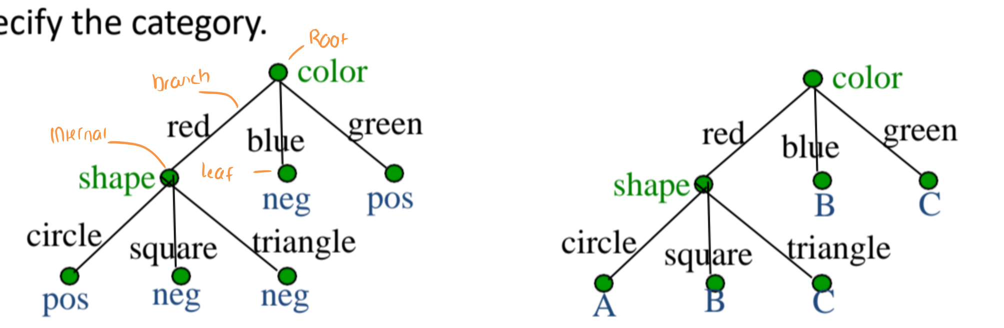
Decision Tree

- Tree-based classifiers for instances represented as feature-vectors. Nodes test features, there is one branch for each value of the feature, and leaves specify the category.



- Can represent arbitrary conjunction and disjunction. Can represent any classification function over discrete feature vectors.
- Can be rewritten as a set of rules, i.e. disjunctive normal form (DNF).
  - red ∧ circle → pos
  - red ∧ circle → A
    blue → B; red ∧ square → B
    green → C; red ∧ triangle → C

Entropy :-

· Measures the amount of uncertainty (impurity) in the dataset

$$\text{Entropy}(S) = -\underbrace{P_1}_{\text{fraction of +ve example}} \log_2(P_1) - \underbrace{P_0}_{\text{fraction of -ve example}} \log_2(P_0)$$

↳ all example in one category $(0 \cdot \log_2 0)$

⇒ Entropy(S) = 0

↳ example are equal $(P_1 = P_0 = 0.5)$

⇒ Entropy(S) = 1

$$\text{Entropy}(S) = \sum_{i=1}^{c} -p_i \log_2(p_i)$$

↳ multi class problem w/ c categories

Information Gain :-

· Measures how well a given attributes seperates the training example

· measures the reduction in entropy after a dataset is split

$$\text{Gain}(S,F) = \text{Entropy}(S) - \sum \underbrace{\frac{|S_v|}{|S|}}_{\text{Fraction of } S_v/S} \text{Entropy}(S_v)$$

- Example:
  - <big, red, circle>: + <small, red, circle>: +
  - <small, red, square>: – <big, blue, circle>: –

examples are equal 2+, 2- ⇒ E = 1

size	color	shape
2+, 2 –: E=1 (starting entropy)	2+, 2 – : E=1	2+, 2 – : E=1
big: 1+,1– (big+ = 1, big- = 1) ⇒ E=1; small: 1+,1– (small+ = 1, small- = 1) E=1	red: 2+,1– (red+ = 2, red- = 1) ⇒ E=0.918; blue: 0+,1– (blue+ = 0, blue- = 1) E=0	circle: 2+,1– E=0.918; square: 0+,1– E=0
Gain=1–(0.5·1 + 0.5·1) = 0	Gain=1–(0.75·0.918 + 0.25·0) = 0.311	Gain=1–(0.75·0.918 + 0.25·0) = 0.311

S_v; entropy (S); S_v/S; entropy (S_v); fraction of n(big)/n(size); gain

Hypothesis Space Search :-

· **Batch learning**

process all training instances at once

· **Greedy search**

Find local optimal solution. May not be the simplest

Overfitting :-

· Category or feature noise cause over/fitting

↳ also cause diff. instance to have diff. class

↳ also arise by incomplete & inadequate features

Pruning

1. Pre Pruning :-

↳ From top to bottom, if data isn't sufficient → stop.

2. Post Pruning :-

↳ Grow the tree, then remove features w/o sufficient data

What to Prune?

1. Cross Validation

↳ split (training data) to two sets → Validation ⇒ evaluate utility of subtrees; → tuning

2. Statistical test

↳ do statistical test on training data set to determine if regularity can be dismiss

3. Minimum description length (MDL)

↳ determine if additional complexity of hypothesis is less complex than just remembering exceptions from pruning

Reduced Error Pruning

post pruning, cross validation → wastes too much data on validation set

↳ 1. split training data → Grow; → Validation

2. take a non-leaf node, n & replace w/ a leaf labelled w/ the current majority class of that node

3. Measure and record the accuracy

4. repeat for all non-leaf node until accuracy on validation set decrease

W/O losing Training Data

↳ grow trees breadth-first rather than depth-first, ⇒ stop growing after reach specified tree complexity

