Decision Trees
& Random Forest

AbdElMoniem Bayoumi, PhD

Fall 2021

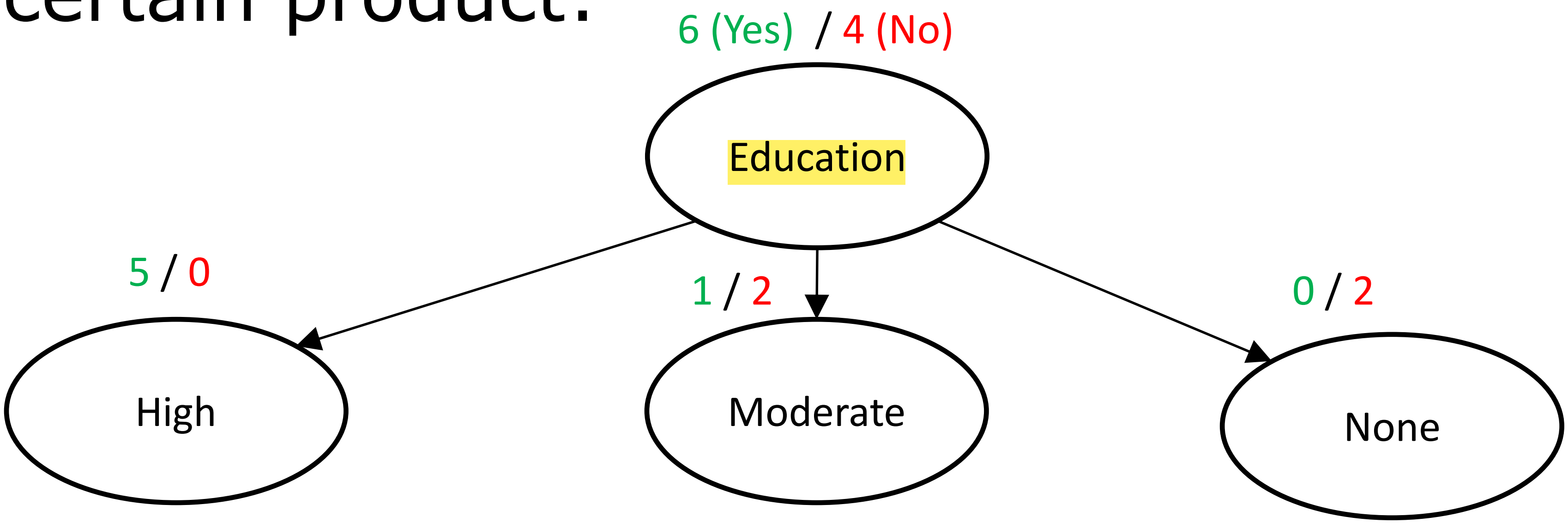
Decision Trees

- Training examples on customers' interest in certain product:

#	Gender	Education	Financial Status	Interested?
1	M	High	R	Y
2	M	Moderate	R	N
3	F	None	P	N
4	M	High	P	Y
5	F	High	M	Y
6	F	None	M	N
7	F	Moderate	P	N
8	F	Moderate	R	Y
9	M	High	R	Y
10	M	High	P	Y
11	**M**	**None**	**P**	**??**

Decision Trees

- Training examples on customers' interest in certain product:



1	M	High	R	Y
4	M	High	P	Y
5	F	High	M	Y
9	M	High	R	Y
10	M	High	P	Y

Pure subset, stop!

