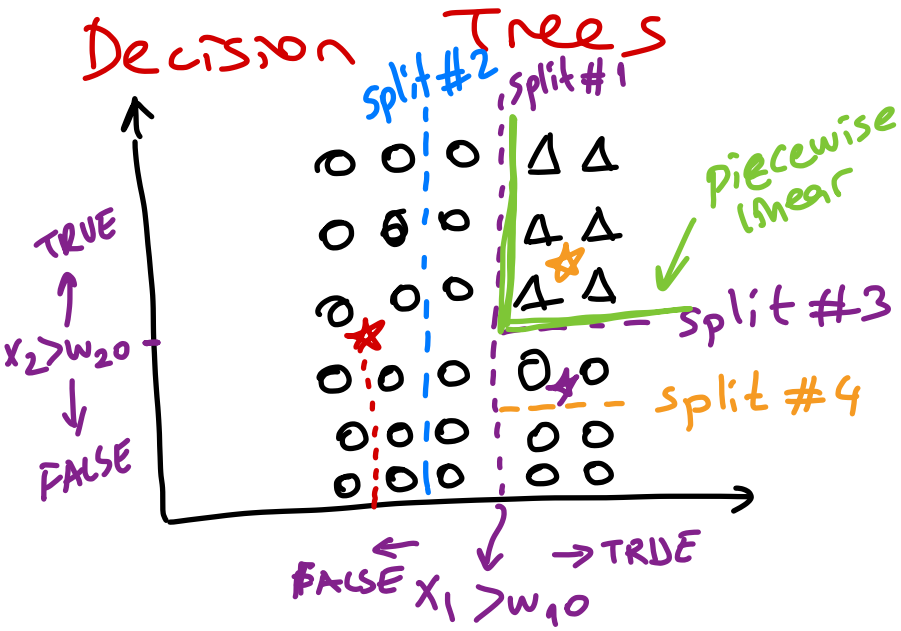
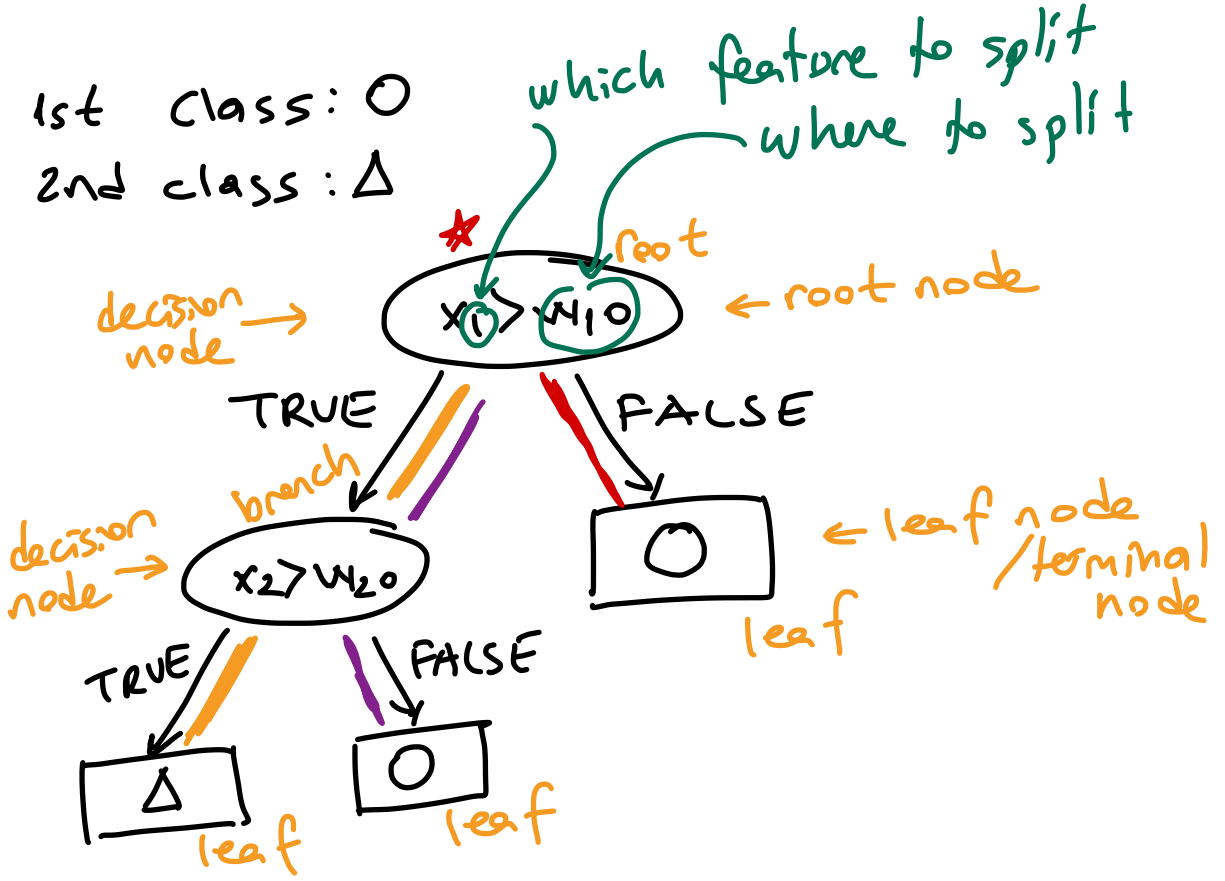
Decision Trees



