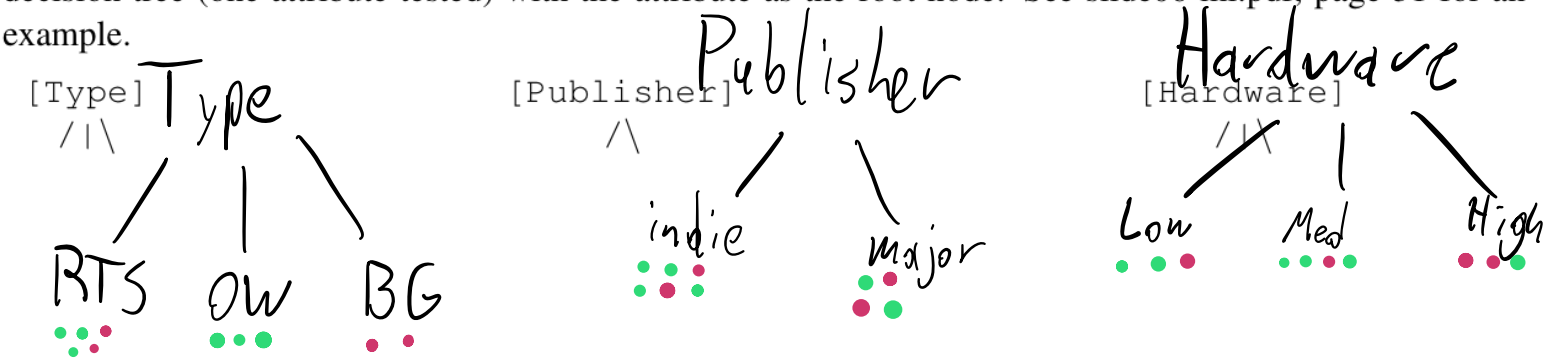
1 Decision Tree Learning

Consider the following set of examples where you are trying to make a decision whether to buy a game or not given the specifications in terms of the attributes Type, Publisher, Hardware (hardware requirements).

Example#	Type	Publisher (Pub)	Hardware Requirments (Req)	Purchase
1	Real Time Strategy (RTS)	Indie	Low	Y
2	Real Time Strategy (RTS)	Major	Low	Y
3	Open World (OW)	Indie	Medium	Y
4	Board Game (BG)	Indie	Low	N
5	Real Time Strategy (RTS)	Major	High	N
6	Board Game (BG)	Major	High	N
7	Open World (OW)	Major	Medium	Y
8	Real Time Strategy (RTS)	Indie	High	Y
9	Real Time Strategy (RTS)	Indie	Medium	N
10	Open World (OW)	Indie	Medium	Y

Problem 1, Written (6 pts): For each attribute (Type, Publisher, and Hardware), draw a depth-one decision tree (one attribute tested) with the attribute as the root node. See slide06-ml.pdf, page 31 for an example.



Problem 2, Written (9 pts): (1) For each of the three cases above, calculate the information gain. (2) Based on the information gains, which attribute would you choose first? Explain why.

$H_{RTS} = H_{parent}$ (same distribution)
$H_{OW} = 0$ (one label)
$H_{BG} = 0$ (one label)
$IG_{Type} = .971 - \frac{5 \cdot .971}{10} = 0.485$

$H_{indie} = -\left(\frac{2}{3}\log_2\frac{2}{3} + \frac{1}{3}\log_2\frac{1}{3}\right) = 0.918$
$H_{major} = -\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1$
$IG_{pub} = .971 - \frac{6 \cdot .918 + 4 \cdot 1}{10} = 0.02$

$H_{Low} = H_{indie}$ (same dist.)
$H_{med} = -\left(\frac{3}{4}\log_2\frac{3}{4} + \frac{1}{4}\log_2\frac{1}{4}\right) = 0.811$
$H_{high} = H_{indie}$ (same dist., opposite labels)
$IG_{hardware} = .971 - \frac{3 \cdot .918 + 4 \cdot .811 + 3 \cdot .918}{10} = 0.095$

$IG = H_{parent} - \frac{\sum_{children} n \cdot H_{child}}{n_{all}}$ n - # data points, H - Entropy

$H = -\sum p \log_2 p$ p - prob(label) Y/N

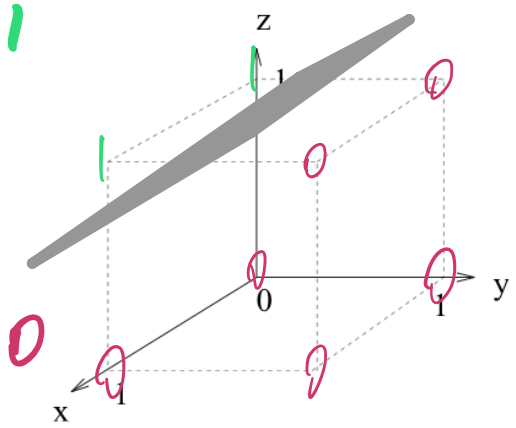
$H_{parent} = -\left(\frac{3}{5}\log_2\frac{3}{5} + \frac{2}{5}\log_2\frac{2}{5}\right)$
$= 0.971$

Type has the highest info gain in first split.