2	M	Moderate	R	N
7	F	Moderate	P	N
8	F	Moderate	R	Y

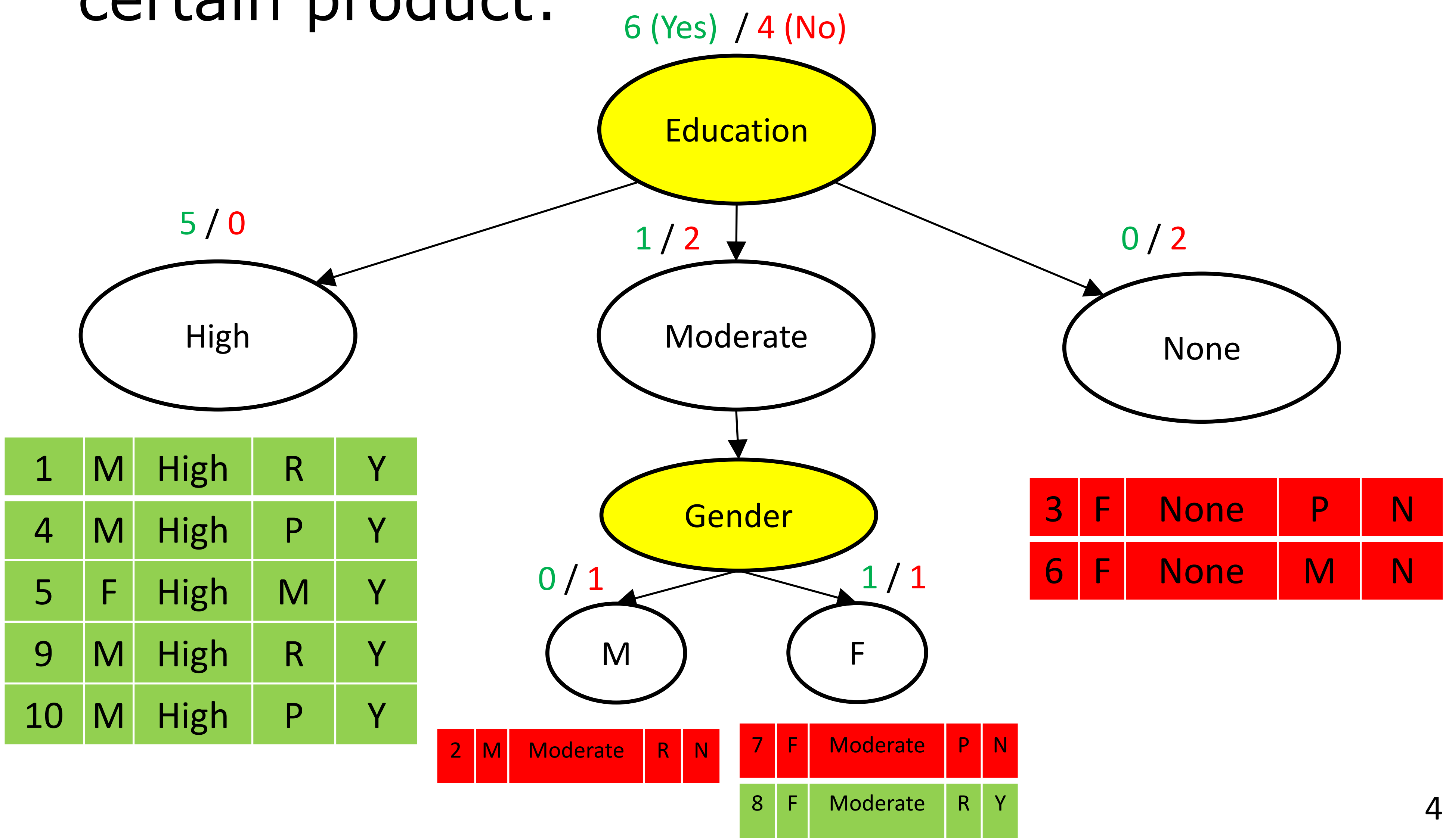
Split again!

3	F	None	P	N
6	F	None	M	N

Pure subset, stop!

