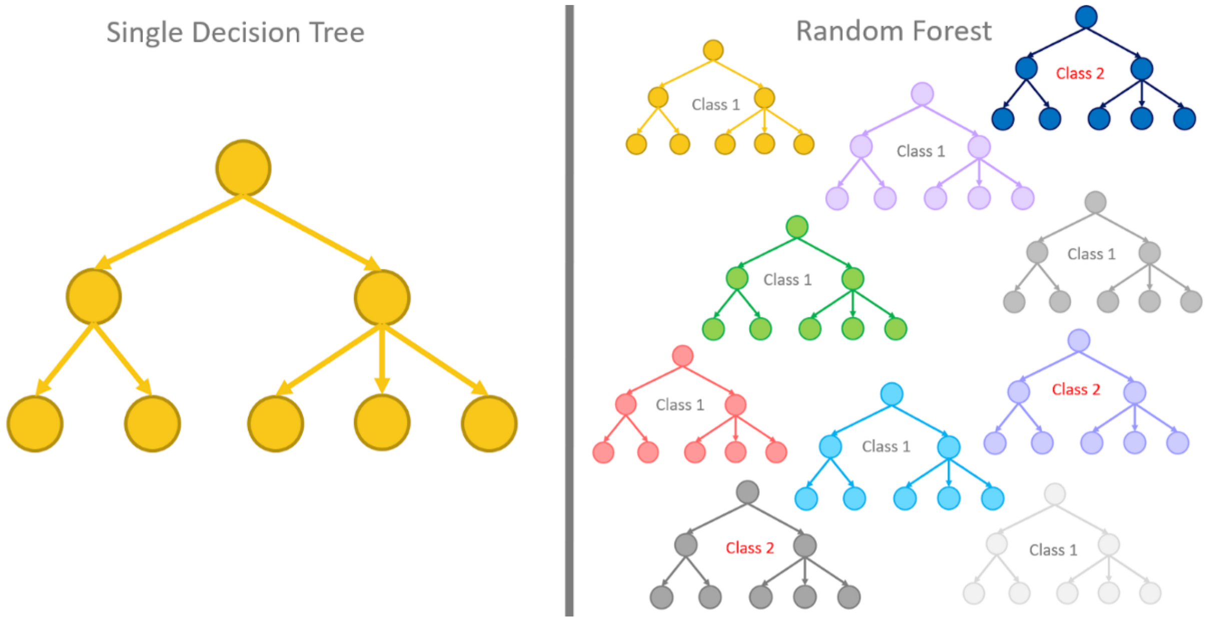
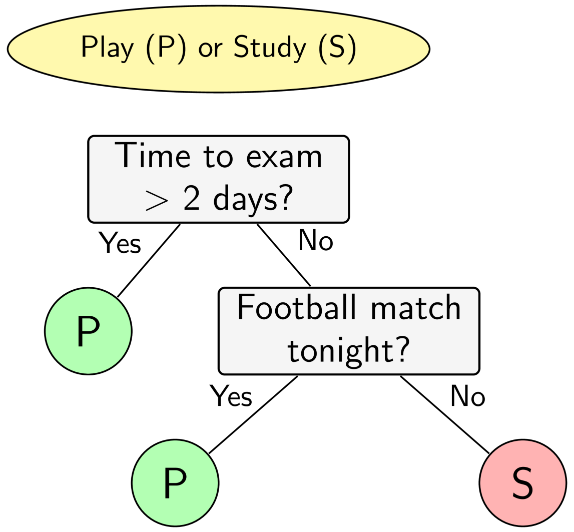
LESSON 4: DECISION TREE