Problem 4, Written (5 pts): Can a single perceptron unit solve the following classification problem?: In other words, can the perceptron learning rule find a set of weights to correctly classify all examples? (1) Answer "yes" or "no" to the question, (2) draw a geometric illustration of the problem in 3D and (3) justify your reasoning.

(Note: for 3D input, the decision boundary is a flat plane.)

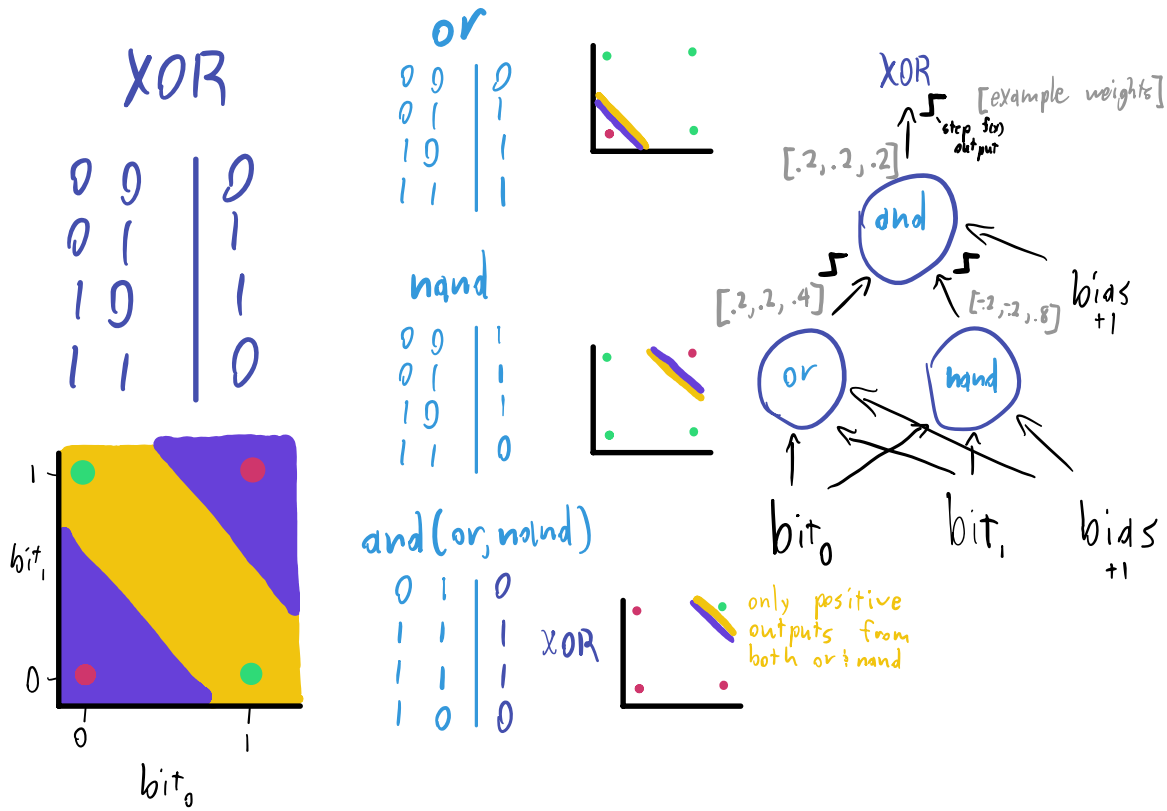
Input x	Input y	Input z	Class
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0



yes.

on the 'bit-cube' there are only 2 labels. They are connected and their edge can be separated by a plane from all of the remaining 0 labels.

Problem 6, Written (5 pts): Show how we can implement the XOR function linking up three perceptron units. You don't need to code anything. Graphically illustrate your idea, both (1) neural network topology, and (2) decision boundary of each unit in the input space.



Problem 7, Written (15 pts): Given $E(w) = \frac{1}{4}(w+5)(w+2)(w-2)(w-7)$, derive a gradient descent learning rule to adjust w. Basically, you need to find:

$$\Delta w(t) = -\alpha \frac{dE}{dw},$$

where α is the learning rate. Note that $\frac{dE}{dw}$ is a function of w.

$E = \frac{1}{4}(w+5)(w^2-4)(w-7)$

$= \frac{1}{4}(w^3+5w^2-4w-20)(w-7)$

$= \frac{1}{4}(w^4-2w^3-39w^2+8w+140)$

$\frac{dE}{dw} = \frac{1}{4}(4w^3-6w^2-78w+8)$

$= w^3-\frac{3}{2}w^2-\frac{39}{2}w+2$

$$\Delta w(t) = -\alpha\left(w_{t-1}^3-\frac{3}{2}w_{t-1}^2-\frac{39}{2}w_{t-1}+2\right)$$