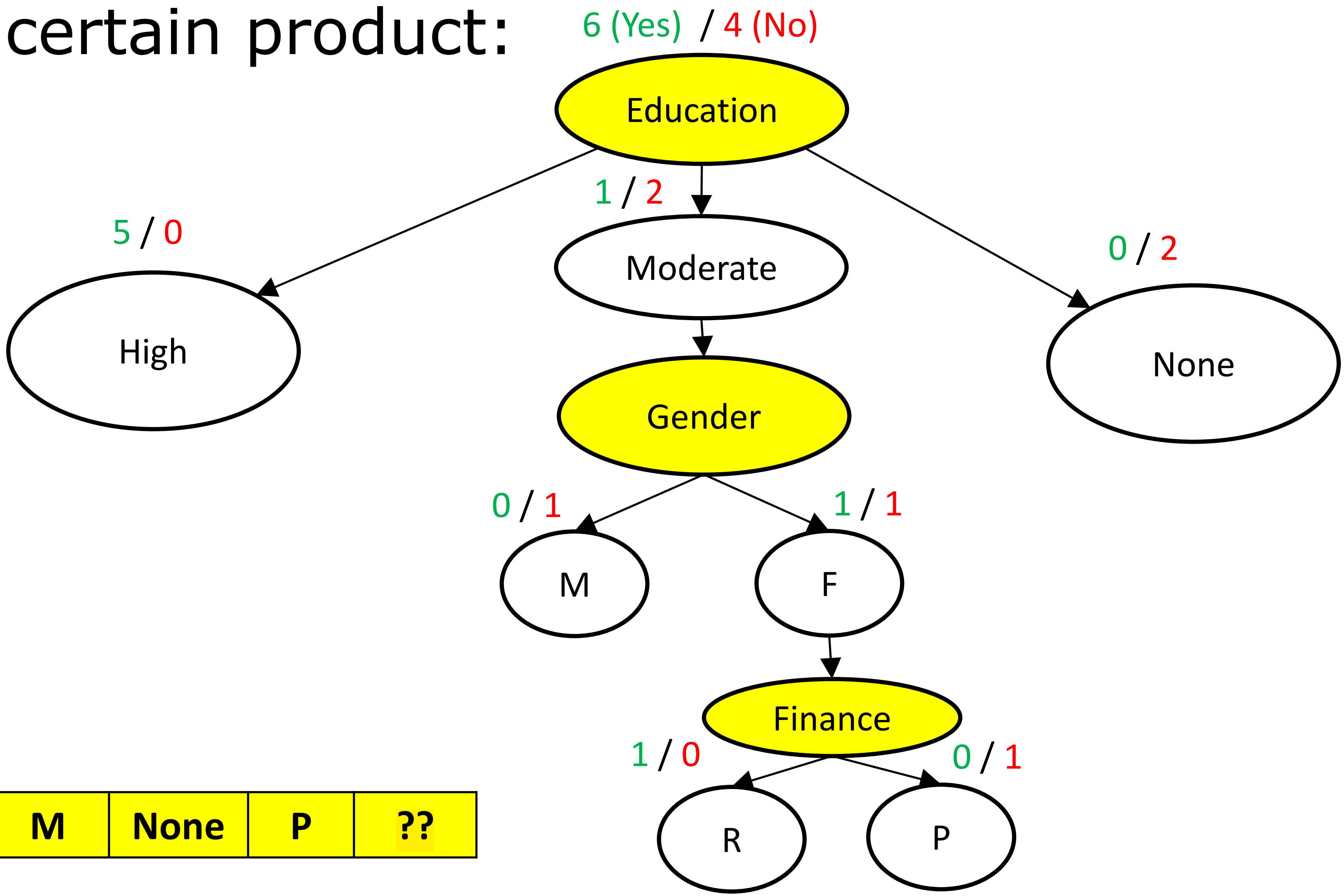
Decision Trees

- Training examples on customers' interest in certain product:



Decision Trees

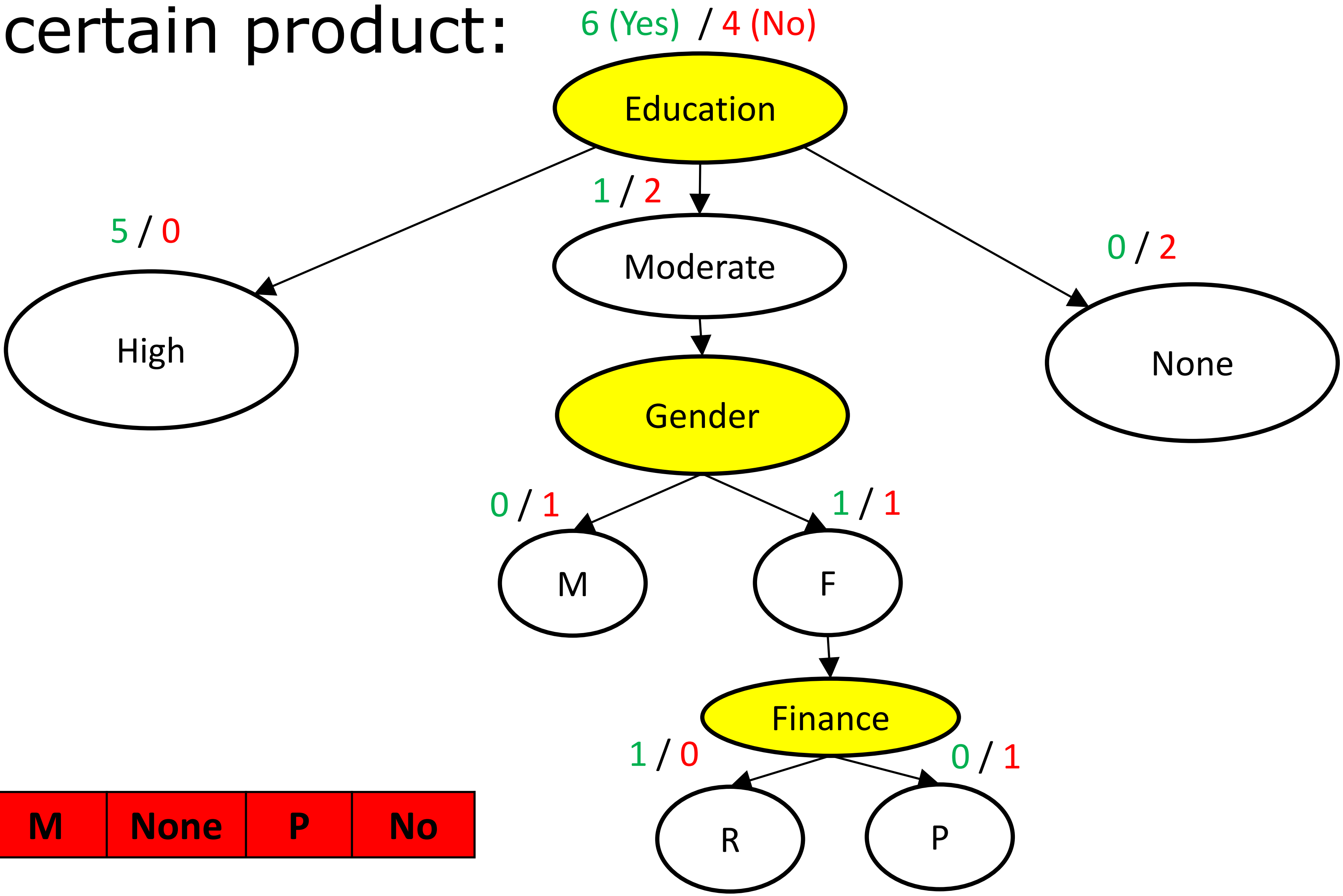
- Training examples on customers' interest in certain product:



