CART : Classification & Regression Tree

CART is a powerful tree based method which can be used for regression modelling and for classification tasks.

- CART is a non-parametric approach and does not require any distributional assumption.

Before we talk about construction of such trees, let us first look at how it works.

Classification Trees

Classification tree (or a decision tree) is an example of a multistage decision process.

Rather than using the complete set of features jointly to make a output decision, different subsets of features are used at different levels of the tree

Consider the following classification tree for for a 3-class problem with 6-dimensional feature vector $\underset{\sim}{x} = (x_1, x_2, x_3, x_4, x_5, x_6)'$.

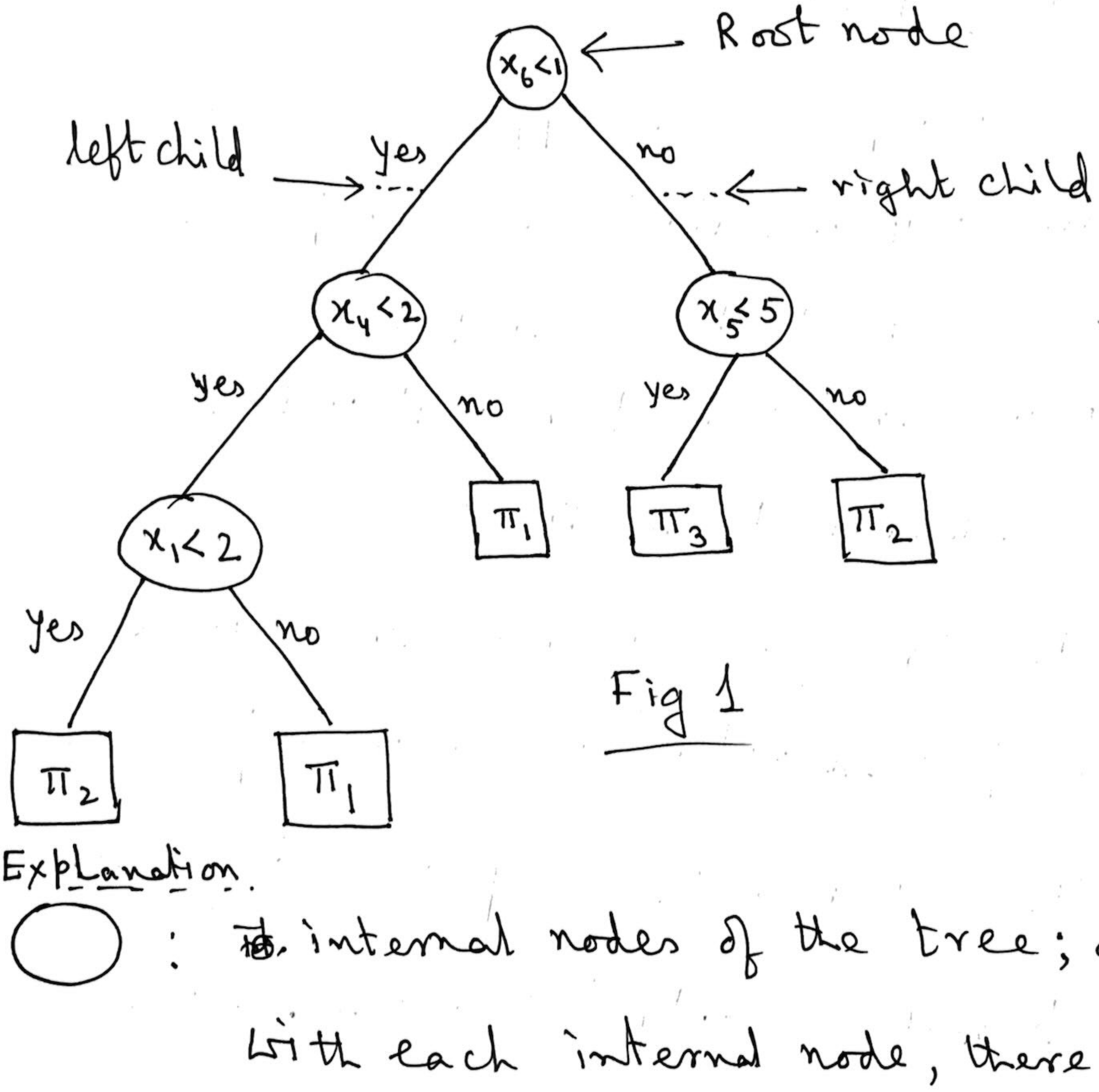


Fig 1

Explanation.

◯ : internal nodes of the tree; associated with each internal node, there is a variable and a threshold

□ : these are called terminal node or leaf: there is a class label associated with each leaf

The top most internal node is the "root" of the tree

Remark: Construction of classification tree would revolve around

(i) how to choose the variable associated with any internal node (including the root node)

(ii) how to find the threshold associated with the variable at any internal node

(iii) how to assign class label at the terminal nodes.

(iv) when to declare a node ~~at~~ as terminal

How to use the classification tree?

Suppose we have a feature vector

$$\underset{\sim}{x} = (5, 3, 6, 2, 1, 3)' \text{ and}$$

wish to use classification tree of Fig 1 to classify the above feature vector.

S1: We begin with the root node; compare the value of x_6 (i.e. 3) with the threshold (which is 1) and find that threshold is exceeded and hence take the right side path (Called the ~~left~~ right child of the node)

