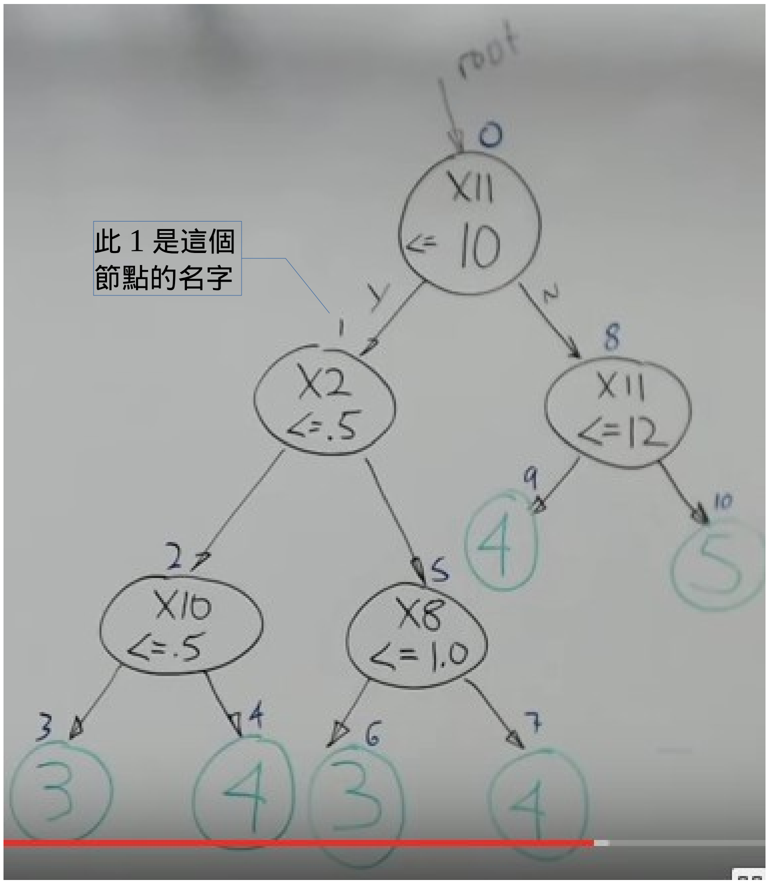

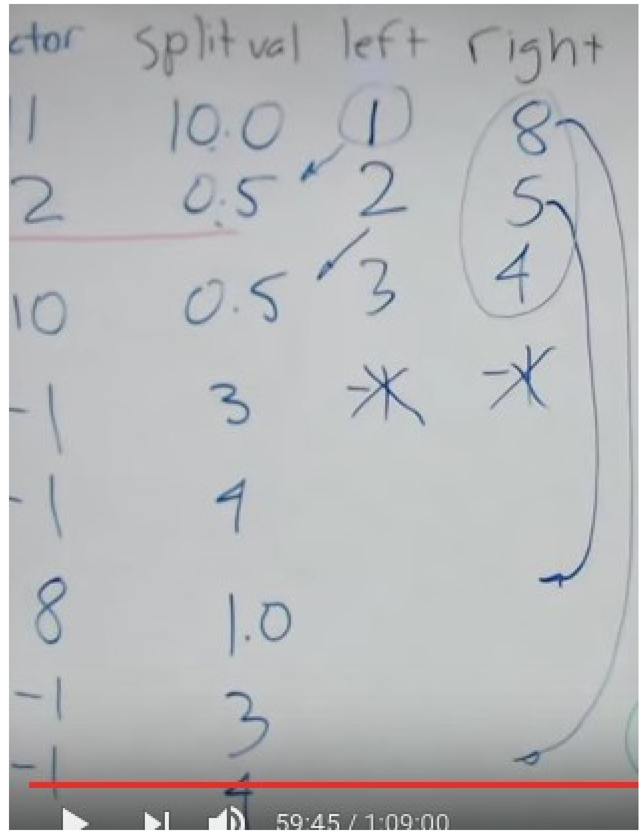

以上右圖是保存 decision tree 的方法, 就是保存在一個數組中. 其中 left 為 1 表示 left child 為 名字叫 1 的節點, right 同理.

Decision Tree: Tabular View

node	Factor	SplitVal	Left	Right
0	X11	9.900	1	8
1	X11	9.250	2	5
2	X2	0.748	3	4
3	Leaf	3.000	NA	NA
4	Leaf	4.000	NA	NA
5	X2	0.648	6	7
6	Leaf	6.000	NA	NA
7	Leaf	5.000	NA	NA
8	X2	0.410	9	12
9	X11	10.900	10	11
10	Leaf	6.000	NA	NA
11	Leaf	8.000	NA	NA
12	X11	10.700	13	14
13	Leaf	7.000	NA	NA
14	Leaf	5.000	NA	NA

Decision Tree Algorithm (JR Quinlan)

```
build_tree(data)
    if data.shape[0] == 1: return [leaf, data.y, NA, NA]
    if all data.y same: return [leaf, data.y, NA, NA]
    else
        determine best feature i to split on
        SplitVal = data[:,i].median()
        lefttree = build_tree(data[data[:,i]<=SplitVal])
        righttree = build_tree(data[data[:,i]>SplitVal])
        root = [i, SplitVal, 1, lefttree.shape[1] + 1]
        return (append(root, lefttree, righttree))
```

How to determine "best" feature?

Goal: Divide and conquer

Group data into most similar groups.

Approaches:

- Information gain: Entropy
- Information gain: Correlation
- Information gain: Gini Index

row #	X2	X10	X11	Y		Tree				
correl	-0.731	0.406	0.826			node	Factor	SplitVal	Left	Right
0	0.885	0.330	9.100	4.000		0	11	?	?	?
1	0.725	0.390	10.900	5.000		1				
2	0.560	0.500	9.400	6.000		2				
3	0.735	0.570	9.800	5.000		3				
4	0.610	0.630	8.400	3.000		4				
5	0.260	0.630	11.800	8.000		5				
6	0.500	0.680	10.500	7.000		6				
7	0.320	0.780	10.000	6.000		7				
						8				
						9				
						10				
						11				
						12				
						13				
						14				

選跟 Y 的 correlation 最大的(X11)作為 root.

Random Tree Algorithm (A Cutler)

```
build_tree(data)
    if data.shape[0] == 1: return [leaf, data.y, NA, NA]
    if all data.y same: return [leaf, data.y, NA, NA]
    else
        determine random feature i to split on
        SplitVal = (data[random,i] + data[random,i]) / 2
        lefttree = build_tree(data[data[:,i]<=SplitVal])
        righttree = build_tree(data[data[:,i]>SplitVal])
        root = [i, SplitVal, 1, lefttree.shape[0] + 1]
        return (append(root, lefttree, righttree))
```

Strengths and weaknesses of decision tree learners

- Cost of learning?
- Cost query?
- Don't have to normalize your data