11	**M**	**None**	**P**	**??**

Decision Trees

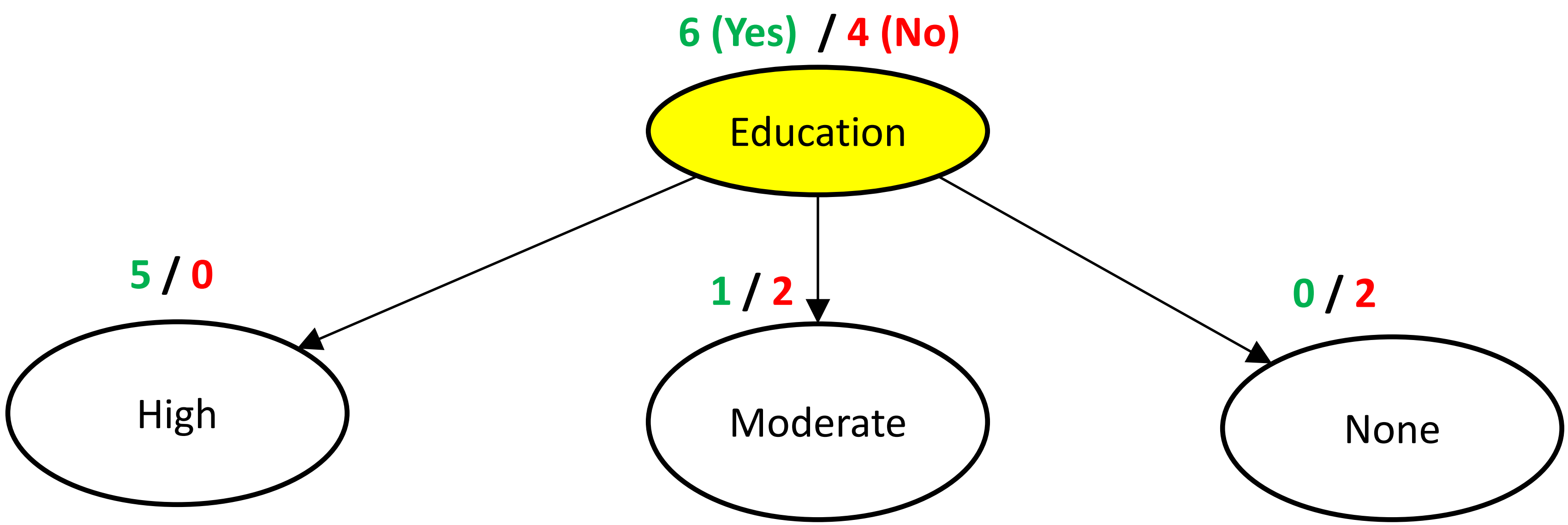
- Training examples on customers' interest in certain product:



11	**M**	**None**	**P**	**No**

Decision Trees

- Additionally, we may **prune** decision trees:
  - Use likelihoods to decide



Building a Decision Tree

Split(node, {training examples of that node}):

1. X <– Get best attribute to split examples
2. For each value of X create a child node
3. Split examples to each child node
4. For each child node:
   i. If subset of examples is pure → stop
   ii. Else: Split(child node, {subset of examples})

ID3 Algorithm

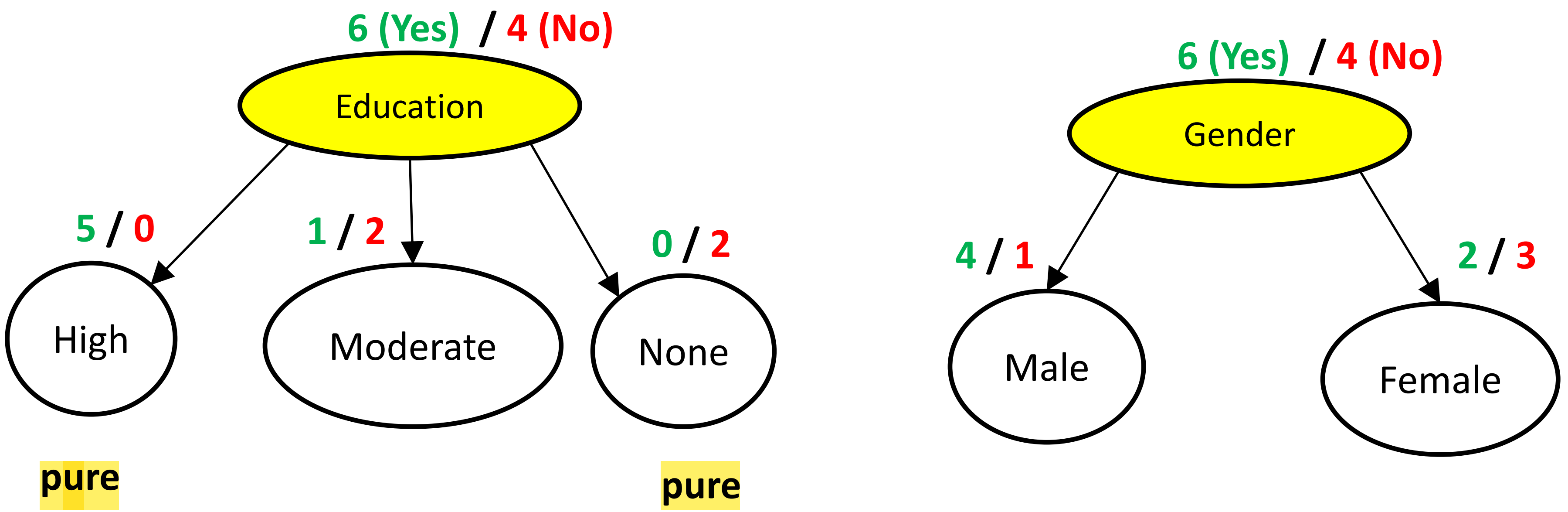
recursive!