S2 : We arrive at the internal node $x_5 < 5$

We compare the value of the x_5 variable in the feature (i.e. 1) with the threshold at that node (i.e. 5) and find that the threshold is not exceeded; hence take the left side path (the left child)

S3 : We arrive at a terminal node with the last path direction. The terminal node that we have reached has a class Label π_3 and hence the allotment of the given feature vector is π_3, i.e. we assign $\underset{\sim}{x} = (5, 3, 6, 2, 1, 3)'$ to π_3

The rule looks so simple!!

Isn't it!!

<u>Remark</u>: The tree classifier in Fig 1 is an example of a binary decision tree.

<u>Remark</u>: Every internal node induces a partition of feature space. These partitions induced are hyperplanes parallel to the coordinate axes.

<u>Remark</u>: The classification partition is induced by the set of terminal nodes

<u>Remark</u>: For every internal node $t \in T$, there is a subspace $U(t) \in \mathcal{X}$. This $U(t)$ is the union of the subspaces of the terminal nodes that are it's descendants (i.e. the terminal nodes below t in the tree structure).

<u>Example</u>: Classification partitions

2-class problem: π_1 & π_2

2-dimensional feature vector

$$\underset{\sim}{x} = (x_1, x_2)$$

Classification tree in given in Fig 2

Remark: The tree classifier in Fig 1 is an example of a "binary decision tree". More generally, the outcome of a decision at any node can also be more than 2.

Remark: Binary trees partition the feature space into 2 parts. The partitions induced (as in Fig 1) are hyperplanes parallel to the coordinate axes.

Let me try to illustrate the concept of partitions induced by such trees with help of a simple example