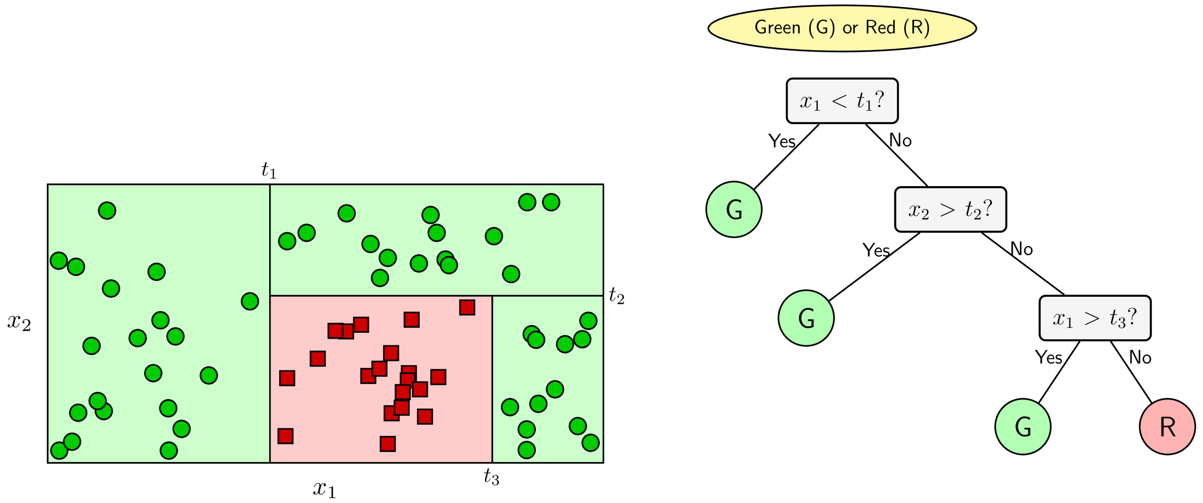
This lecture was refered by machinelearningcoban.com

1. Decision tree introduction



Decision tree is a machine learning model which simulated human thinking while making decision.

Decision tree can be seen as a series of **if ... else ...** logic.



(a) (b)

Red, grey and green pattern on the tree are called **node**.

The red and green node are called **leaf node** or **terminal node**.

The grey one contains a question and is called **non-leaf node**.

The yellow one is call the **root**.

The **non-leaf node** can contains one or multiple **child node**.

The **child node** can be the **non-leaf node** or the **leaf node**.

All **child nodes** which are from one **non-leaf node** are call **sibling node**.

The tree whose all **non-leaf nodes** have only two **child nodes** is called **binary decision tree**.

The questions in the **non-leaf nodes** of the **binary decision tree** can be converted into yes/no question.

The tree whose **non-leaf nodes** have more than two **child nodes** can be converted into the **binary decision tree** because almost all questions can be converted into yes/no question.

2. Entropy function for Decision tree

One problem of decision tree model is how to choose the attribute to split our data samples of each node?

One simple way is choosing the best attribute base on one specific criteria for each node.

But, what is the criteria?

The good attribute to split our data sample is the attribute to create the leaf node (or almost the leaf node).

The bad attribute to split our data sample is the attribute cannot split our data clearly.

And we use the **entropy function** to evaluate the purity of the splitting method.

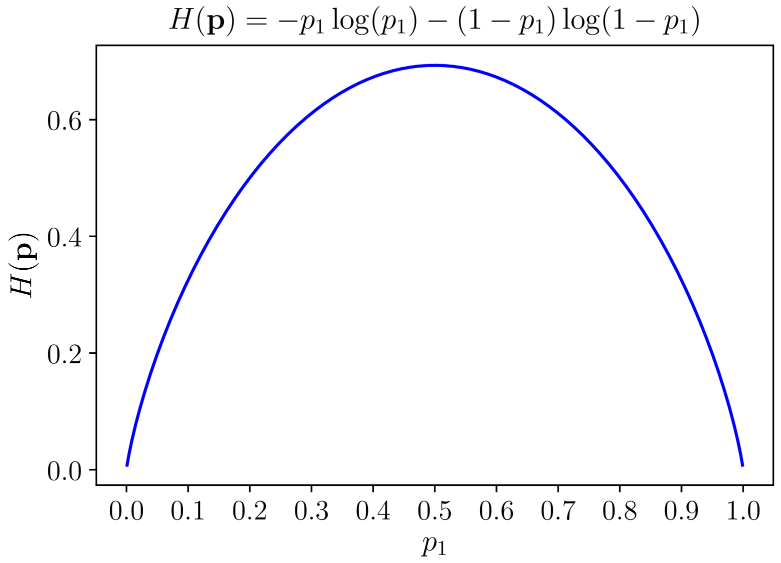
The higher value of the **entropy function**, the more impurity of the splitting method.