Building a Decision Tree

- ID3
- C4.5 algorithm (improvement of ID3)
- CART → (Classification And Regression Tree)
- CHAID → (Chi-square automatic interaction detection)
- MARS → (multivariate adaptive regression splines)

Selection of Best Attribute

- Which attribute to choose for splitting?
  - Goal: get heavily biased subsets (i.e., decrease uncertainty)
  - measure purity of split (symmetric)



Entropy (w.r.t given example)

$$\mathbf{H(S)} = -\boldsymbol{p_{yes}log_2(p_{yes})} - \boldsymbol{p_{no}log_2(p_{no})}$$

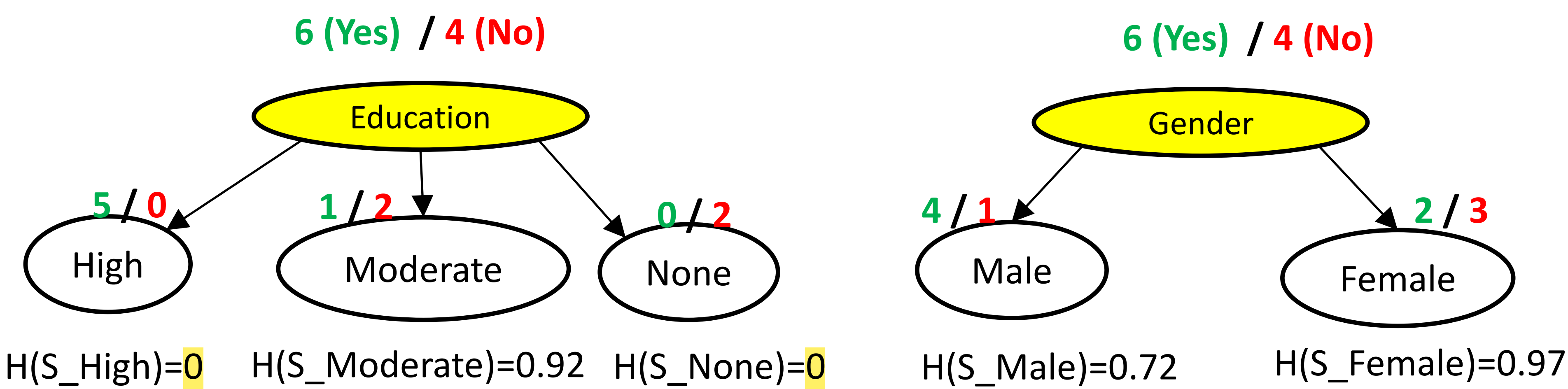
- $\mathbf{H(S)}$: entropy of example subsets $\boldsymbol{S}$
- $\boldsymbol{p_{yes}}$: % of yes examples within subset $\boldsymbol{S}$
- $\boldsymbol{p_{no}}$: % of no examples within subset $\boldsymbol{S}$

- Hints:
  - $p_{yes} = 1$ or $p_{no} = 1$ → $\mathrm{H(S)} = 0$
  - $p_{yes} = 0.5$ → $\mathrm{H(S)} = 1$

Entropy (w.r.t given example)

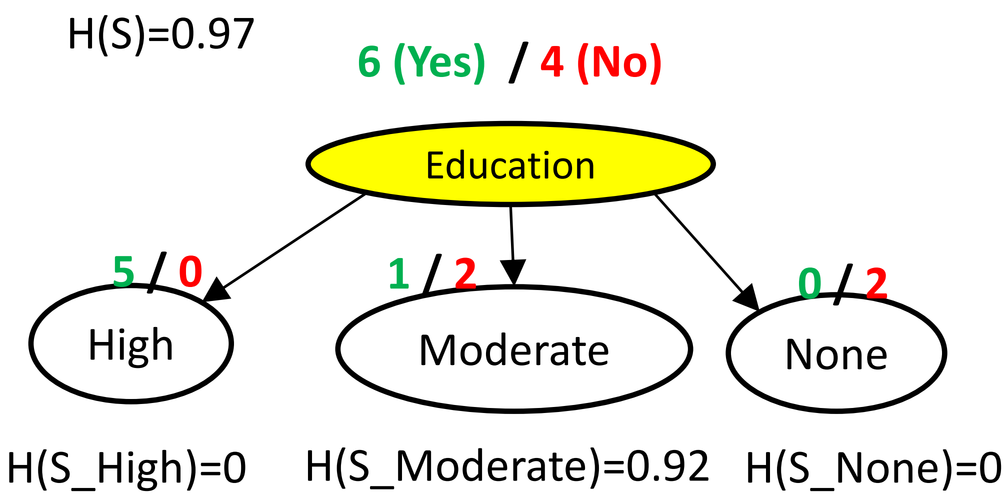
$$\mathbf{H(S)} = -p_{yes}log_2(p_{yes}) - p_n log_2(p_{no})$$