1st class: O

2nd class: Δ



★ predicted label= ?

$x_1 > w_{10}$

FALSE

O

★ predicted label= ?

$x_1 > w_{10}$

TRUE

$x_2 > w_{20}$

TRUE

Δ

★ predicted label= ?

$x_1 > w_{10}$

TRUE

$x_2 > w_{20}$

FALSE

O

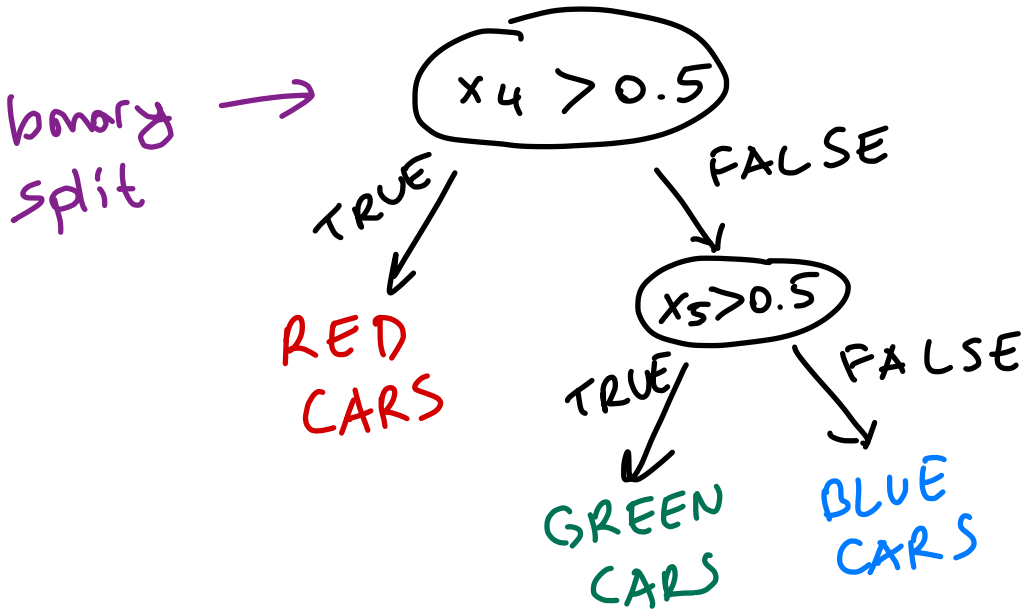
if $x_1 \leq w_{10} \Rightarrow \hat{y} = O$