The lower value of the **entropy function**, the more purity of the splitting method.

We have a discrete variable x which can be one of n values $x_1, x_2, \dots, x_n$. We have the probability of x is equal to value x_i is p_i $(0 \leq p_i \leq 1, \sum_{i=1}^{n} p_i = 1)$. So we have $p = (p_1, p_2, \dots, p_n)$.

$$H(p) = -\sum_{i=1}^{n} p_i \log(p_i)$$

If n = 2 (means x can be one of 2 values)

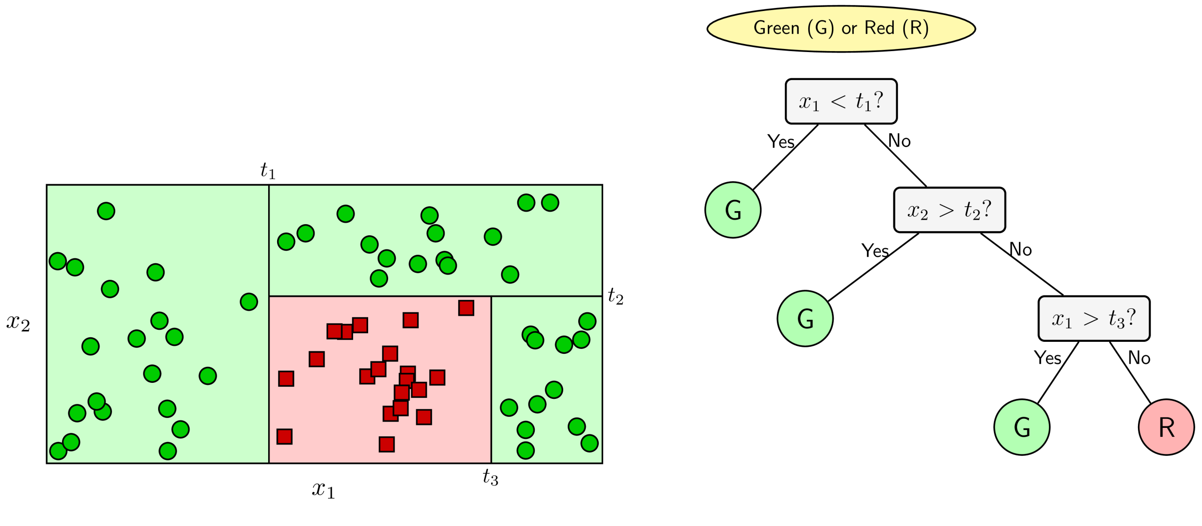


Generally, if n > 2 , $H(p)$ will be smallest if we have $p_i = 1$ and $H(p)$ will be largest if every p_i is similar.

3. Iterative Dichotomiser 3 (ID3)

ID3 is also called **entropy-based decision tree** because it uses the entropy function as its loss function.

Specifically, ID3 model try to choose the attribute for each node to minimize the sum of all entropy value each node.



(a)

(b)

We have a classification problem with C classes. For a non-leaf node, we have S is a set of data points and S contains N data points.

For each class c in C, we have N_c data samples which belong to class c and we have the probability of a random sample belong to class c is $\frac{N_c}{N}$.