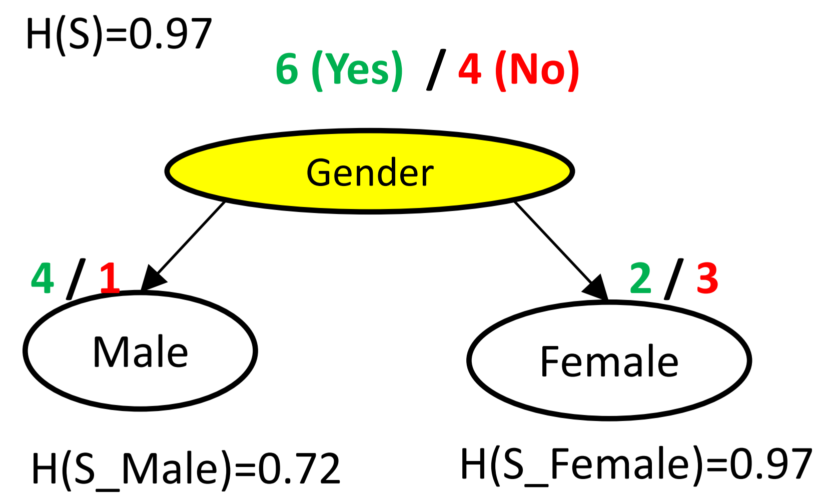
- $\mathbf{H(S)}$: entropy of example subsets S
- p_{yes}: % of yes examples within subset S
- p_{no}: % of no examples within subset S



Information Gain

$$Gain(S,X) = H(S) - \underbrace{\sum_{V \in Values(X)} \frac{|S_V|}{|S|} H(S_V)}_{\text{weighted sum of entropies}}$$





Better!

$$Gain(S, Education) = 0.97 - \frac{5}{10} * 0 - \frac{3}{10} * 0.92 - \frac{2}{10} * 0 = 0.694$$

$$Gain(S, Gender) = 0.97 - \frac{5}{10} * 0.72 - \frac{5}{10} * 0.97 = 0.125$$

Overfitting

- Decision trees can split until all training examples are correctly classified
  - all leaf nodes are pure
  - Some leaf nodes can have just one example, i.e., singletons
  - will not generalize on new data

Avoid Overfitting

- Stop splitting when not statistically significant

- Post-prune based on validation set

Better way!

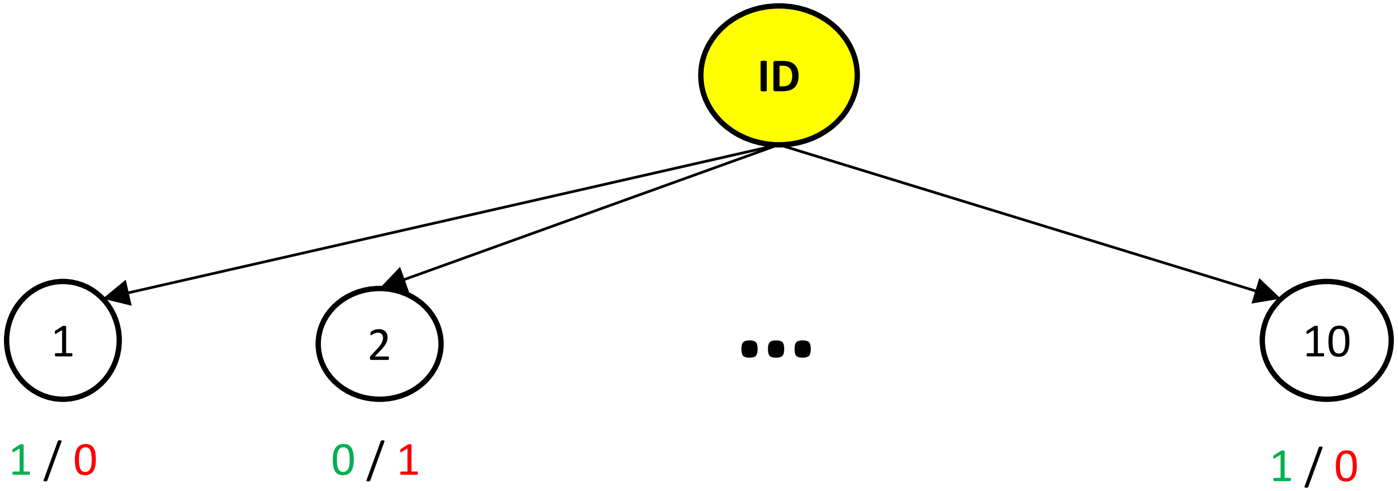
Subtree replacement pruning:

1. For each node (ignoring leaf nodes):
   i. Consider removing that node and all its children
   ii. Measure performance on validation set
2. Remove node that leads to best improvement
3. Repeat until further removals are harmful

Greedy approach, but not optimal → optimality here is intractable!