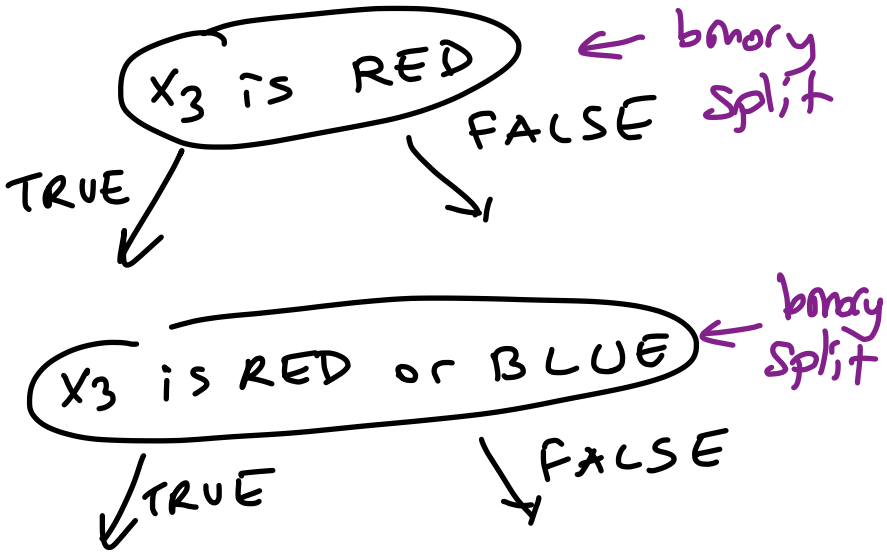
if $x_1 > w_{10}$ & $x_2 \leq w_{20} \Rightarrow \hat{y} = O$

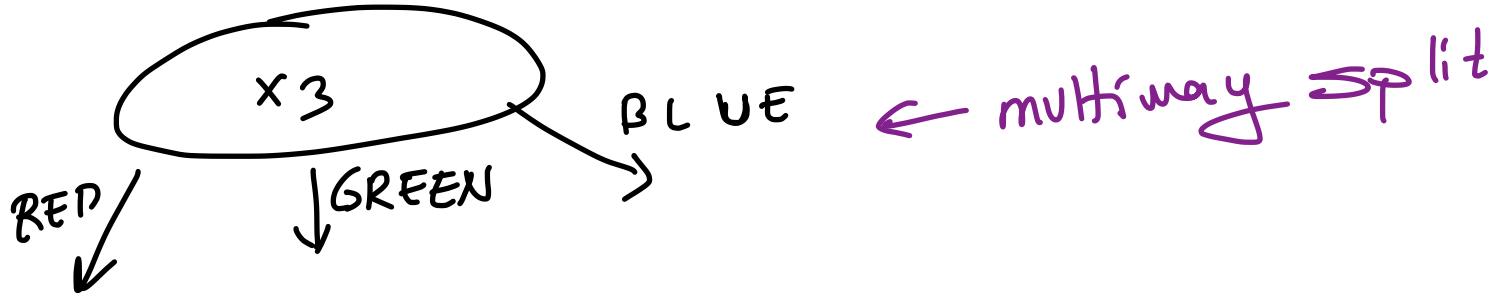
if $x_1 > w_{10}$ & $x_2 > w_{20} \Rightarrow \hat{y} = \Delta$

$X_3 = \{$ RED, GREEN, BLUE $\}$ $\Rightarrow$

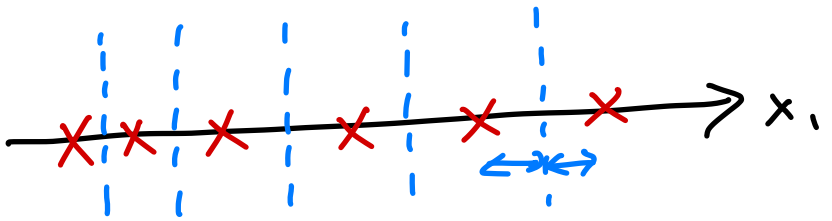
	X_4	X_5	X_6
R	1	0	0
G	0	1	0
B	0	0	1







How can we learn on which feature and where to split?



N data points $\Rightarrow (N-1)$ possible splits

D features $\Rightarrow D(N-1)$ possible splits in total.

Univariate Trees

Each decision node uses only one feature.

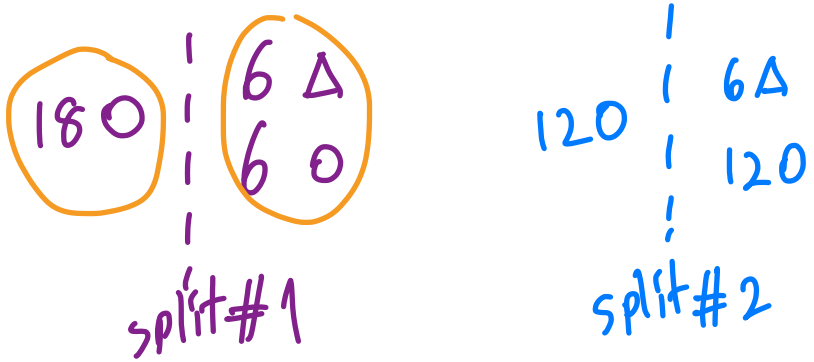
$$f_m(x): \quad x_j > w_{m0} \quad [x_j = w_{m0}]$$