We have the entropy of this node,

$$H(S) = -\sum_{c=1}^{C} \frac{N_c}{N} \log\left(\frac{N_c}{N}\right)$$

Assume we choose attribute x for this node and x can create K child nodes $S_1, S_2, \ldots, S_K$ with number of each child node are $m_1, m_2, \ldots, m_K$.

$$H(x, S) = \sum_{k=1}^{K} \frac{m_k}{N} H(S_k)$$

The **information gain** on x is calculated by

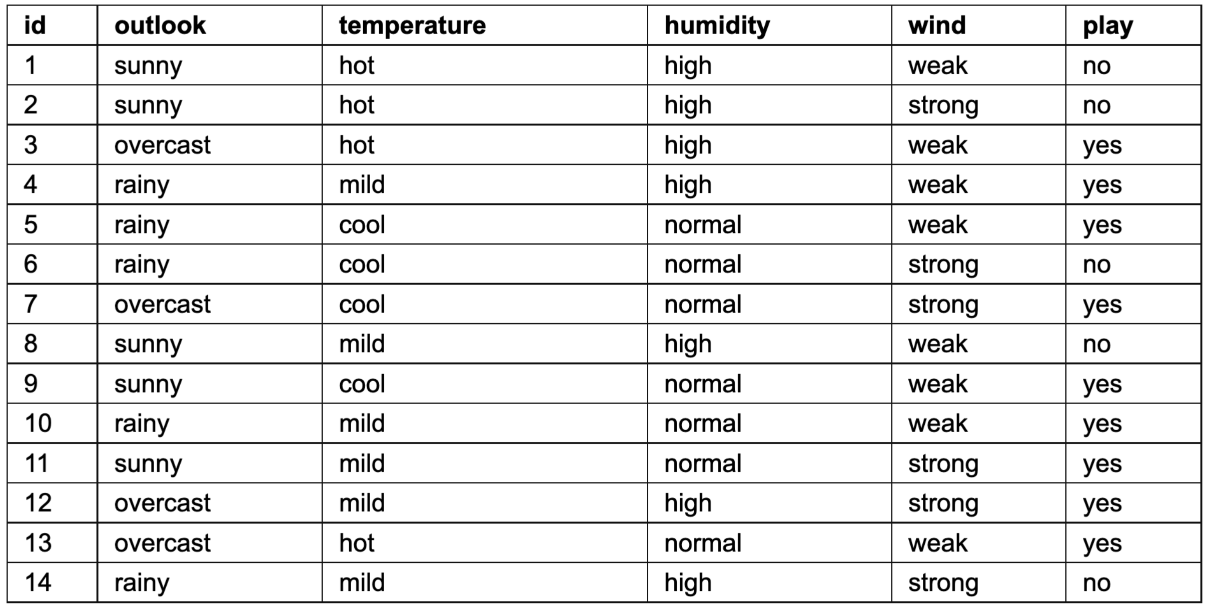
$$G(x, S) = H(S) - H(x, S)$$

After each attribute of node, we want to **gain lots of information** or **reduce the entropy of root node**.

$$x^* = \arg\max_x G(x, S) = \arg\min_x H(x, S)$$

4. Example

We have an example of using the dataset below



id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	no

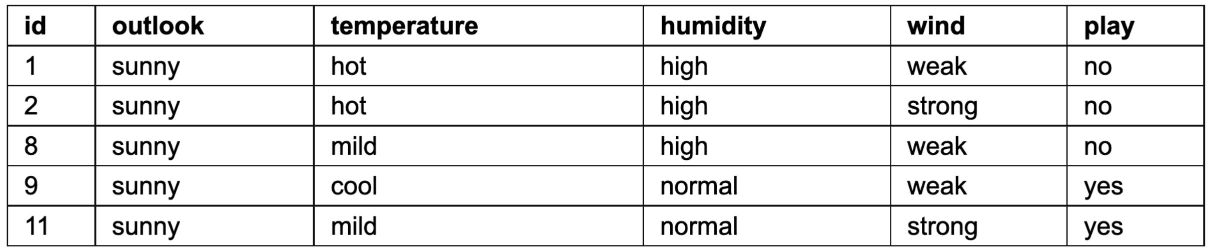
We will use ID3 to solve this dataset

STEP 1: We calculate the entropy of root node.