Example: 2-class problem : Π_1 & Π_2

2-dimensional feature vector

$$\underset{\sim}{x} = (x_1, x_2)'$$

Let the constructed classification tree be given by Fig 2

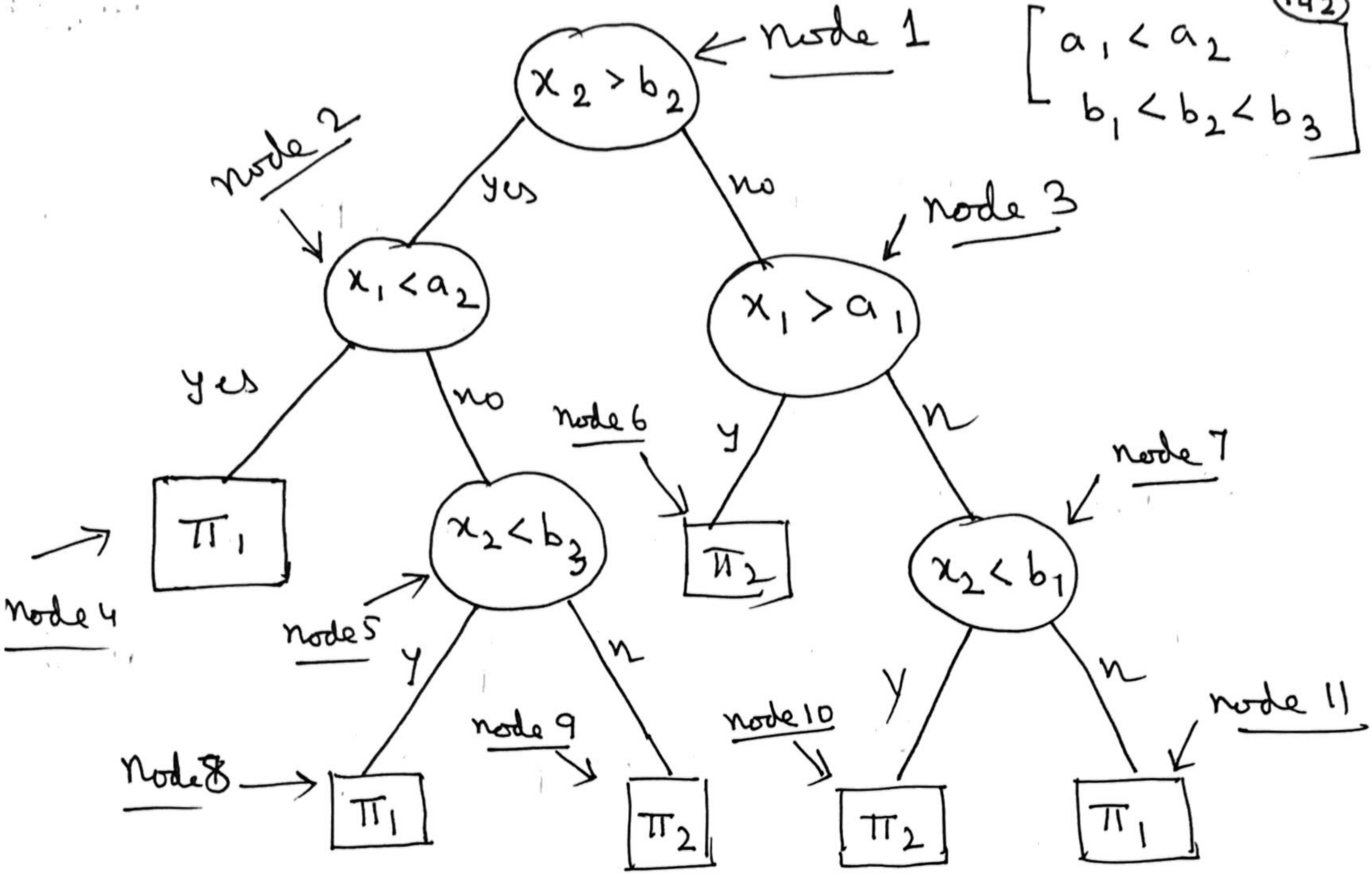


Fig 2.