1. run trials of REP using diff. random splits of grow and validation sets

2. Record their complexity & let C = ave. complexity

3. grow a tree breadth-first using all training data stop when complexity ≡ C

ID3 :-

overfitted & overclassified; 1 attribute is tested at a time; can be expensive

- understandable
- fastest
- short tree
- able to prune
- use all dataset

↳ algorithm for building decision tree

↳ Build tree from top-down, no backtrack

↳ Information Gain is most useful

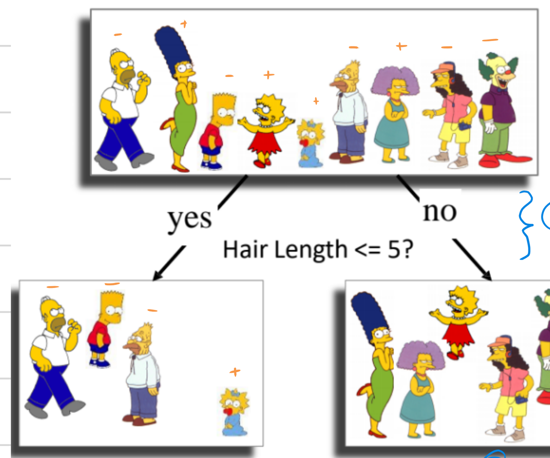
Information Gain :-

↳ Based on decrease in entropy

1. calculate entropy of dataset
2. split into different attributes
3. calculate entropy for each branch. then added proportionally to get total entropy
4. Info. Gain = Entropy − Result Entropy
5. largest Info Gain is chosen as decision Node

E=0 (same category) is labelled as leaves

Predicting Gender



① calculate entropy for all dataset

$$Entropy(S) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n}\log_2\left(\frac{n}{p+n}\right)$$

$$Entropy(4F,5M) = -(4/9)\log_2(4/9) - (5/9)\log_2(5/9) = 0.9911 \rightarrow \text{Entropy}$$

② split into attributes

Let us try splitting on *Hair length* (attributes)

③ calculate entropy for each branch

$$Entropy(1F,3M) = -(1/4)\log_2(1/4) - (3/4)\log_2(3/4) = 0.8113$$

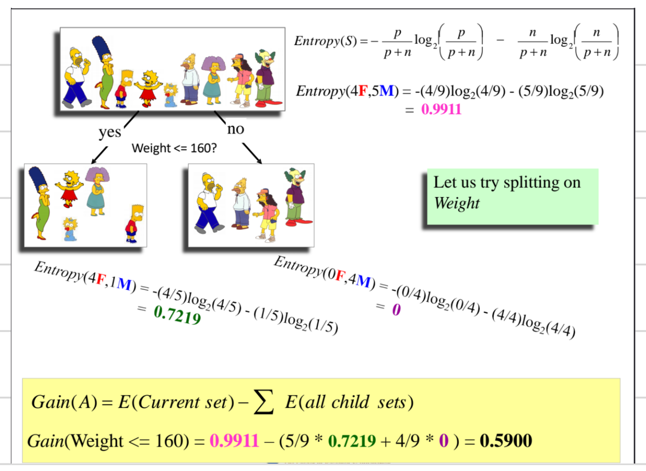
$$Entropy(3F,2M) = -(3/5)\log_2(3/5) - (2/5)\log_2(2/5) = 0.9710$$

④

$$Gain(A) = \underbrace{E(\text{Current set})}_{\text{prev. entropy (this attribute)}} - \sum \underbrace{E(\text{all child sets})}_{\text{total current entropy}}$$

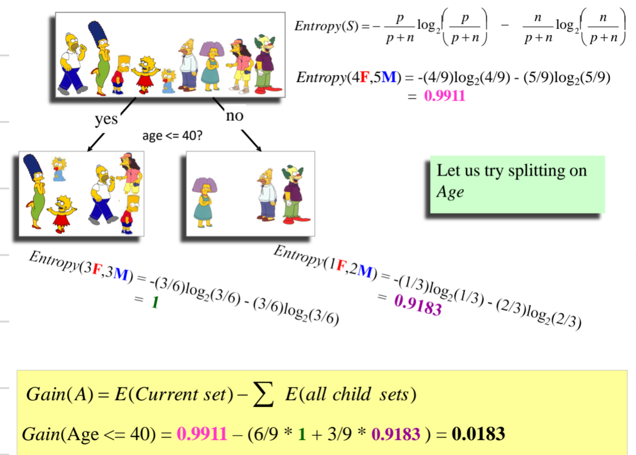
$$Gain(\text{Hair Length} <= 5) = 0.9911 - (4/9 * 0.8113 + 5/9 * 0.9710) = 0.0911$$ info gain