Problem with Information Gain

- What if we split on customer ID?
  - All subsets are pure → good or bad?



Highest information gain!

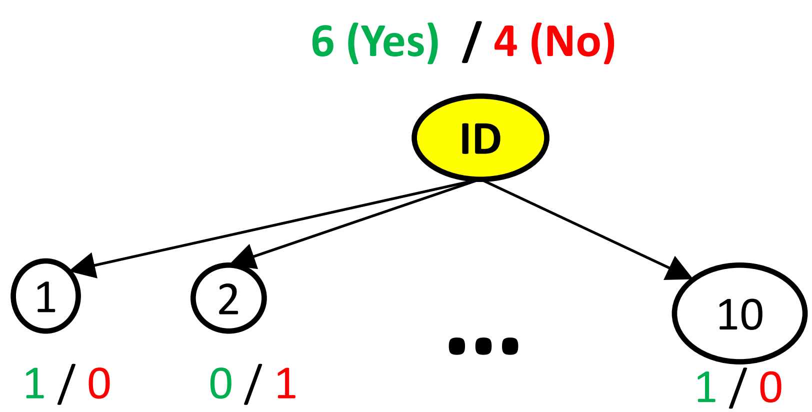
But, what about new customer #11 ?

How to avoid the selection of such attribute?

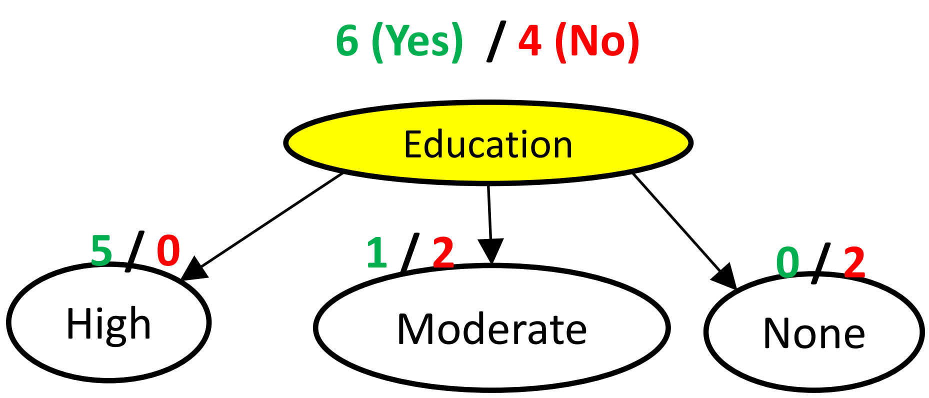
Gain Ratio

$$SplitEntropy(S,X) = -\sum_{V\in Values(X)} \frac{|S_V|}{|S|} log_2\left(\frac{|S_V|}{|S|}\right)$$

Quantifies how tiny the subsets obtained from splitting on attribute X !



$SplitEntropy(S, ID) = 3.32$



$SplitEntropy(S, Education) = 1.49$

Gain Ratio

$$\boldsymbol{SplitEntropy(S,X) = -\sum_{V \in Values(X)} \frac{|S_V|}{|S|} log_2\left(\frac{|S_V|}{|S|}\right)}$$

Quantifies how tiny the subsets obtained from splitting on attribute X !

$$\boxed{\boldsymbol{GainRatio(S,X) = \frac{Gain(S,X)}{SplitEntropy(S,X)}}}$$

Penalizes attributes with many values!

Decision Trees

- Interpretable, i.e., not black box

- Get rules from the tree
  - Can get logic formula in DNF (disjunctive normal form)

Continuous Attributes

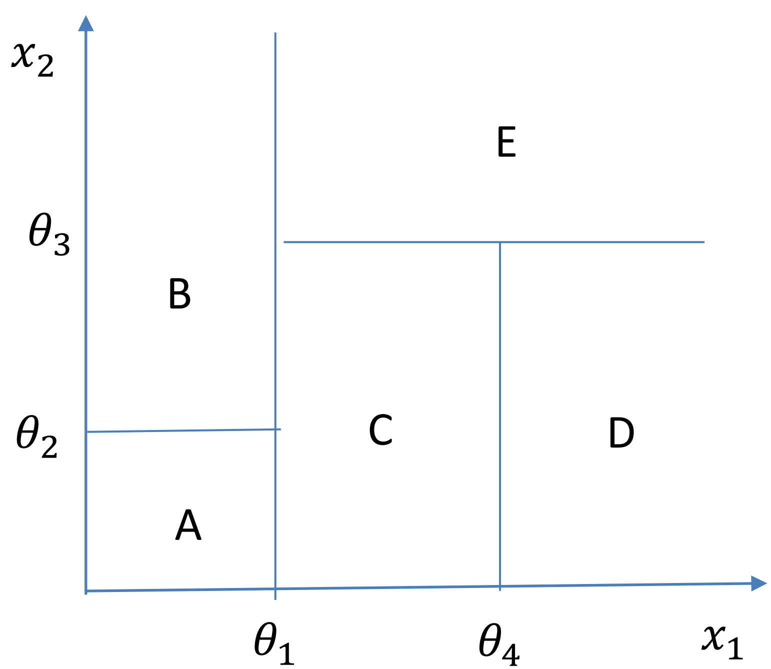
- We build trees using attributes with real values