To see how the above leads to partition of the feature space with class labels for partition regions follow the sequence of figures.

node 1 $\Rightarrow$

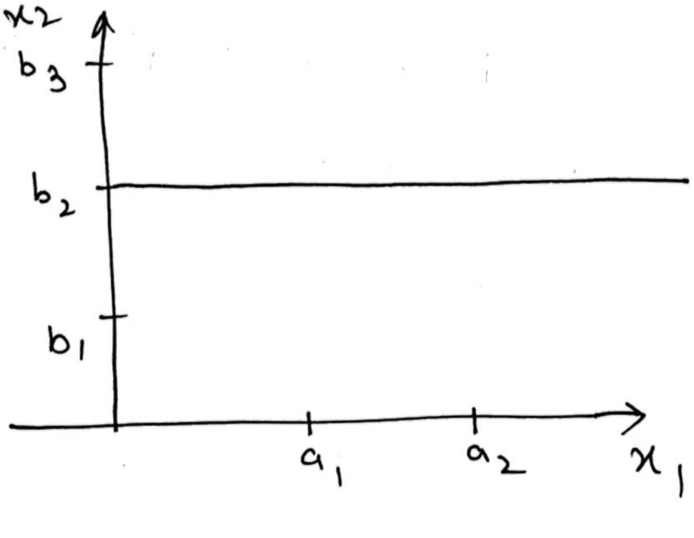


Fig 2.1

node 2 $\Rightarrow$

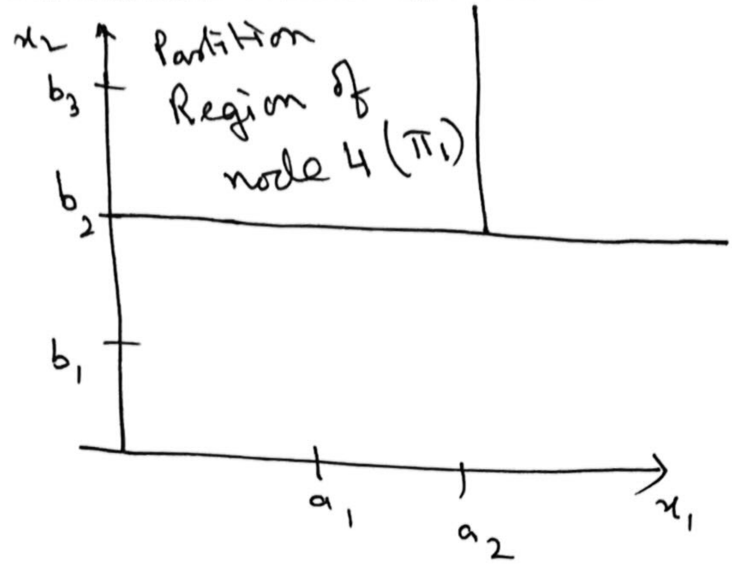


Fig 2.2

node 5 $\Rightarrow$

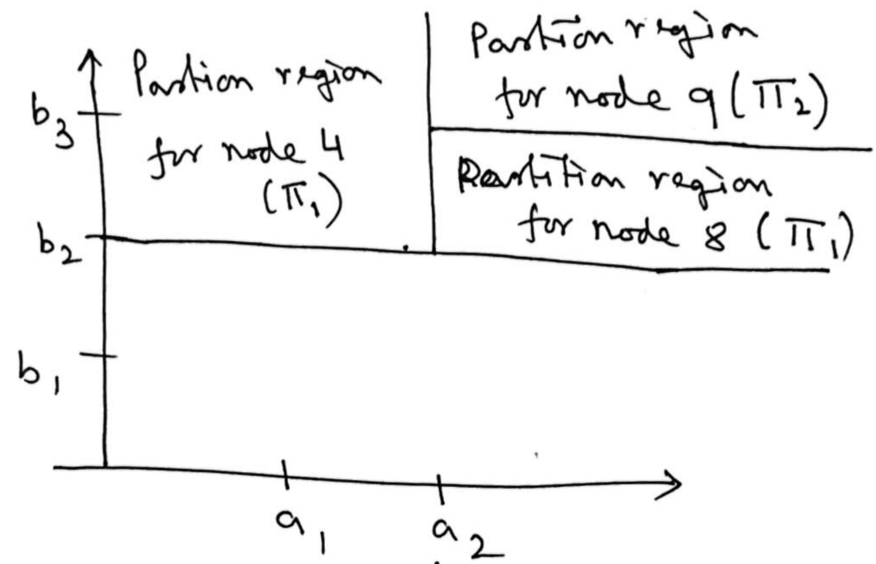


Fig 2.3

node 3 $\Rightarrow$

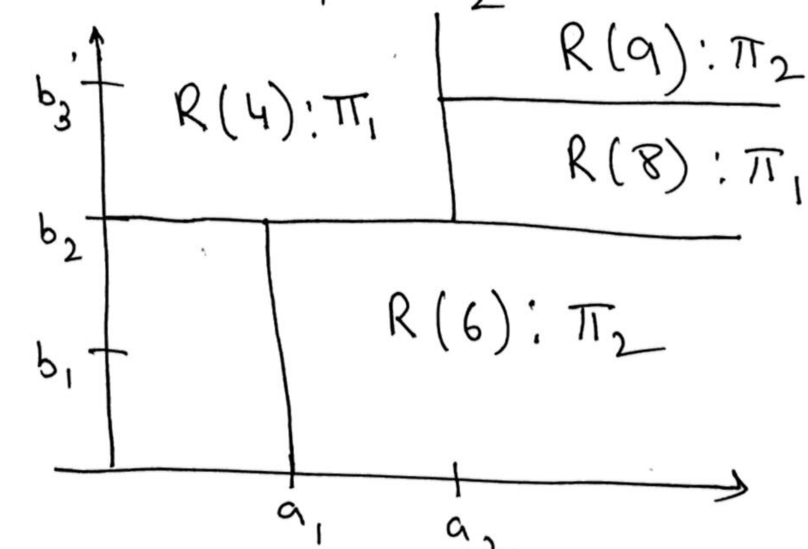


Fig 2.4

Finally, node 7 $\Rightarrow$

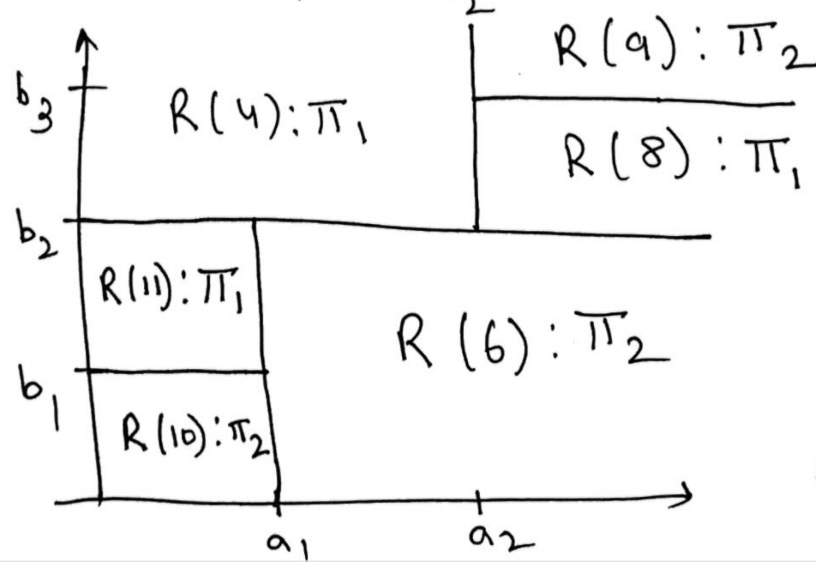
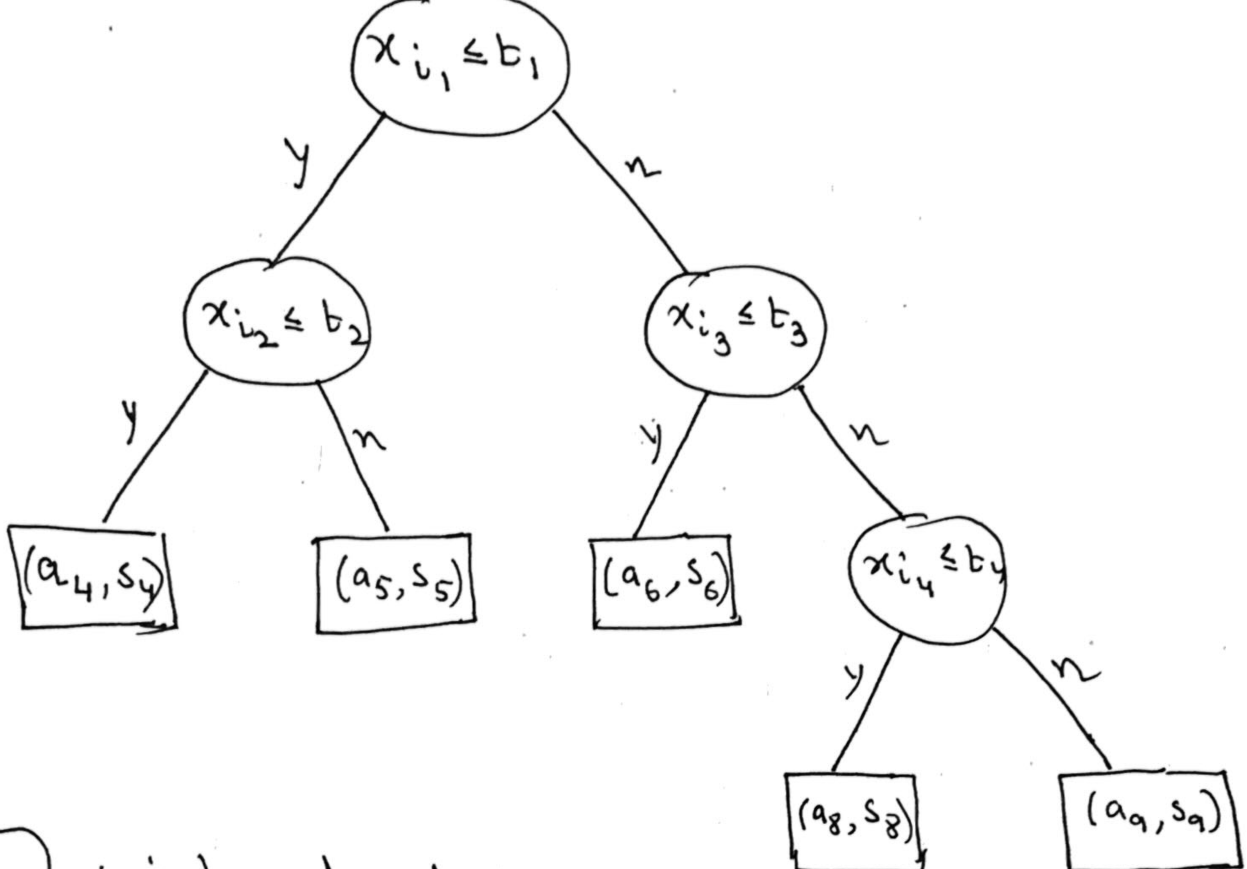


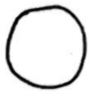
Fig 2.5

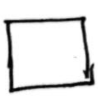
This is the final partition of the feature space that the tree classifier induces.

Regression Trees

Consider the following example of a regression tree



 : internal nodes

 : terminal nodes

i_j : j^{th} split variable

t_j : split point value

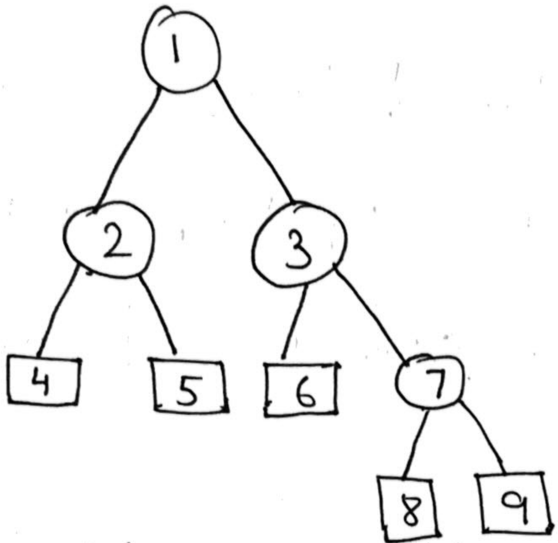
a_i : avg of all the response values of patterns in $\mathcal{L}$ reaching terminal node i.