TRUE

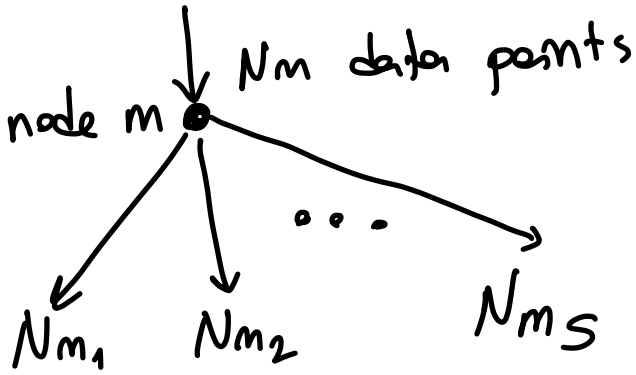
Left child $L_m = \{x \mid x_j > w_{m0}\}$ $[x_j = w_{m0}]$

FALSE

Right child $R_m = \{x \mid x_j \le w_{m0}\}$ $[x_j \neq w_{m0}]$



Goodness of a split ⇒ Is Split#1 better than Split#2?



S = # of splits (branches)

N_m = # of data point that reach node m

K = # of classes.

$$N_m = N_{m_1} + N_{m_2} + \cdots + N_{m_S}$$

$$N_m = N_{m,1} + N_{m,2} + \cdots + N_{m,K}$$

$$N_m = \sum_{s=1}^{S} N_{m_s} \quad \text{(splits)}$$

$$N_m = \sum_{c=1}^{K} N_{m,c} \quad \text{(classes)}$$

$$p_{mc} = \hat{P}r(y = c \mid x_m) = \frac{N_{m,c}}{N_m}$$

$0 \cdot \log_2(0) \triangleq 0$

$$I_m = -\sum_{c=1}^{K} p_{mc} \log_2(p_{mc})$$

Impurity of a node

$$I_m = -\left[\frac{18}{18}\cdot\log_2\left(\frac{18}{18}\right) + \underbrace{\frac{0}{18}\cdot\log_2\left(\frac{0}{18}\right)}_{0\cdot\log_2(0)}\right] = 0$$

18 { 18 ○ , 0 △ }

$$I_m = -\left[\frac{6}{12}\cdot\log_2\left(\frac{6}{12}\right) + \frac{6}{12}\cdot\log_2\left(\frac{6}{12}\right)\right] = 1$$

12 { 6 ○ , 6 △ }

12 0

0 Δ

$$I_m = -\left[\frac{12}{12}\log_2\left(\frac{12}{12}\right) + \frac{0}{12}\cdot\log_2\left(\frac{0}{12}\right)\right] = 0$$

12 0

6 Δ

$$I_m = -\left[\frac{12}{18}\cdot\log_2\left(\frac{12}{18}\right) + \frac{6}{18}\cdot\log_2\left(\frac{6}{18}\right)\right] \cong 0.918$$

$$I_m' = \sum_{s=1}^{S} \frac{N_{m,s}}{N_m}\left[-\sum_{c=1}^{K} P_{msc}\log_2(P_{msc})\right]$$

class index, split index, node index

I_m': Impurity of a split

$\frac{N_{m,s}}{N_m}$: weights

P_{msc}: impurity of a child node

I_m': impurity of the split

minimum is better

$$I_m'(\text{split \#1}) = \left[\frac{18}{30}\cdot[0] + \frac{12}{30}\cdot[1]\right] = 0.4$$

$$I_m'(\text{split \#2}) = \left[\frac{12}{30}\cdot[0] + \frac{18}{30}\cdot[0.918]\right] = 0.55$$

Split #1 is better then Split #2.

⇒ at each decision (internal) node
⇒ - for all features
- for all possible splits
- calculate impurity
- pick the best split among all possible splits
⇒ stop when all terminal nodes are "pure"

POSSIBLE PROBLEM ⇒ OVERFITTING (Train accuracy is low)

PRUNING

① Prepruning

(A)
- fix maximum depth
- if you reach this depth, stop

(B)
- you will not split if your node has a specified amount of your data set

② Postpruning

- grow your tree until it is completely pure
- prune your tree step by step until your misclassification error starts increasing on a validation data set.