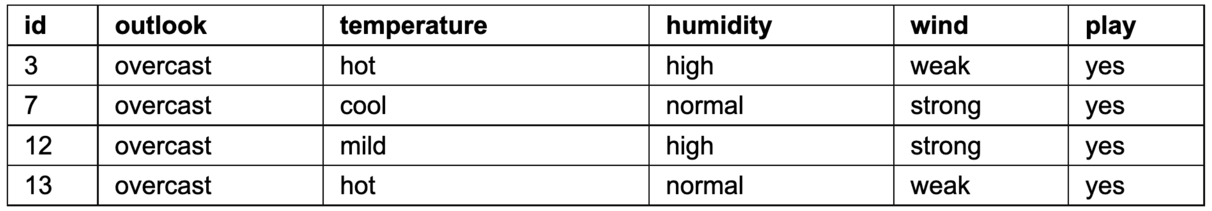
$$H(S) = -\frac{5}{14} \log\left(\frac{5}{14}\right) - \frac{9}{14} \log\left(\frac{9}{14}\right) \approx 0.65$$

STEP 2: We calculate the entropy of each attribute to choose the first attribute in our tree

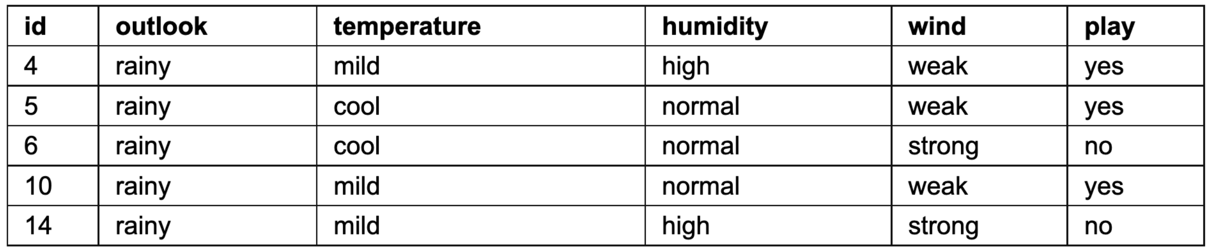
For attribute **outlook**



id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
11	sunny	mild	normal	strong	yes



id	outlook	temperature	humidity	wind	play
3	overcast	hot	high	weak	yes
7	overcast	cool	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes



id	outlook	temperature	humidity	wind	play
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
10	rainy	mild	normal	weak	yes
14	rainy	mild	high	strong	no

$$H(S_s) = -\frac{2}{5}\log\left(\frac{2}{5}\right) - \frac{3}{5}\log\left(\frac{3}{5}\right) \approx 0.673$$

$$H(S_o) = -\frac{0}{4}\log\left(\frac{0}{4}\right) - \frac{4}{4}\log\left(\frac{4}{4}\right) = 0$$

$$H(S_r) = -\frac{3}{5}\log\left(\frac{2}{5}\right) - \frac{3}{5}\log\left(\frac{3}{5}\right) \approx 0.673$$

$$H(outlook, S) = \frac{5}{14}H(S_s) + \frac{4}{14}H(S_o) + \frac{5}{14}H(S_r) \approx 0.48$$

For attribute **temperature**

id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
13	overcast	hot	normal	weak	yes

id	outlook	temperature	humidity	wind	play
4	rainy	mild	high	weak	yes
8	sunny	mild	high	weak	no
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
14	rainy	mild	high	strong	no

id	outlook	temperature	humidity	wind	play
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
9	sunny	cool	normal	weak	yes

$$H(S_h) = -\frac{2}{4}\log\left(\frac{2}{4}\right) - \frac{2}{4}\log\left(\frac{2}{4}\right) \approx 0.693$$
$$H(S_m) = -\frac{4}{6}\log\left(\frac{4}{6}\right) - \frac{2}{6}\log\left(\frac{2}{6}\right) \approx 0.637$$
$$H(S_c) = -\frac{3}{4}\log\left(\frac{3}{4}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right) \approx 0.562$$
$$H(temperature, S) = \frac{4}{14}H(S_h) + \frac{6}{14}H(S_m) + \frac{4}{14}H(S_c) \approx 0.631$$