s_i : st. dev of - i.

Remark : Regression trees look very much the same as classification trees, with split variables & split points. The terminal nodes are characterized by avg of responsevariable values (in contrast to class labels as in classification trees) of patterns in the learning set $\mathcal{L}$ that reaches a particular terminal node.

Remark : As in the classification trees, regression trees also induce a partition of the feature vector space. The predicted value of the response of a feature vector reaching a particular terminal node (i.e. belonging to the corresponding partition) would typically be the average of all the responses in the learning set (under sq error loss), whose feature vectors ~~of~~ ~~which~~ ~~patterns~~ belongs to that partition.

Remark: suppose the following is a classification or a regression tree



T : tree defined by the nodes $\{1, 2, \ldots, 9\}$

$\tilde{T}$: set of terminal nodes $= \{4, 5, 6, 8, 9\}$

Partition induced by the tree is defined through the terminal nodes

$$U(4),\ U(5),\ U(6),\ U(8),\ U(9)$$

each $U(i) \in \mathcal{X} \leftarrow$ feature space

$$U(i) \cap U(j) = \phi$$

$$i \neq j \ ; \ i, j = 4, 5, 6, 8, 9$$

$$U(4) \cup U(5) \cup U(6) \cup U(8) \cup U(9) = \mathcal{X}$$

Towards tree construction

Some basic definitions

Tree: A tree is defined to be a set T of positive integers together with 2 functions $l(.)$ and $r(.)$ from T to $T \cup \{0\}$.

Each member of T corresponds to a node in the tree.
$l(t)$ and $r(t)$ are left node and right node values of t $(\forall\, t \in T)$ $\ni$

(i) $\forall\, t \in T$, either $l(t)$ and $r(t)$ are both 0 (in such a situation t is terminal) or $l(t)$ and $r(t)$ are both > 0 (internal node)

(ii) apart from the root node $(t=1)$ there is a unique parent $s \in T \ni \forall\, t \neq 1$ $\exists$ an $s \in T \ni t = l(s)$ or $t = r(s)$.

<u>Subtree</u> A subtree is a non-empty subset T_1 of T together with 2 f^ns $l_1(\cdot)$ and $r_1(\cdot)$ $\ni$

$$l_1(t) = \begin{cases} l(t), & \text{if } l(t) \in T_1 \\ 0, & \text{o/w} \end{cases}$$

& $$r_1(t) = \begin{cases} r(t), & \text{if } r(t) \in T_1 \\ 0, & \text{o/w.} \end{cases}$$

and T_1, $l_1(\cdot)$ and $r_1(\cdot)$ form a tree.

<u>Prunned subtree</u>: A subtree that has same root as the original tree

<u>Terminal node set</u>:

$\tilde{T}$: set of terminal nodes.

$$\tilde{T} = \{t_1, t_2, \ldots, t_M\} ; \; t_i \in T$$

<u>Partition induced by a tree T</u>

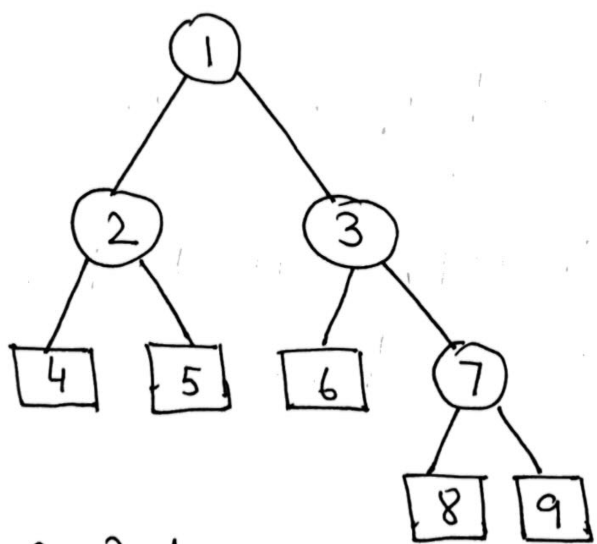
$\{U(t) : t \in \tilde{T}\}$: partition of feature space induced by T with terminal node set $\tilde{T}$

$$U(t) \cap U(s) = \phi \;\; \forall \, t \neq s ; \; t, s \in \tilde{T}$$

and $$\bigcup_{t \in \tilde{T}} U(t) = \mathcal{X}$$

Note! Remember that associated with every non-terminal node t, there is a subspace $U(t)$ of $\mathcal{X}$, which is union of the subspaces of the terminal nodes that are its descendants.

Example:



$T = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$

$\tilde{T} = \{4, 5, 6, 8, 9\}$

t	$l(t)$	$r(t)$
1	2	3
2	4	5
3	6	7
4	0	0
5	0	0
6	0	0

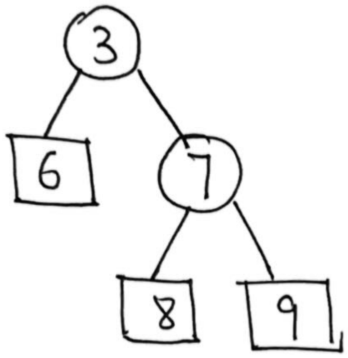
t	$l(t)$	$r(t)$
7	8	9
8	0	0
9	0	0

enough to know the tree structure

Let

$T_1 = \{3, 6, 7, 8, 9\}$

$\underline{\underline{T_1}}$

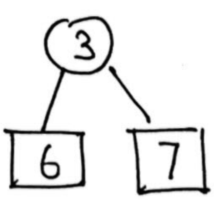


$$l_1(t) = \begin{cases} l(t), & \text{if } l(t) \in T_1 \\ 0, & \text{o/w} \end{cases}$$

$$r_1(t) = \begin{cases} r(t), & \text{if } r(t) \in T_1 \\ 0, & \text{o/w} \end{cases}$$

t	$l_1(t)$	$r_1(t)$
3	6	7
6	0	0
7	8	9
8	0	0
9	0	0

$T_2 = \{3, 6, 7\}$

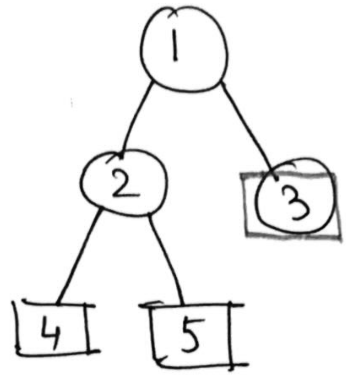


$\underline{\underline{T_2}}$

t	$l_2(t)$	$r_2(t)$
3	6	7
6	0	0
7	0	0

Also, take

$$T_3 = \{1, 2, 3, 4, 5\}$$



$\underline{\underline{T_3}}$

t	$l_3(t)$	$r_3(t)$
1	2	3
2	4	5
3	0	0
4	0	0
5	0	0.

<u>Remark</u> : T_1, T_2, T_3 are all subtrees of T

T_1 is <u>not a prunned subtree</u> of T

T_2 is a prunned subtree of T_1 but is not a prunned subtree of T.

T_3 is a prunned subtree of T

<u>Note</u> : $\{2, 3, 4, 5\}$ is not a subtree, as 2 and 3 has no parent.