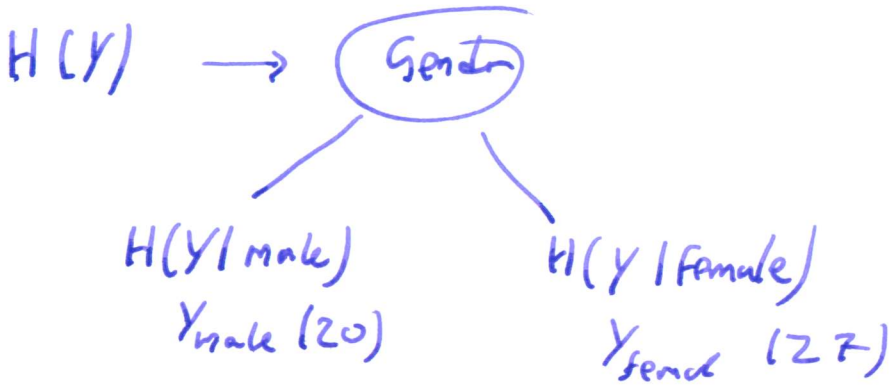


but which one?

We can think of the nodes as sets of observations & we need to make a decision on what to split

We keep splitting until we gets get sets that only contain Car owners or non-Car owners.

To decide how to split we use information gain



$Y_{male} = \{y \in Y \mid y \text{ is male}\}$

In our example we have 20 males, 27 females

With Conditional Entropy you need to scale.

So

$$\text{information-gain}(Y, \text{Gender}) = H(Y) - \left(\frac{|Y_{male}|}{|Y|} H(Y_{male}) + \frac{|Y_{female}|}{|Y|} H(Y_{female}) \right)$$

For example

$$H(Y_{male}) = -\frac{12}{20} \cdot \log_2 \frac{12}{20} - \frac{8}{20} \log_2 \frac{8}{20}$$

owns a car — does not own a car

$$= 0.9710$$

IG for Gender

$$= 0.9917 - \left(\frac{20}{47} \cdot 0.9710 + \frac{27}{47} \cdot 0.9751 \right)$$

$$= 0.025$$

If instead we split on education

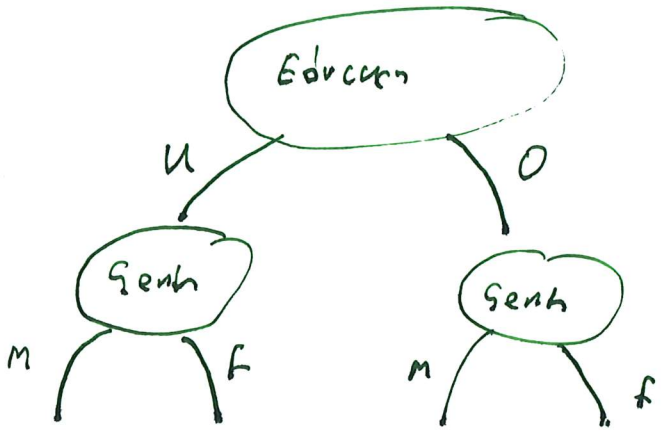
$$H(Y) - \left(\frac{|Y_u|}{|Y|} \cdot H(Y_u) + \frac{|Y_o|}{|Y|} \cdot H(Y_o) \right)$$

$$H(Y_u) = -\frac{2}{22} \cdot \log_2\left(\frac{2}{22}\right) - \frac{20}{22} \cdot \log_2\left(\frac{20}{22}\right)$$

$$H(Y_o) = -\frac{21}{24} \cdot \log_2\left(\frac{21}{24}\right) - \frac{4}{24} \cdot \log_2\left(\frac{4}{24}\right)$$

So the information gain on splitting on education would be 0.553

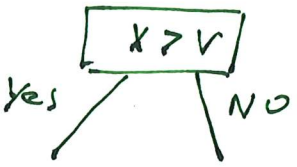
So we split on education first.



To work out the leaves you would have to look at for example the gender of the 2 University educated car owners.

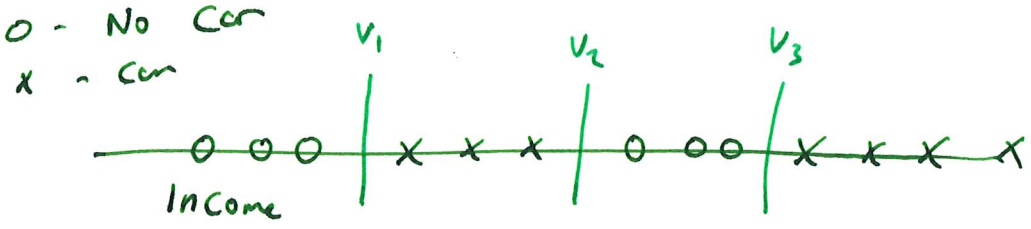
So far all of our features have been Categorical

If you have numerical values then you want to build trees



But how do we decide which value V?
We could try all the possible values. This is not so efficient. One useful observation is that the value V must come from the training set.

If X was say the income we would sort the data



You then pick values V between the groups of car & non-car owners.

You simply use the information gain on the new feature.

ID3 is only one learning algorithm.
There are others & other information gain measures. See for CART with Gini impurity or C4.5 with the gain ratio.

It is easy to overfit with decision trees. This is because they are very expressive. You can limit the depth to avoid this.

Extensions to avoid overfitting.

Bagging :- Split your sample into smaller subsets

Train a tree on each & then build a classifier that returns the average.

Random forests

Instead of dividing up the training set divide the features into random smaller sets & train on each subset of features.

The intuition is that you are trying to learn small trees that work on a small number of features.