attributes : Weight



IG = 0.59

attributes : Age



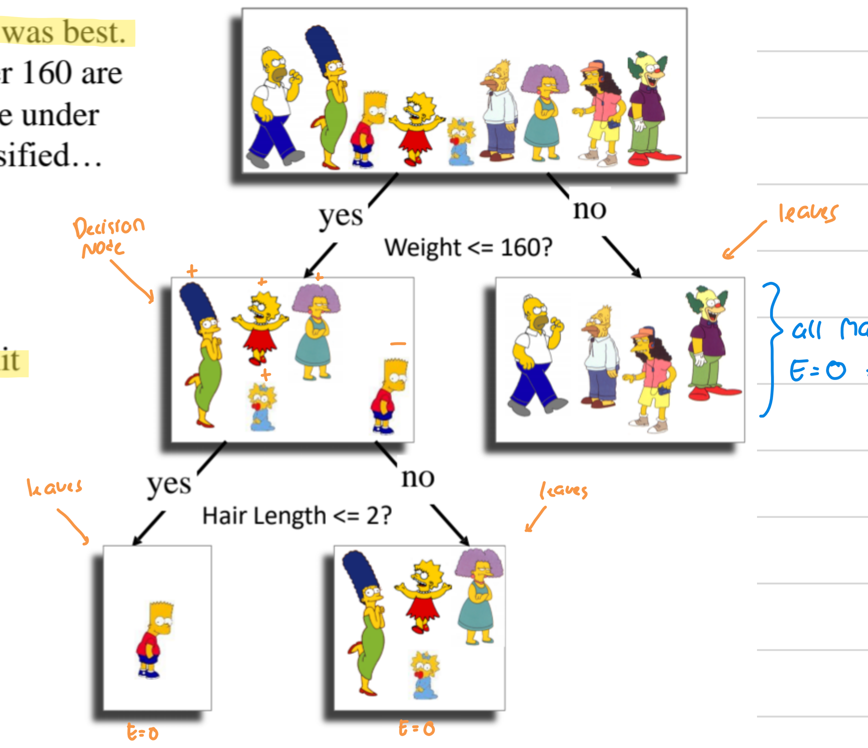
IG = 0.0183

⑤ choose subtree w/ largest gain

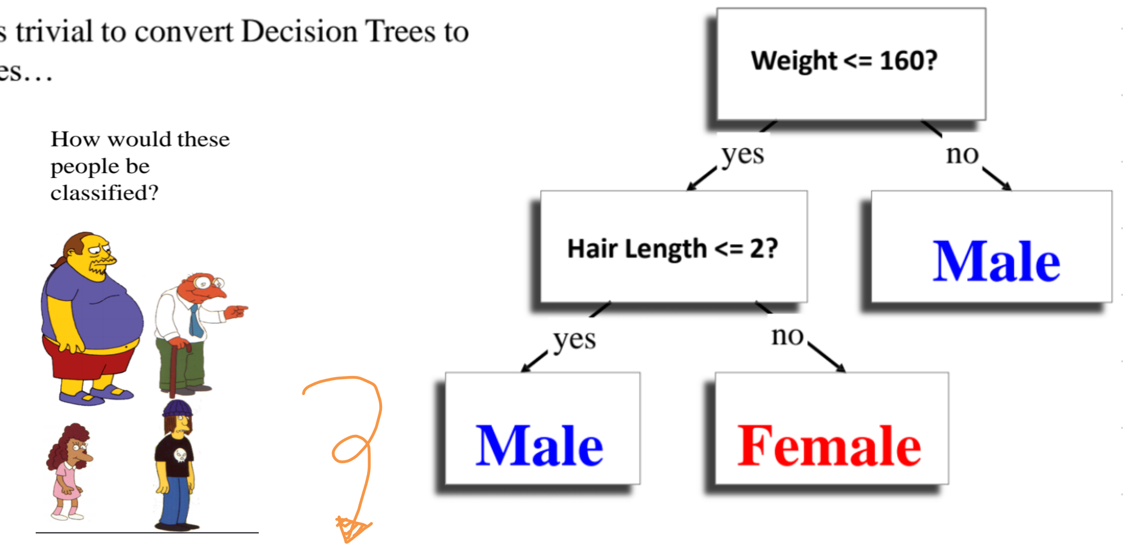
Of the 3 features we had, *Weight* was best. But while people who weigh over 160 are perfectly classified (as males), the under 160 people are not perfectly classified… So we simply recurse!

This time we find that we can split on *Hair length*, and we are done!

2nd attribute



It is trivial to convert Decision Trees to rules…



Rules to Classify Males/Females

If *Weight* **greater than** 160, classify as Male
Elseif *Hair Length* **less than or equal** to 2, classify as Male
Else classify as Female