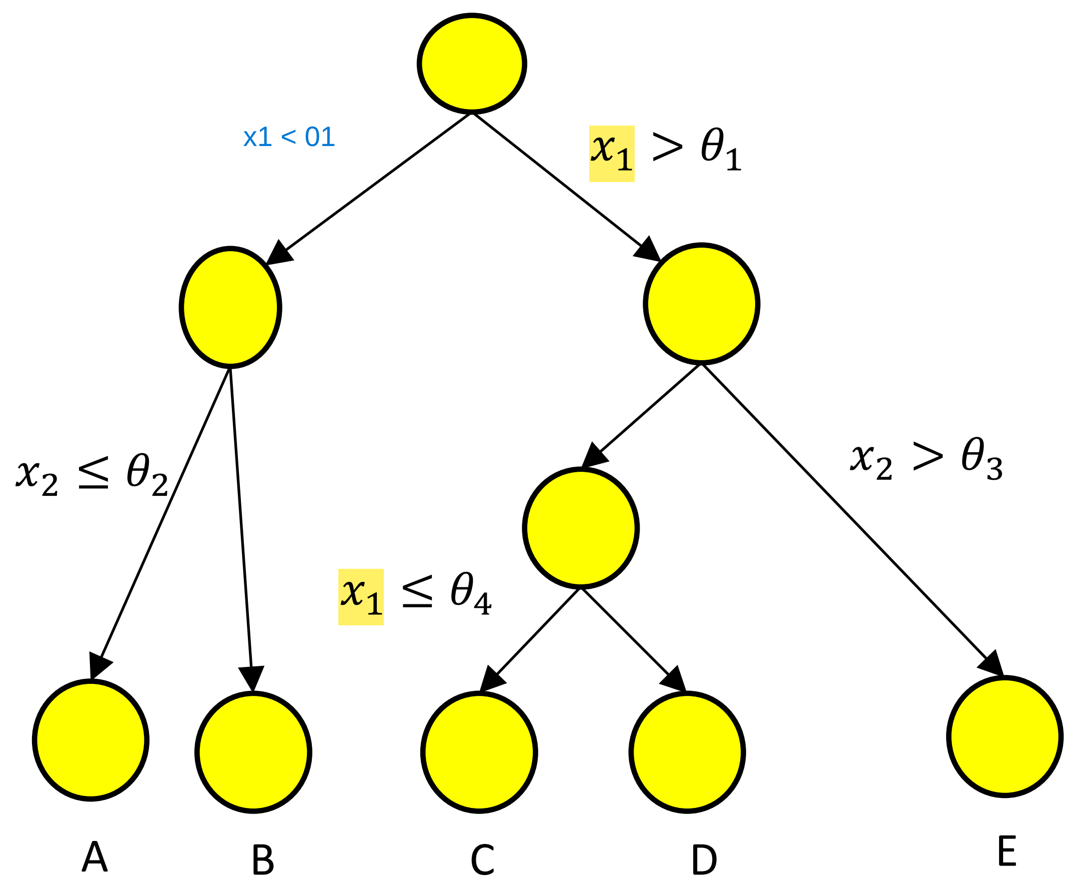
- Same as discrete attributes

- Continuous attributes can be repeated, unlike discrete attributes

Continuous Attributes

- Real values of attributes are sorted and average of each two adjacent examples is a threshold to be considered





Multiclass Classification & Regression

- Entropy in multi-class classification:

$$\mathbf{H(S)} = -\sum_{i} p_i \log_2(p_i)$$

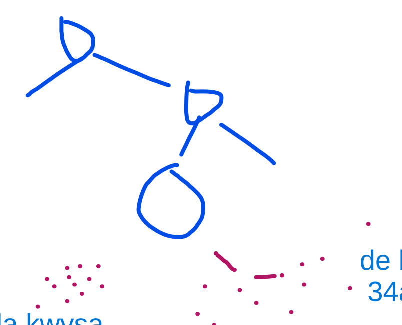
- Regression:

tb msh hwa keda dyman hyb2a sabet?

  – Predicted output → avg. of training examples in subset (or linear regression at leaves)
  – Minimize variance in subsets (instead of maximize gain)

how can we use trees with regression?

we classify the training examples based on multiple thresholds, then when evaluating new test example, it falls in certain region.
we take the average of this region, and produce the output as this average.



de kda msh kwysa,
34an el variance hena kber

de keda kwysa
34an el variance
olyl

Pros & Cons

- Pros:
  - Interpretable
  - Easily handles irrelevant attributes (Gain = 0)
  - ~~Can handle missing data (out of scope)~~
  - Very compact (#num of nodes << #num of attributes after pruning)
  - Very fast at testing time: O(depth)

- Cons:
  - ID3 greedy (may not find best tree)
  - Only axis-aligned splits of data (continuous data)

Acknowledgement

- These slides have been designed relying on materials of Victor Lavrenko and Kilian Weinberger