Similarly, for attribute **humidity** and **wind**, we have:

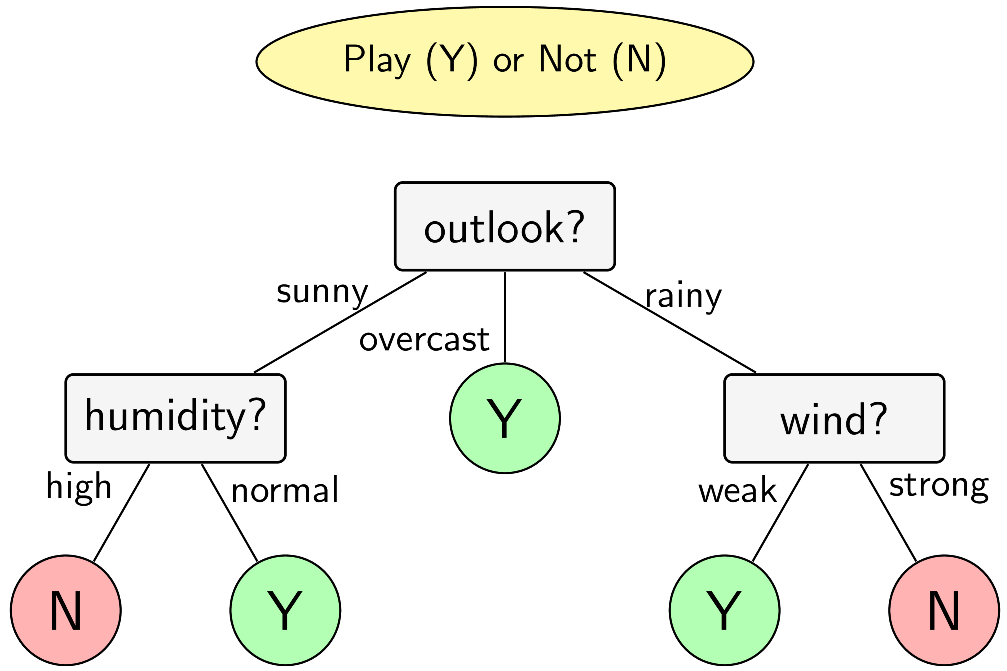
$$H(outlook, S) \approx 0.48$$
$$H(temperature, S) \approx 0.631$$
$$H(humidity, S) \approx 0.547$$
$$H(wind, S) \approx 0.618$$

Because $H(outlook, S)$ is the smallest, we choose **outlook** to be the first attribute in our ID3.

STEP 3: We continue to calculate the entropy of each attribute to choose the attribute for each child node of **outlook**

For `outlook = sunny` , we use **humidity** to have the zero entropy (`humidity = high => play = no` and `humidity = normal => play = yes`).

For `outlook = rainy` , we use **wind** to have the zero entropy (`wind = weak => play = yes` and `wind = strong => play = no`).



5. Problems of ID3 and Decision tree

The biggest problem of ID3 or Decision tree is **overfitting**.

And we have some conditions to stop creating new nodes or stop splitting the data points:

- `entropy = 0` means all data points of this node belong to one class. We stop.
- `number of data points < a threshold` makes some data points in this node will be mis-classified. We stop.
- `distance between the node and root > a threshold` makes the tree more complicate. We stop.
- `number of leaf nodes > a threshold` makes the tree more complicate. We stop.
- `information gain < a threshold` doesn't reduce the entropy much. We stop

Another solution to reduce **overfitting** is **pruning**.

First, we train the ID3 until all the data points in the training set is classified exactly.

Second, we prune several leaf nodes and make their non-leaf nodes become leaf nodes.

Third, we can base some criterias to evaluate the pruning process:

- We can use a validation set to evaluate
- We can add a regularization term to the loss function
  - Because all the data points in the training set is classified exactly, the `loss value = 0`.
  - We add a term λK to the loss function and start the pruning process.
  - We need to balance the entropy term and the λK to optimize the model

In [ ]: