Decision Trees

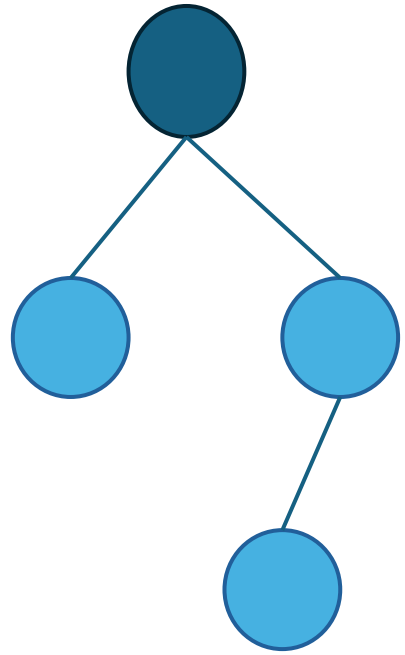
10 February 2026

Alex Lyman

Trees

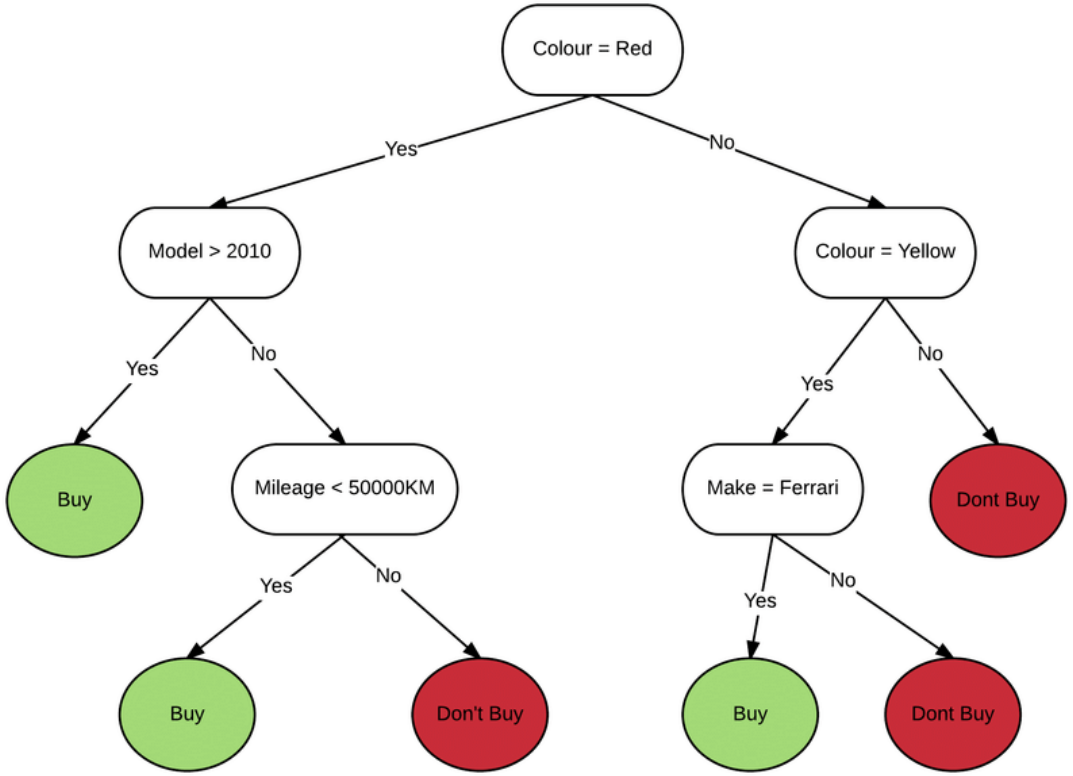
Trees (Data Structure)

- Trees are an important data structure in CS. Binary search trees, etc.
- Tree is a non-cyclical data structure made of nodes and edges.
  - Node: A single point or bubble in the tree.
  - Edge: The line connecting two nodes.
  - Root Node: The very top node of the tree.
  - Leaf: A node with no children (no outgoing edges).
  - Depth: The number of edges from the Root to a specific node.
- Trees are ‘upside down’ with the root at the top



Decision Tree

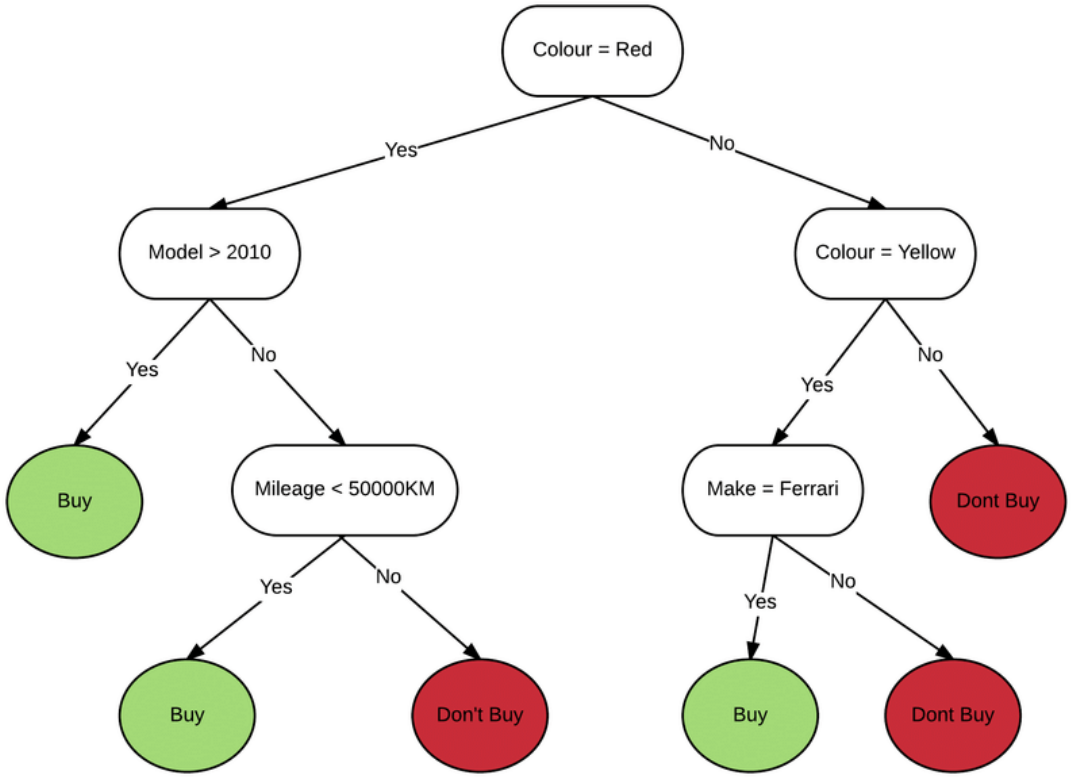
- What if we could make a classifier using a tree to model the decision structure?
  - (Obviously we can or else class would be really short)
- Decision trees are very easy to interpret, very intuitive.
- Each node represents asking a question to classify.
- Decision Trees *love* overfitting.



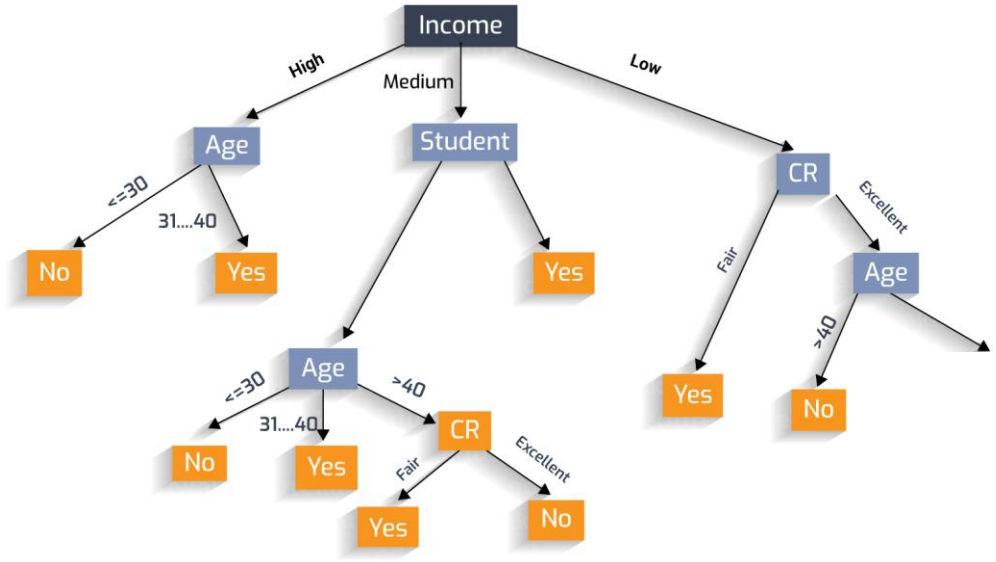
Decision Tree

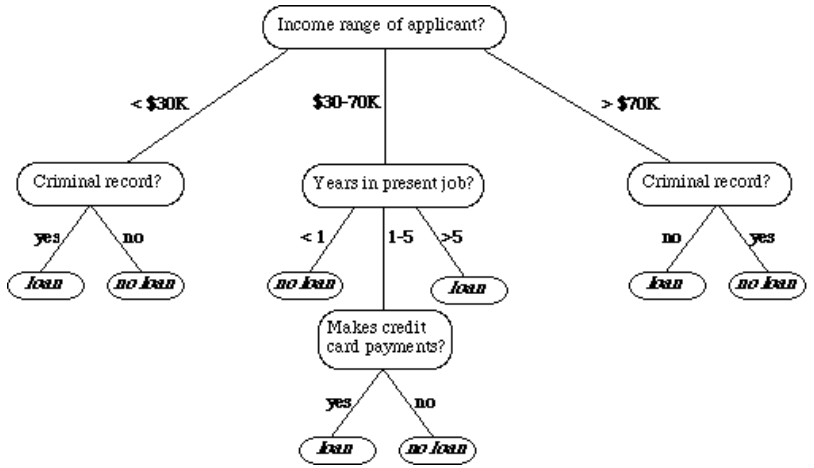
- Decision tree is a classifier in the form of a tree.
  - Decision node: specifies a test on a **single attribute**
  - Leaf node: indicates the value of the target attribute
  - Edge: split of one attribute
  - Path: collection of choices from root to leaf

- Decision trees classify instances or examples by starting at the root of the tree and moving through it until hitting a leaf node.



Sample Decision Trees

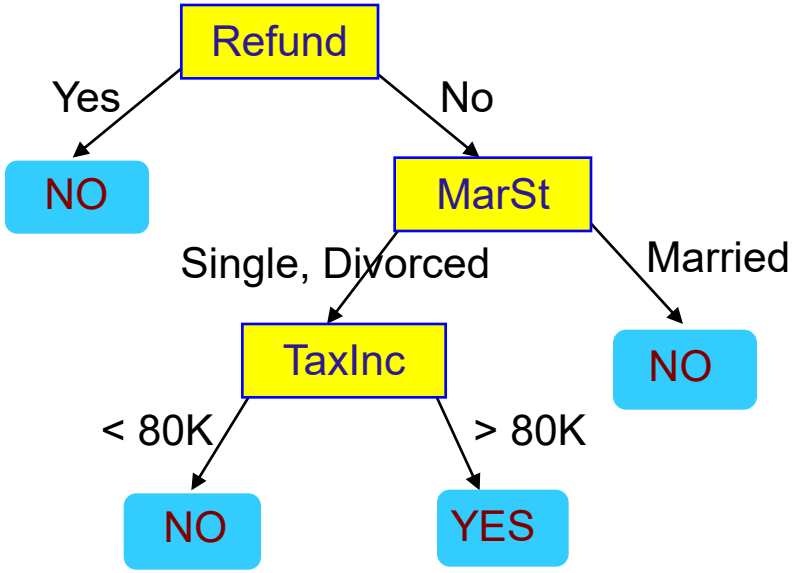




Example of a Decision Tree

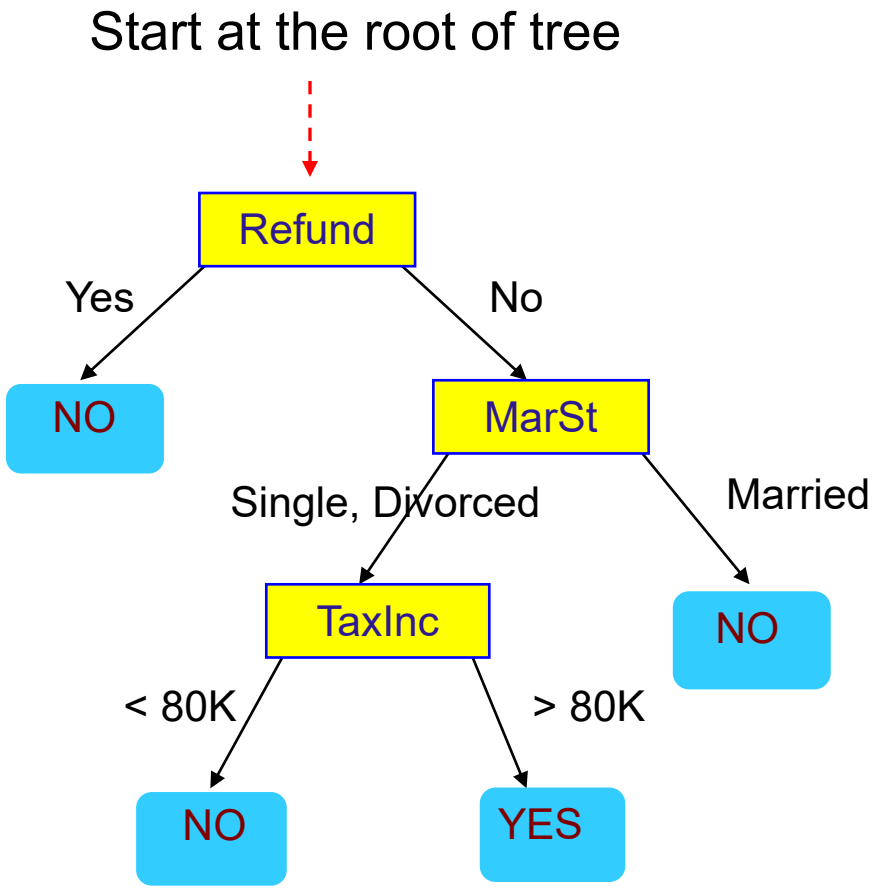
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data



Model: Decision Tree

Apply Model to Test Data



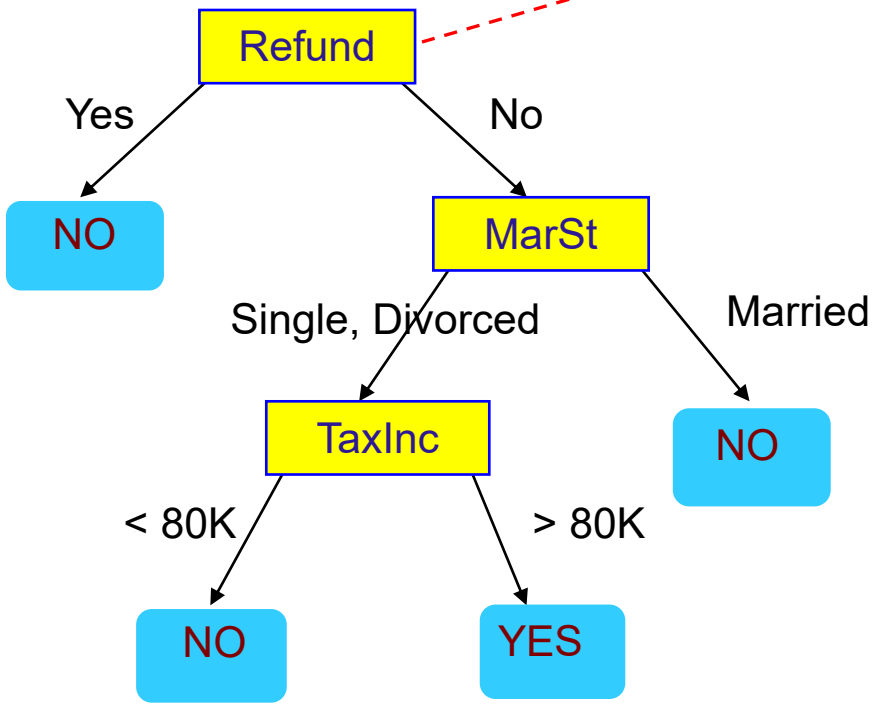
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

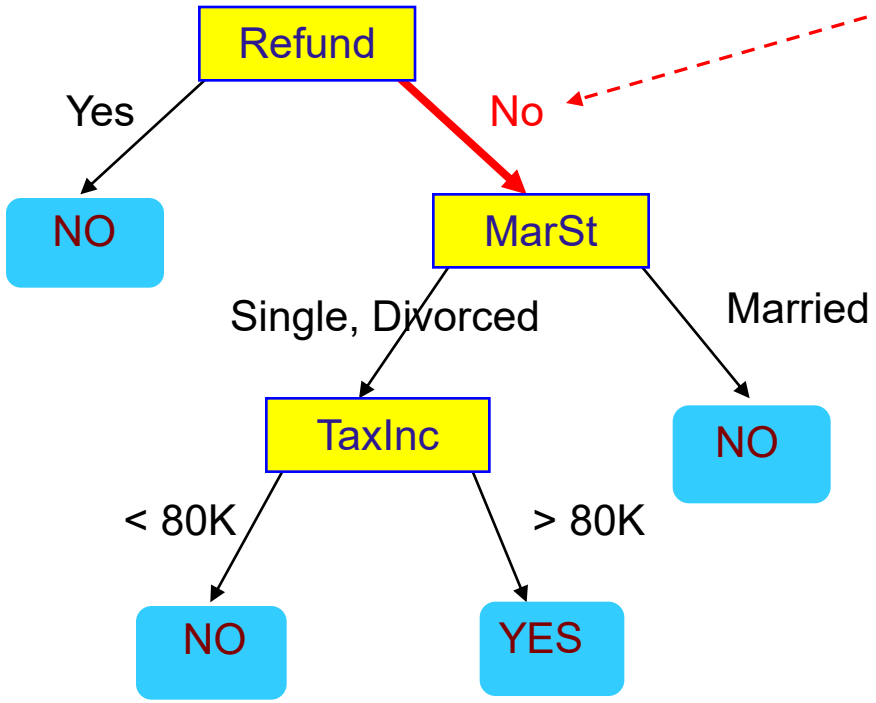
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

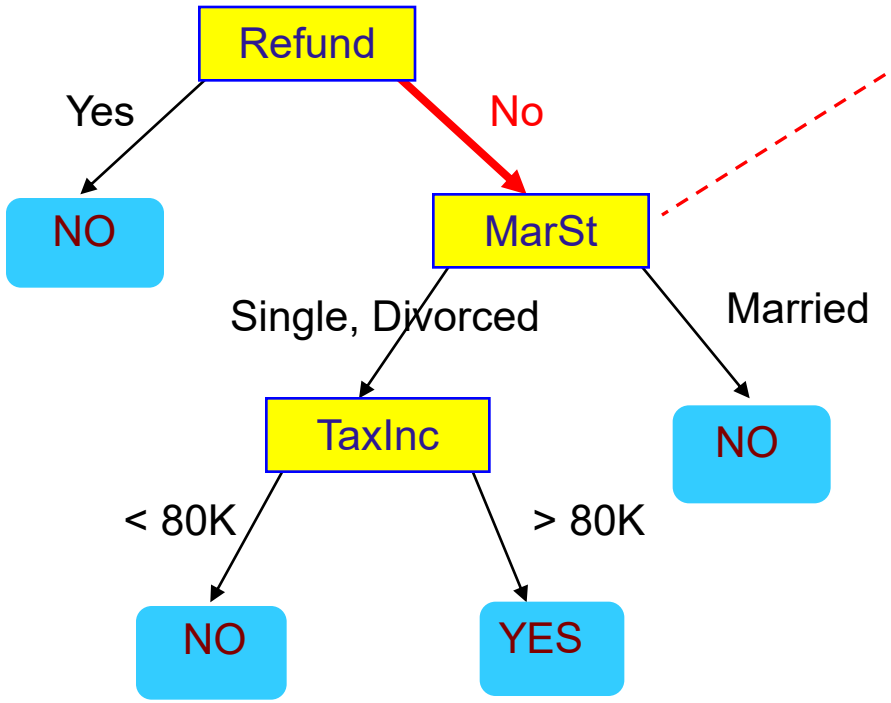
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

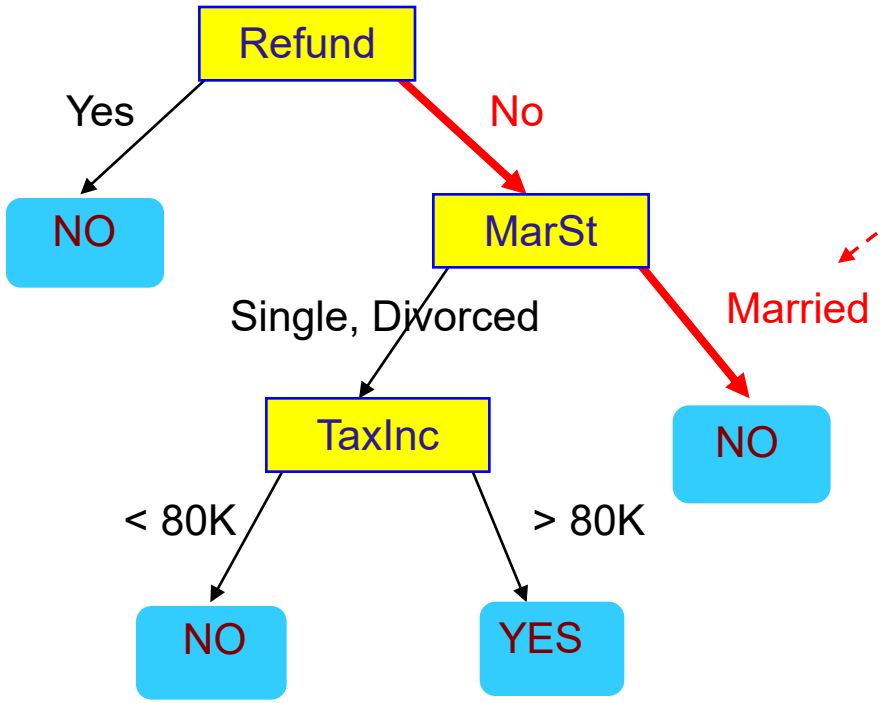
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

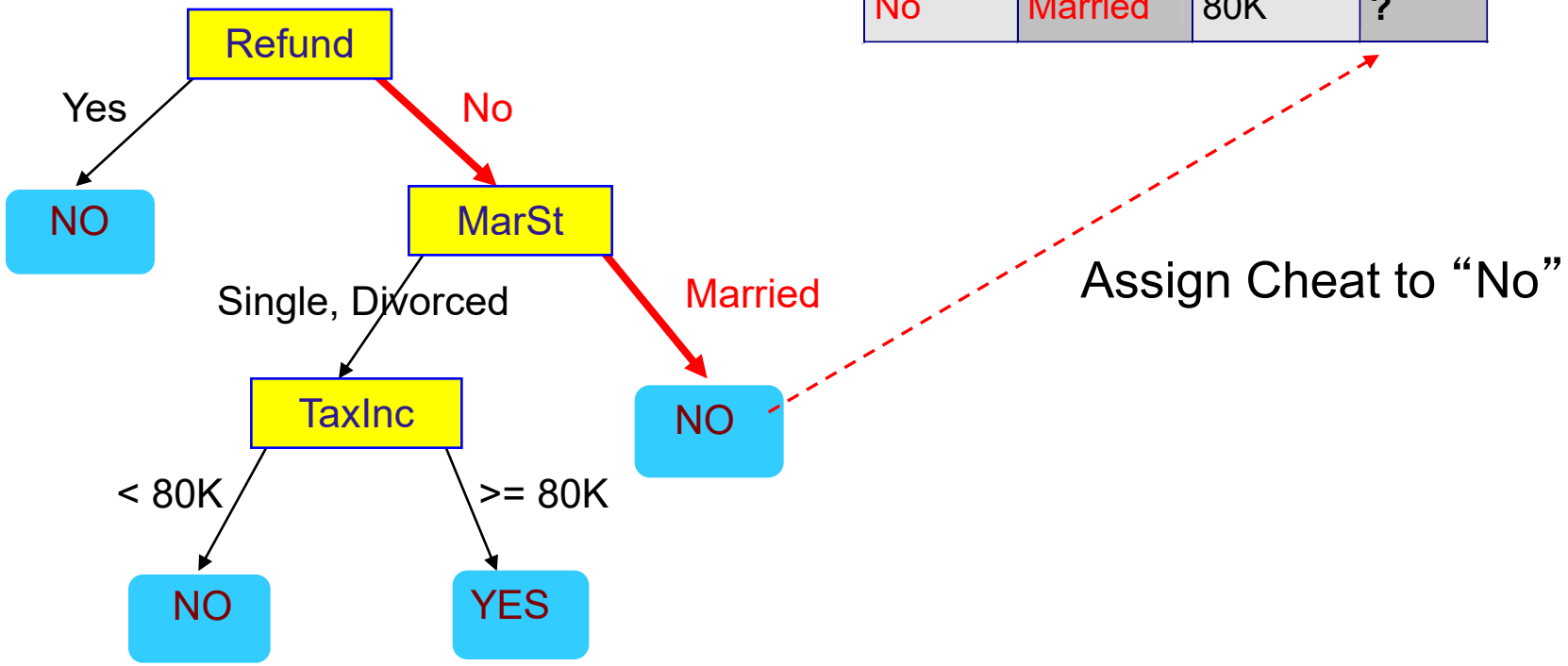
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

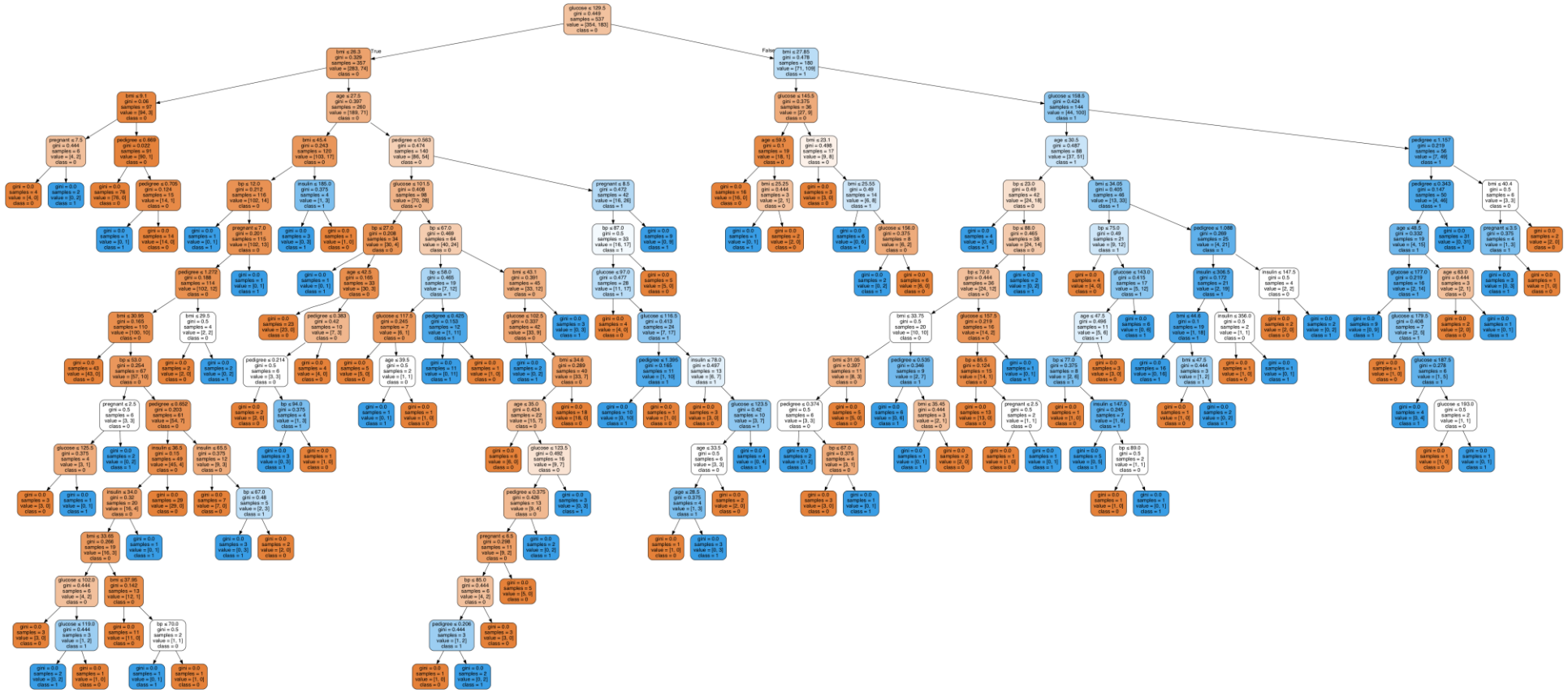
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Decision Trees

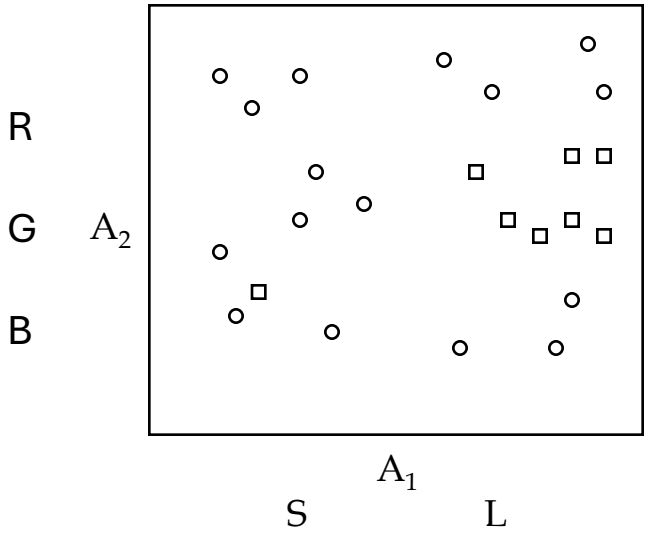
- Can grow very large and difficult to understand
- Typically they can overfit the training data



Constructing a Decision Tree From A Plot

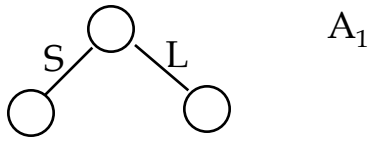
Decision Tree Learning

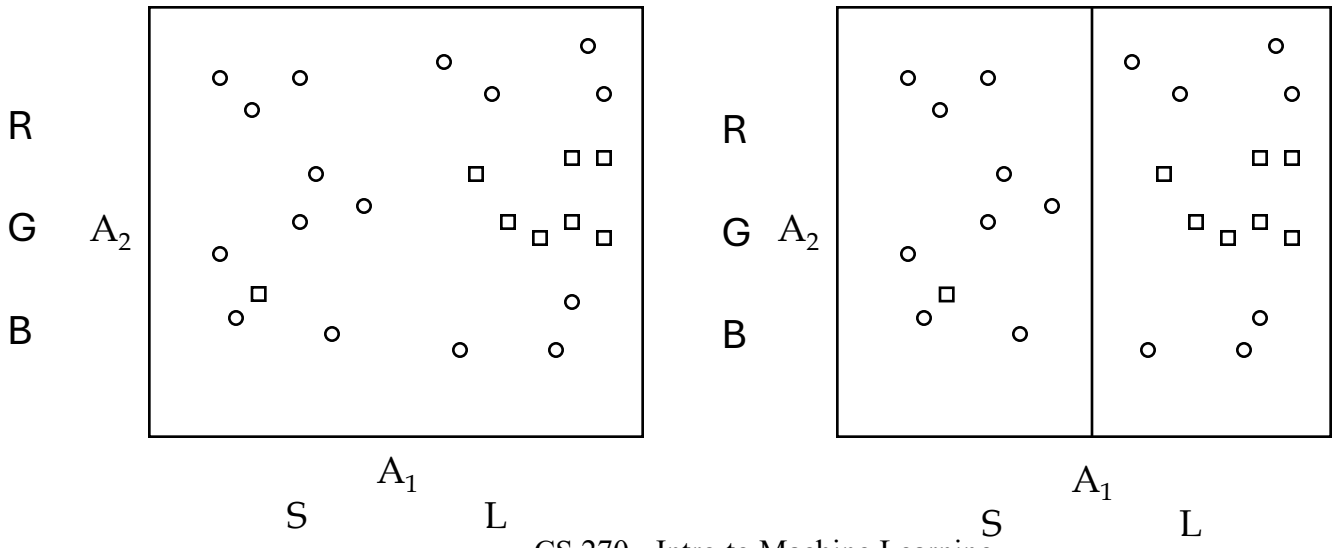
- Assume A_1 is nominal binary feature (Size: S/L)
- Assume A_2 is nominal 3 value feature (Color: R/G/B)
- Target is square or circle
- Create leaf nodes by partitioning up space.
- A goal is to get "pure" leaf nodes.
- What feature would you split on?



Decision Tree Learning

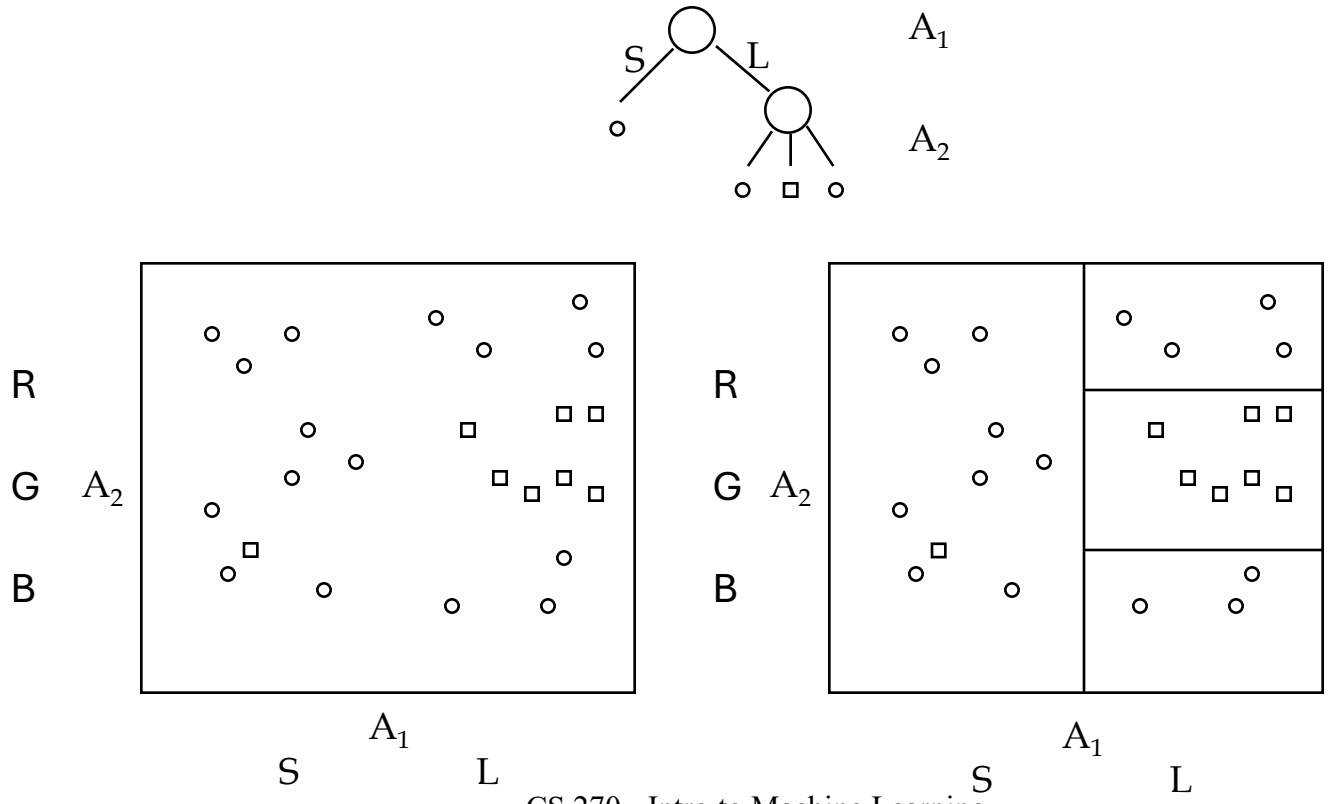
- Assume A_1 is nominal binary feature (Size: S/L)
- Assume A_2 is nominal 3 value feature (Color: R/G/B)
- Next step for left child?
- Next step for right child?





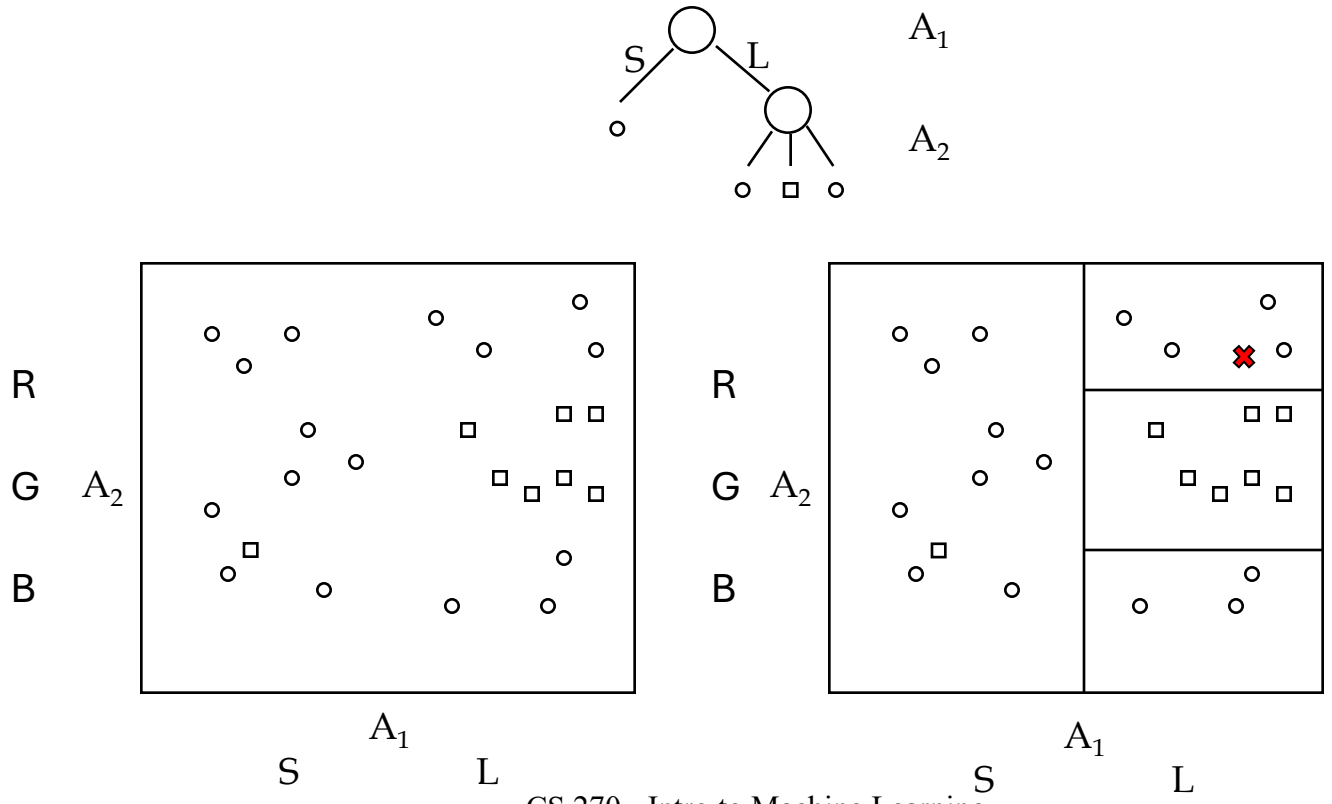
Decision Tree Learning

- Assume A_1 is nominal binary feature (Size: S/L)
- Assume A_2 is nominal 3 value feature (Color: R/G/B)
- Decision surfaces are **axis-aligned Hyper-Rectangles**
- **Label leaf nodes with their majority class**



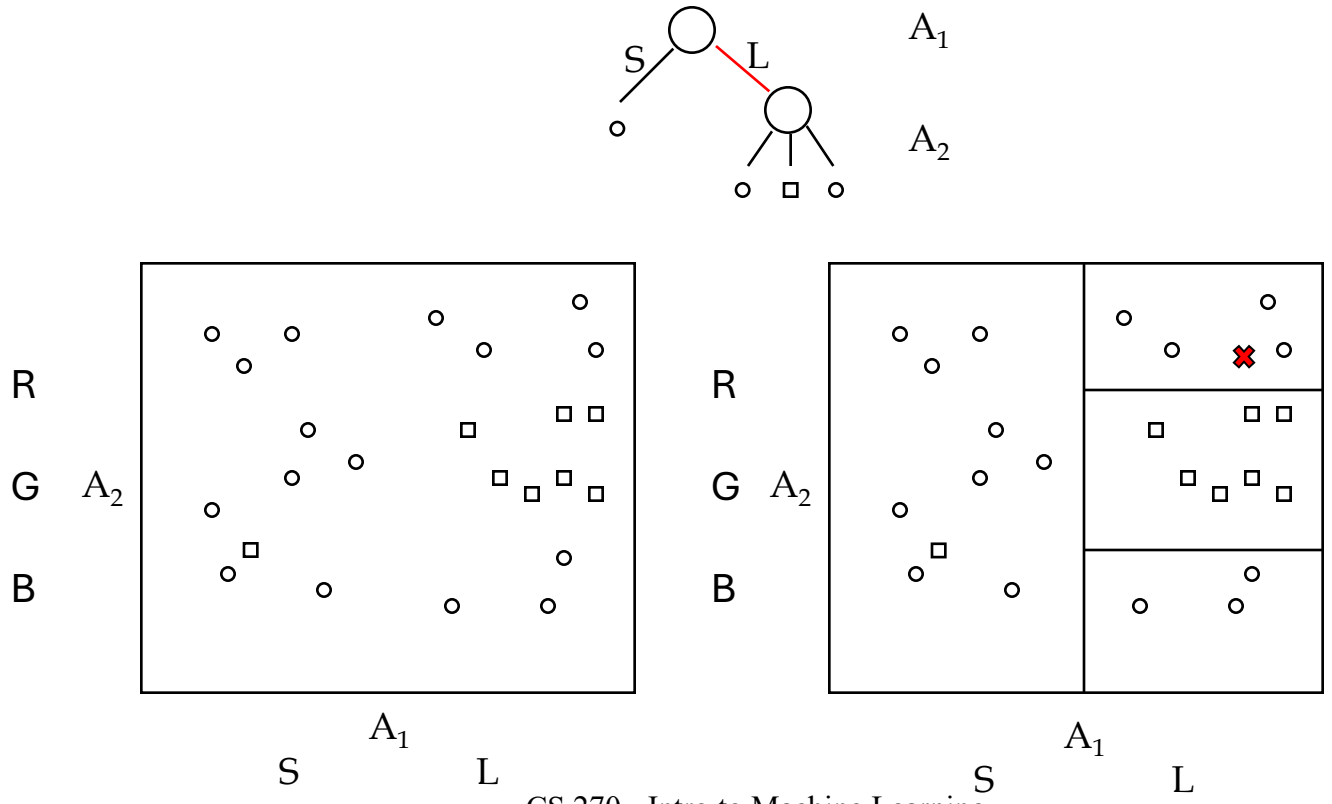
Decision Tree Learning

- Assume A_1 is nominal binary feature (Size: S/L)
- Assume A_2 is nominal 3 value feature (Color: R/G/B)
- Decision surfaces are **axis-aligned Hyper-Rectangles**
- **Label leaf nodes with their majority class**



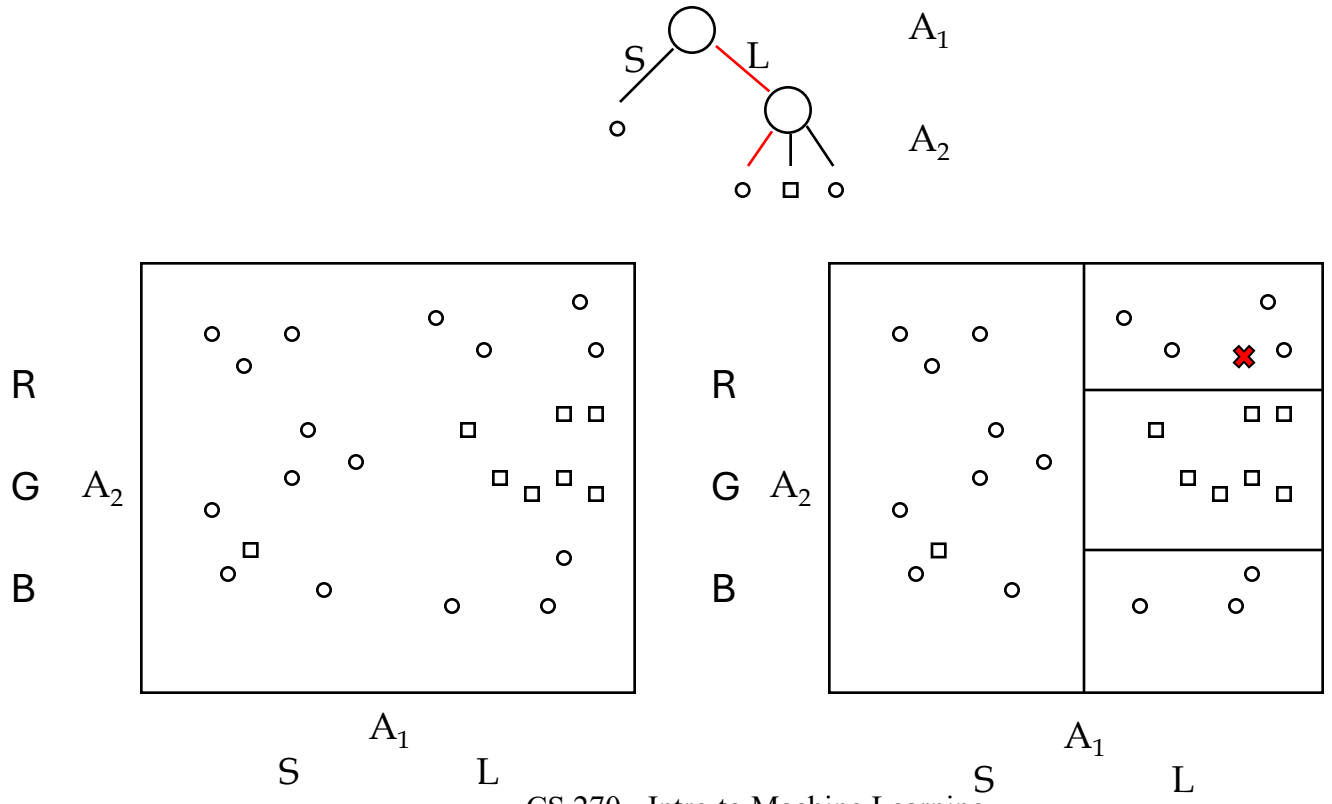
Decision Tree Learning

- Assume A_1 is nominal binary feature (Size: S/L)
- Assume A_2 is nominal 3 value feature (Color: R/G/B)
- Decision surfaces are **axis-aligned Hyper-Rectangles**
- **Label leaf nodes with their majority class**



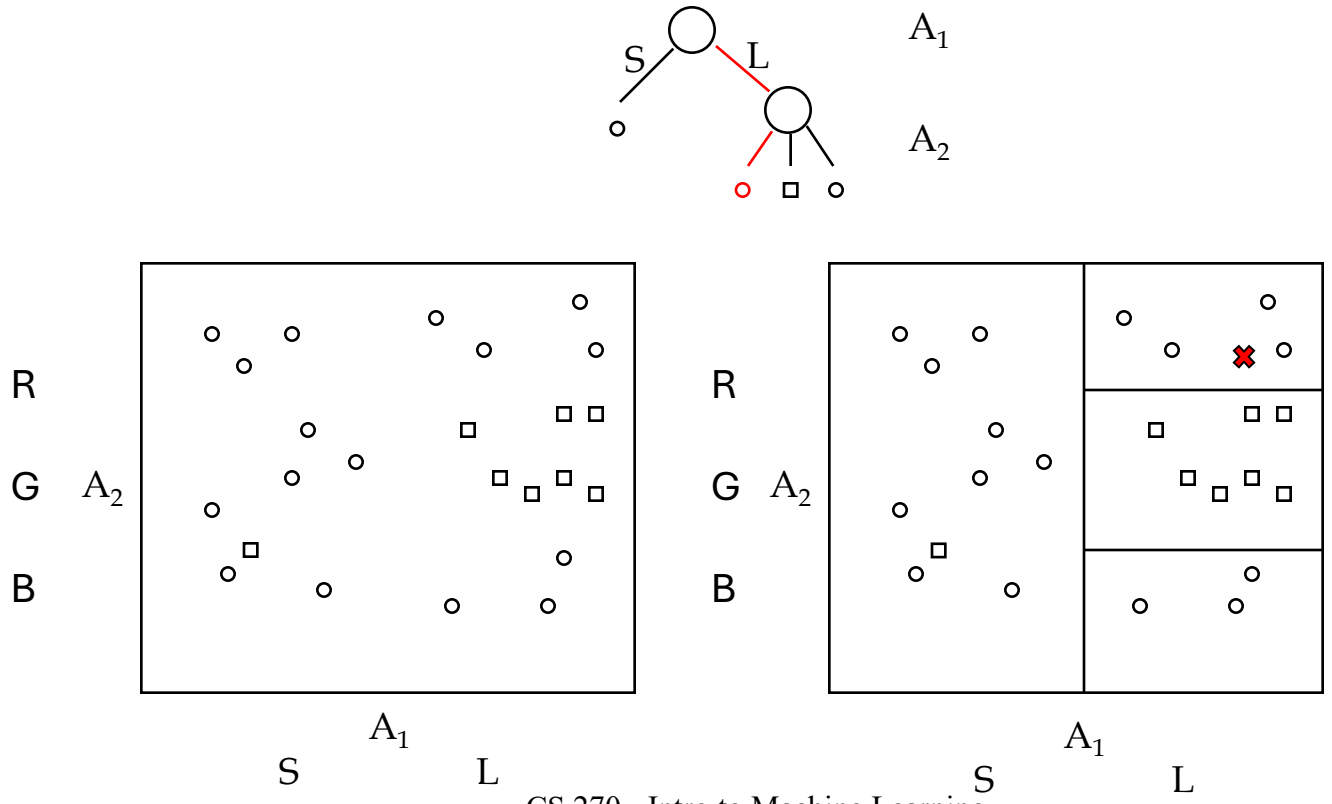
Decision Tree Learning

- Assume A_1 is nominal binary feature (Size: S/L)
- Assume A_2 is nominal 3 value feature (Color: R/G/B)
- Decision surfaces are **axis-aligned Hyper-Rectangles**
- **Label leaf nodes with their majority class**



Decision Tree Learning

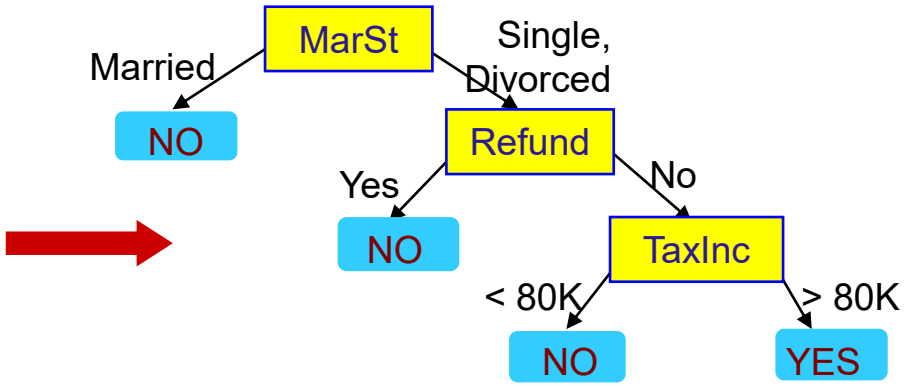
- Assume A_1 is nominal binary feature (Size: S/L)
- Assume A_2 is nominal 3 value feature (Color: R/G/B)
- Decision surfaces are **axis-aligned Hyper-Rectangles**
- **Label leaf nodes with their majority class**



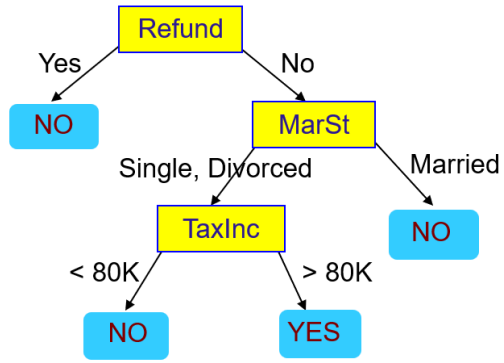
Non-Uniqueness

- Decision trees are not unique:
  - Given a set of training instances *T*, there generally exists a number of decision trees that are consistent with (or fit) *T*
  - In other words, you can make multiple different decision trees with the same dataset.

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



OR



Decision Tree Algorithms

Decision Tree Algorithms

- J Ross Quinlan – Australia, ML researcher
  - **ID3** (Iterative Dichotimiser 3) – 1986
  - C4.5 – (Version 4.5 written in C) 1993, Handles real valued inputs
  - **C5.0** – More efficient implementation
- Leo Breiman - UC Berkeley
  - **CART** (Classification and Regression Trees) – 1984
    - This is the decision tree approach currently supported in Sklearn
  - **Random Forests - 2001**
- Independently discovered

Constructing Decision Trees

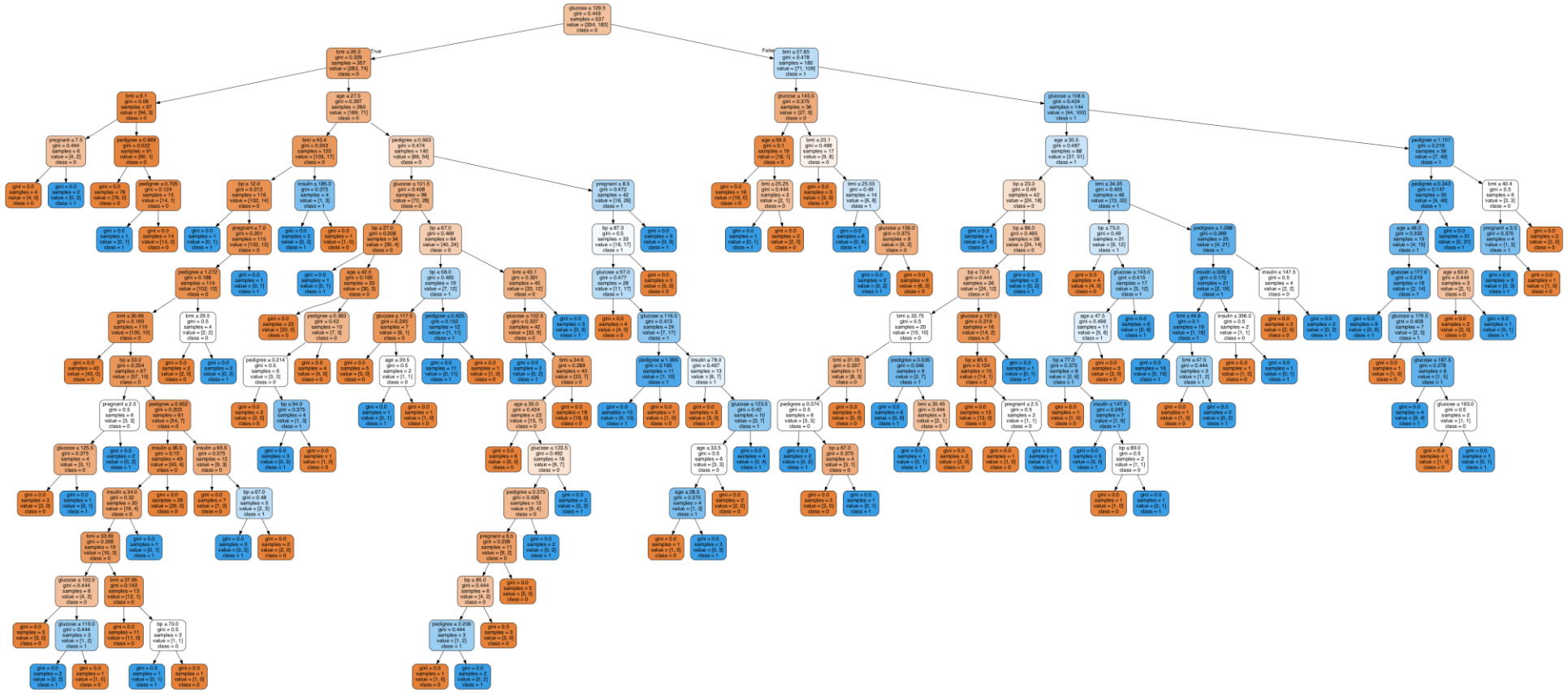
- Exponentially many decision trees can be constructed from a given set of attributes

- Finding the most accurate tree is NP-hard

- **In practice: greedy algorithms**
  - Grow a decision tree by making a series of ***locally optimum decisions on which attributes to use*** for partitioning the data

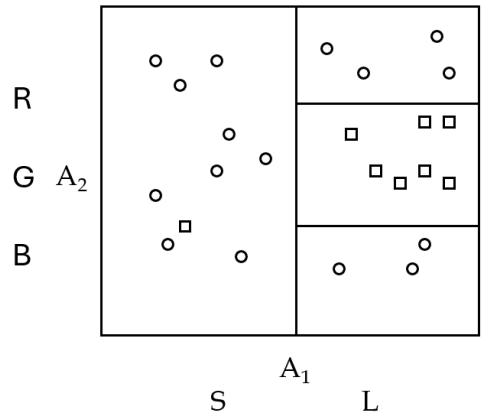
Decision Trees

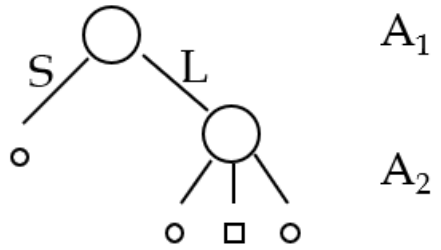
- Can grow very large and difficult to understand



Decision Tree Algorithms

- Think of creating Decision Trees like playing a game of 20 questions.
- You want to ask the ‘best question’ at each step. (If each question cut the group roughly in half, we’d be doing great, right?)
- ID3 is GREEDY
  - (cares about taking the next best choice)
- At each step, “Which question would create the cleanest, purest groups”
- What is group purity?
- How to measure group purity?





Decision Tree Algorithms

- Not all attributes are created equal
- Each attribute of an instance may be thought of as contributing a certain amount of information to its classification
  - Think 20-Questions:
    - What are good questions?
    - Ones whose answers maximize information gained
  - For example, determine shape of an object:
    - Num. sides contributes a certain amount of information
    - Color contributes a different amount of information
  - Guess Who? Example:
    - "Is the person Frank?" – bad question since it gives us very little information
    - "Is the person a girl?" - good question, eliminates half the field

- ID3 measures information gained by making each attribute the root of the current subtree, and subsequently chooses the attribute that produces the **greatest information gain.**
- *Draw subtree on board here*
- We'll quantify information gain later.

ID3's Question

- Remember Non-Uniqueness
- Given a training set, which of all of the decision trees consistent with that training set should we pick?

- More precisely:
  - Given a training set, which of all of the decision trees consistent with that training set has the greatest likelihood of correctly classifying unseen instances of the population?

Decision Tree Learning: ID3 Algorithm

Function ID3(*Training-set*, *Attributes*)

- If all elements in *Training-set* are in same class, then return leaf node labeled with that class
- Else if *Attributes* is empty, then return leaf node labeled with majority class in *Training-set*
- Else if *Training-Set* is empty, then return leaf node labeled with default majority class
- Else
  - Select and remove *A* from *Attributes*
  - Make *A* the root of the current tree
  - For each value *V* of *A*
    - Create a branch of the current tree labeled by *V*
    - *Partition_V* ← Elements of *Training-set* with value *V* for *A*
    - ID3(*Partition_V*, *Attributes*)
    - Attach result to branch *V*

Illustrative Training Set

Risk Assessment for Loan Applications

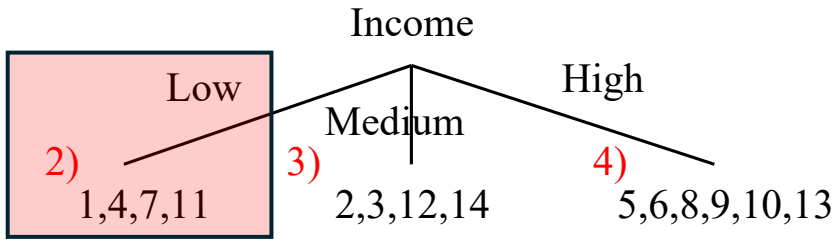
Client #	Credit History	Debt Level	Collateral	Income Level	RISK LEVEL
1	Bad	High	None	Low	**HIGH**
2	Unknown	High	None	Medium	**HIGH**
3	Unknown	Low	None	Medium	**MODERATE**
4	Unknown	Low	None	Low	**HIGH**
5	Unknown	Low	None	High	**LOW**
6	Unknown	Low	Adequate	High	**LOW**
7	Bad	Low	None	Low	**HIGH**
8	Bad	Low	Adequate	High	**MODERATE**
9	Good	Low	None	High	**LOW**
10	Good	High	Adequate	High	**LOW**
11	Good	High	None	Low	**HIGH**
12	Good	High	None	Medium	**MODERATE**
13	Good	High	None	High	**LOW**
14	Bad	High	None	Medium	**HIGH**

ID3 Example

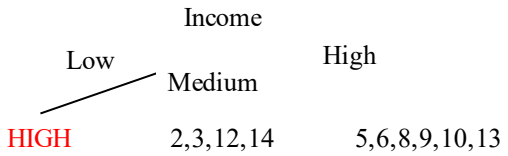
Risk Assessment for Loan Applications

Client #	Credit History	Debt Level	Collateral	Income Level	RISK LEVEL
1	Bad	High	None	Low	**HIGH**
2	Unknown	High	None	Medium	**HIGH**
3	Unknown	Low	None	Medium	**MODERATE**
4	Unknown	Low	None	Low	**HIGH**
5	Unknown	Low	None	High	**LOW**
6	Unknown	Low	Adequate	High	**LOW**
7	Bad	Low	None	Low	**HIGH**
8	Bad	Low	Adequate	High	**MODERATE**
9	Good	Low	None	High	**LOW**
10	Good	High	Adequate	High	**LOW**
11	Good	High	None	Low	**HIGH**
12	Good	High	None	Medium	**MODERATE**
13	Good	High	None	High	**LOW**
14	Bad	High	None	Medium	**HIGH**

1) Choose Income as root of tree.



2) All examples are in the same class, HIGH.
Return Leaf Node.

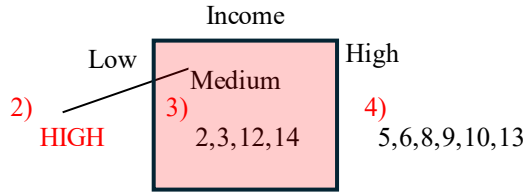


ID3 Example

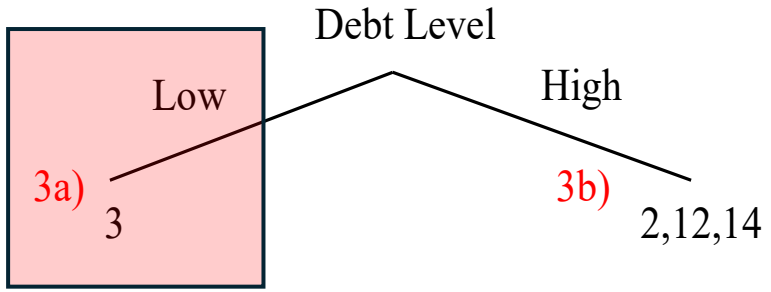
Risk Assessment for Loan Applications

Client #	Credit History	Debt Level	Collateral	Income Level	RISK LEVEL
1	Bad	High	None	Low	**HIGH**
2	Unknown	High	None	Medium	**HIGH**
3	Unknown	Low	None	Medium	**MODERATE**
4	Unknown	Low	None	Low	**HIGH**
5	Unknown	Low	None	High	**LOW**
6	Unknown	Low	Adequate	High	**LOW**
7	Bad	Low	None	Low	**HIGH**
8	Bad	Low	Adequate	High	**MODERATE**
9	Good	Low	None	High	**LOW**
10	Good	High	Adequate	High	**LOW**
11	Good	High	None	Low	**HIGH**
12	Good	High	None	Medium	**MODERATE**
13	Good	High	None	High	**LOW**
14	Bad	High	None	Medium	**HIGH**

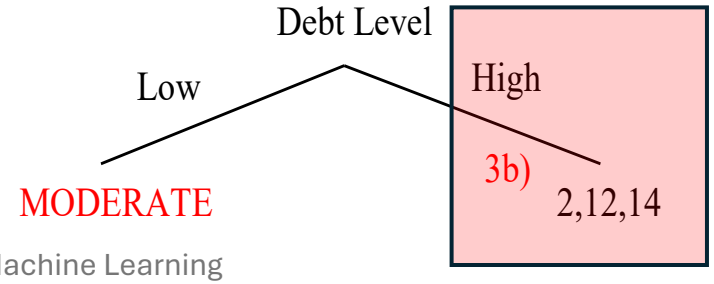
1) Choose Income as root of tree.



3) Choose Debt Level as root of subtree.



3a) All examples are in the same class, MODERATE.
Return Leaf node.

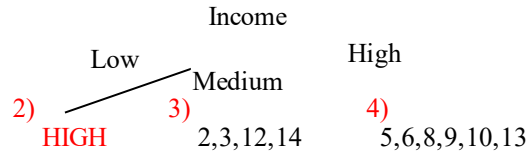


ID3 Example

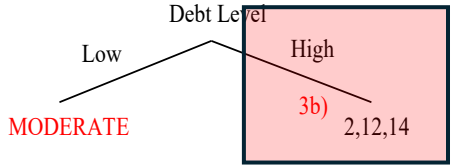
Risk Assessment for Loan Applications

Client #	Credit History	Debt Level	Collateral	Income Level	RISK LEVEL
1	Bad	High	None	Low	**HIGH**
2	Unknown	High	None	Medium	**HIGH**
3	Unknown	Low	None	Medium	**MODERATE**
4	Unknown	Low	None	Low	**HIGH**
5	Unknown	Low	None	High	**LOW**
6	Unknown	Low	Adequate	High	**LOW**
7	Bad	Low	None	Low	**HIGH**
8	Bad	Low	Adequate	High	**MODERATE**
9	Good	Low	None	High	**LOW**
10	Good	High	Adequate	High	**LOW**
11	Good	High	None	Low	**HIGH**
12	Good	High	None	Medium	**MODERATE**
13	Good	High	None	High	**LOW**
14	Bad	High	None	Medium	**HIGH**

1) Choose Income as root of tree.

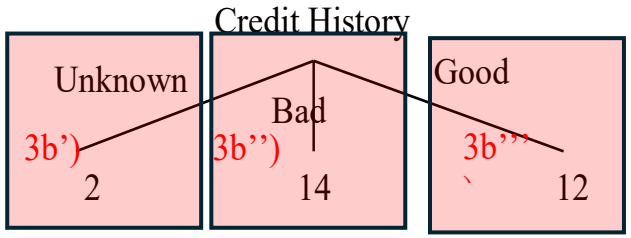


3a) All examples are in the same class, MODERATE.
Return Leaf node.

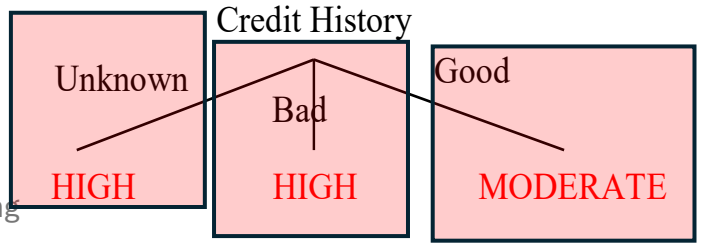


Proceed Depth First

3b) Choose Credit History as root of subtree.

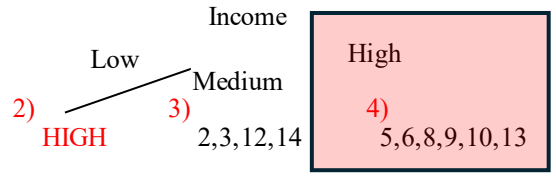


3b'-3b''') All examples are in the same class.
Return Leaf nodes.



ID3 Example

1) Choose Income as root of tree.

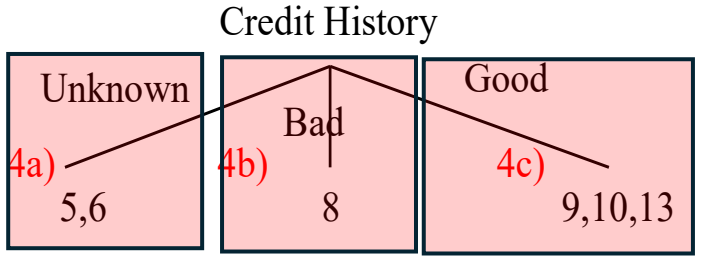


3 is done. Now pop back and handle 4

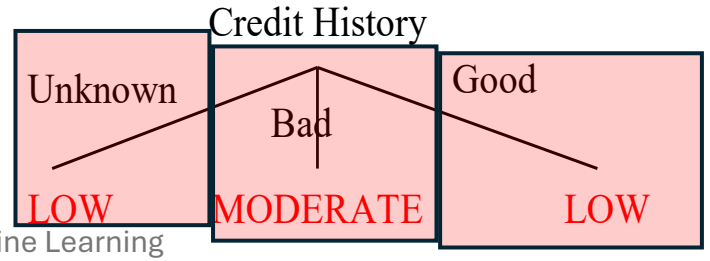
Risk Assessment for Loan Applications

Client #	Credit History	Debt Level	Collateral	Income Level	RISK LEVEL
1	Bad	High	None	Low	**HIGH**
2	Unknown	High	None	Medium	**HIGH**
3	Unknown	Low	None	Medium	**MODERATE**
4	Unknown	Low	None	Low	**HIGH**
5	Unknown	Low	None	High	**LOW**
6	Unknown	Low	Adequate	High	**LOW**
7	Bad	Low	None	Low	**HIGH**
8	Bad	Low	Adequate	High	**MODERATE**
9	Good	Low	None	High	**LOW**
10	Good	High	Adequate	High	**LOW**
11	Good	High	None	Low	**HIGH**
12	Good	High	None	Medium	**MODERATE**
13	Good	High	None	High	**LOW**
14	Bad	High	None	Medium	**HIGH**

4) Choose Credit History as root of subtree.

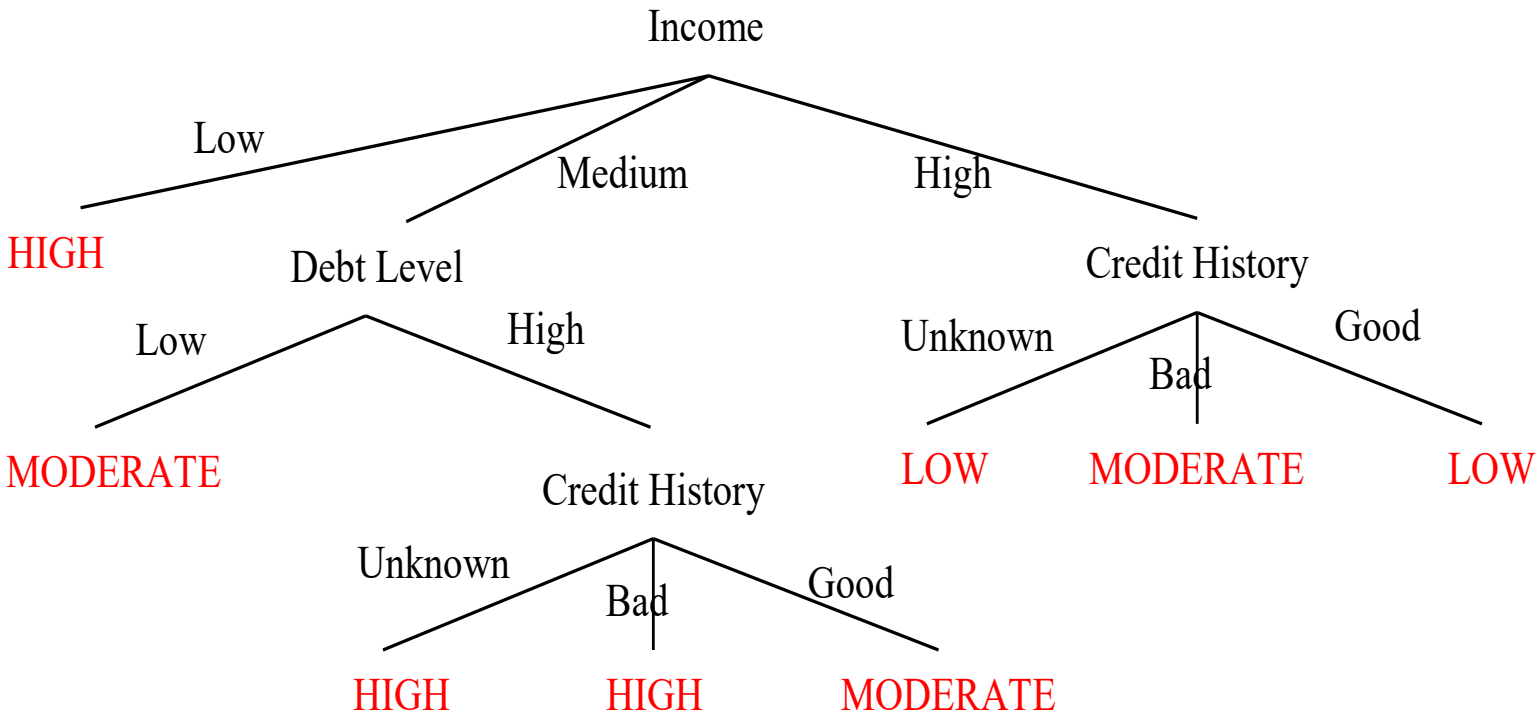


4a-4c) All examples are in the same class.
Return Leaf nodes.



ID3 Example

Attach subtrees at appropriate places.



ID3's (Approximate) Bias

- ID3 (and family) prefers simpler decision trees
- Occam's Razor Principle:
  - "It is vain to do with more what can be done with less...Entities should not be multiplied beyond necessity."
- Intuitively:
  - Always accept the simplest answer that fits the data, avoid unnecessary constraints
  - Simpler trees are more general

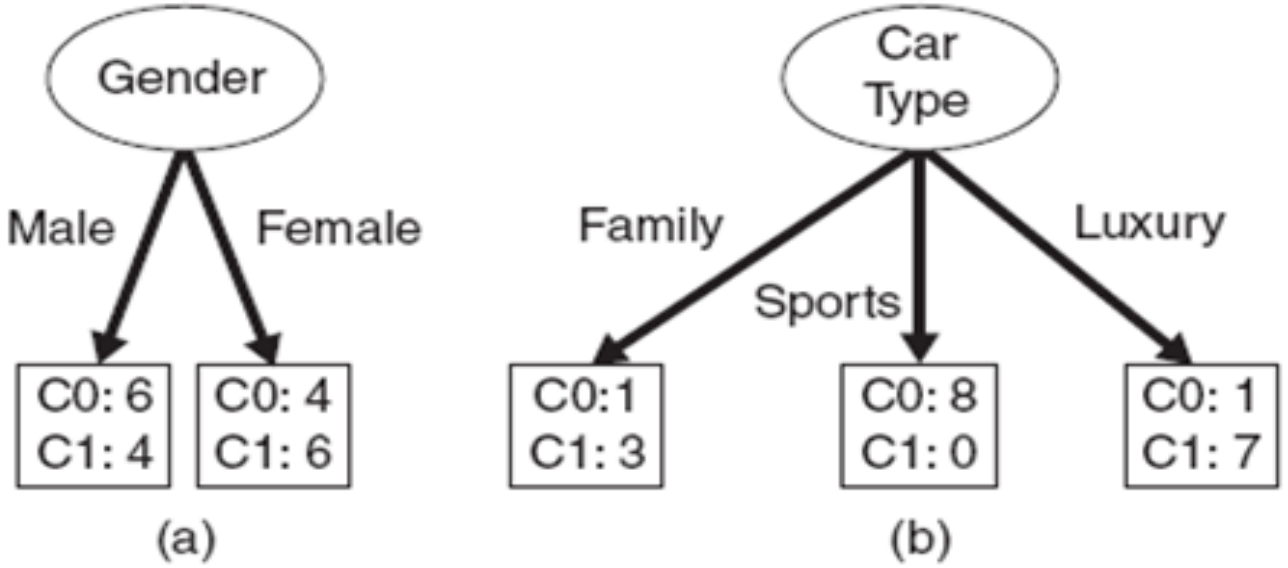
ID3's Question Revisited

ID3 builds a decision tree by recursively selecting attributes and splitting the training data based on the values of these attributes

Practically:

Given a training set, how do we select attributes so that the resulting tree is as small as possible, i.e. contains as few attributes as possible?

Selecting the best split

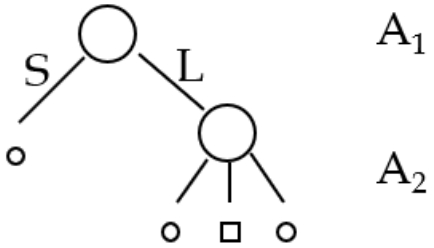


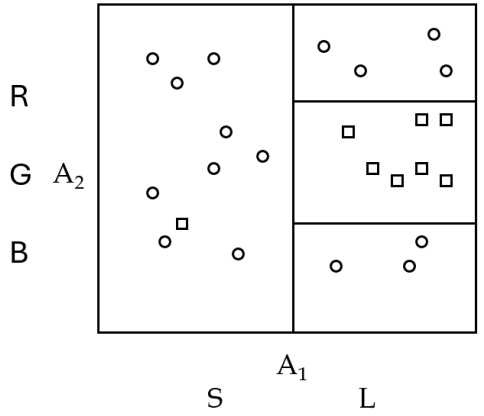
If you have a choice of these two attributes as the next one to split on, which would you choose?
Why?

Selecting the best split

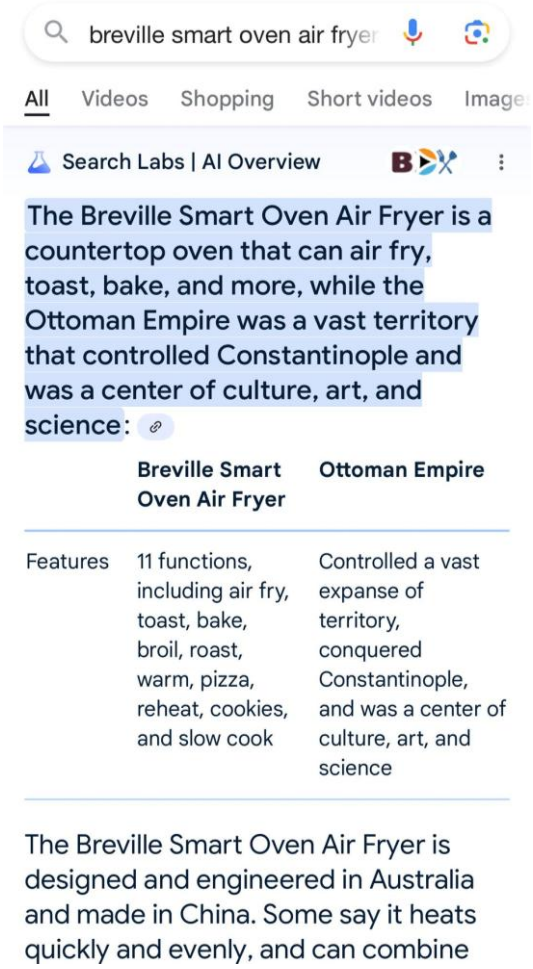
- **p(i|t)**: fraction of records associated with node **t** belonging to class **i**
- **Best split** is selected based on the degree of **impurity** of the child nodes
  - Class distribution **(0,1)** has **high purity**
  - Class distribution **(0.5,0.5)** has the **smallest purity (highest impurity)**

- **Intuition:**
  - high purity → small value of impurity measures → better split





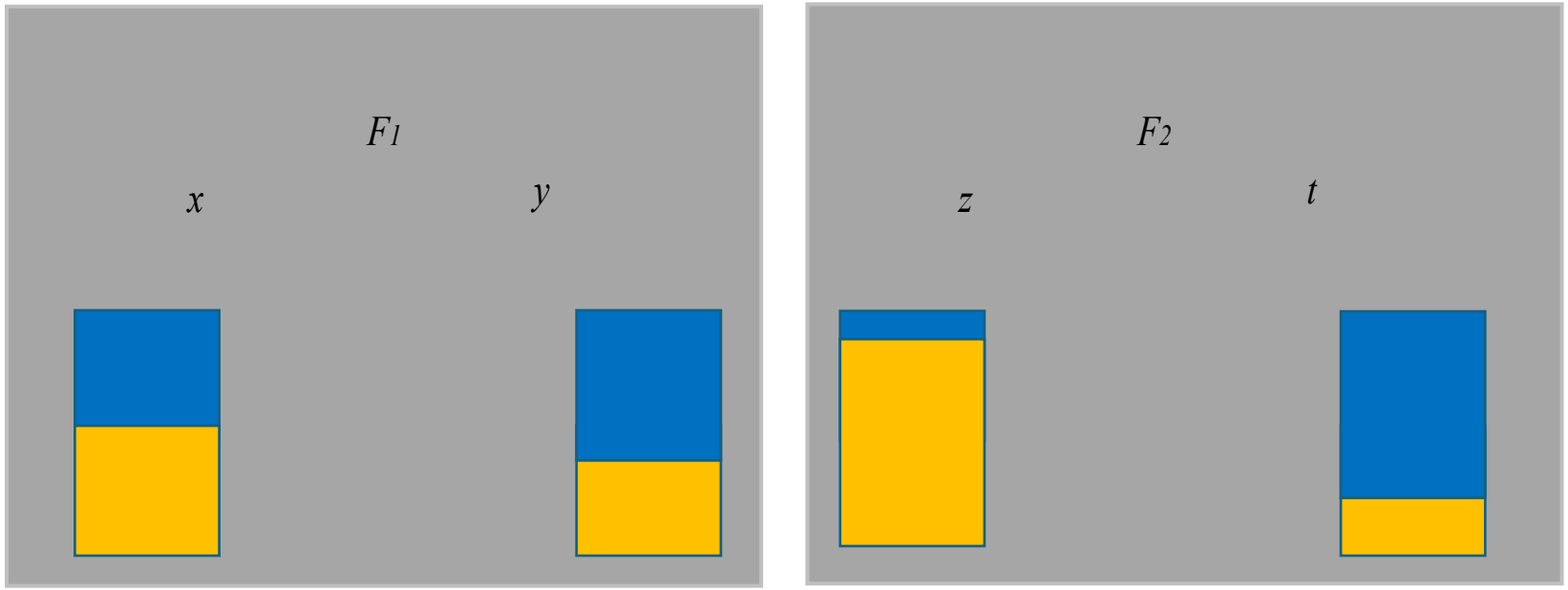
Break Time!


breville smart oven air fryer
All
Videos
Shopping
Short videos
Image
Search Labs | AI Overview
The Breville Smart Oven Air Fryer is a countertop oven that can air fry, toast, bake, and more, while the Ottoman Empire was a vast territory that controlled Constantinople and was a center of culture, art, and science:
Breville Smart Oven Air Fryer
Ottoman Empire
Features
11 functions, including air fry, toast, bake, broil, roast, warm, pizza, reheat, cookies, and slow cook
Controlled a vast expanse of territory, conquered Constantinople, and was a center of culture, art, and science
The Breville Smart Oven Air Fryer is designed and engineered in Australia and made in China. Some say it heats quickly and evenly, and can combine

Purity Measures

Purity/Information/Entropy (I)

- Assume 2 classes (Yellow and Blue) and 2 features F_1 and F_2, each with two values, such that:



- Which of F_1 or F_2 would we choose and why?

Impurity measures

- **p(i|t)**: Probability of class i given you are in node t
  - The fraction of records associated with node **t** belonging to class **i**

$$\text{Entropy}(t) = -\sum_{i=1}^{c} p(i \mid t) \log p(i \mid t)$$

$$\text{Gini}(t) = 1 - \sum_{i=1}^{c} \left[p(i \mid t) \right]^2$$

$$\text{Classification error}(t) = 1 - \max_i \left[p(i \mid t) \right]$$

Entropy

Entropy (Information Theory)

- Can we quantify surprise?
- Consider two possible pieces of information, with their respective probabilities:
- "The sun will rise tomorrow." (P=.99999)
  - How much information does that give us?
  - Pretty much none.
- "Aliens just landed in Provo." (P=0.00001)
  - How much information does that give us?
  - Lots!
- **Information is the *opposite* of Probability.**
- If an event is likely (P≈1), it contains almost **0 bits** of news.
- If an event is rare (P≈0), it contains **many bits** of news.

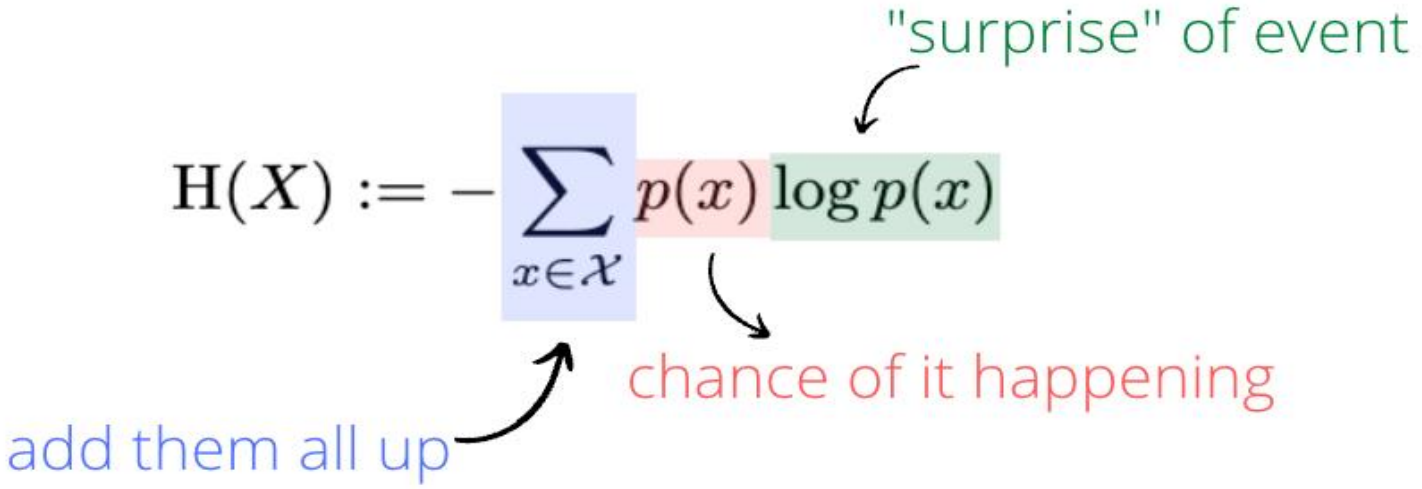
Entropy

- Let's think about the equation for entropy:
- We take a log! We do that a lot, but why in this context?
- We want a function that goes from 0→∞ as probability goes from 1→0.
  - Try P=1 (Certainty): log(1)=0. → 0 Surprise. (Perfect!)
  - Try P=0.5 (Coin Flip): −log2(0.5)=1. → 1 bit of Surprise.
  - Try P=0.0 (Impossible): −log(0)=∞. → Infinite Surprise.

$$\text{Entropy}(t) = -\sum_{i=1}^{c} p(i \mid t) \log p(i \mid t)$$

Entropy

- Now, we need to combine the surprises for the whole dataset.
- Entropy is simply the **Expected Value** (Weighted Average) of the surprise.
- "If I pick a random data point, how surprised do I expect to be on average?"

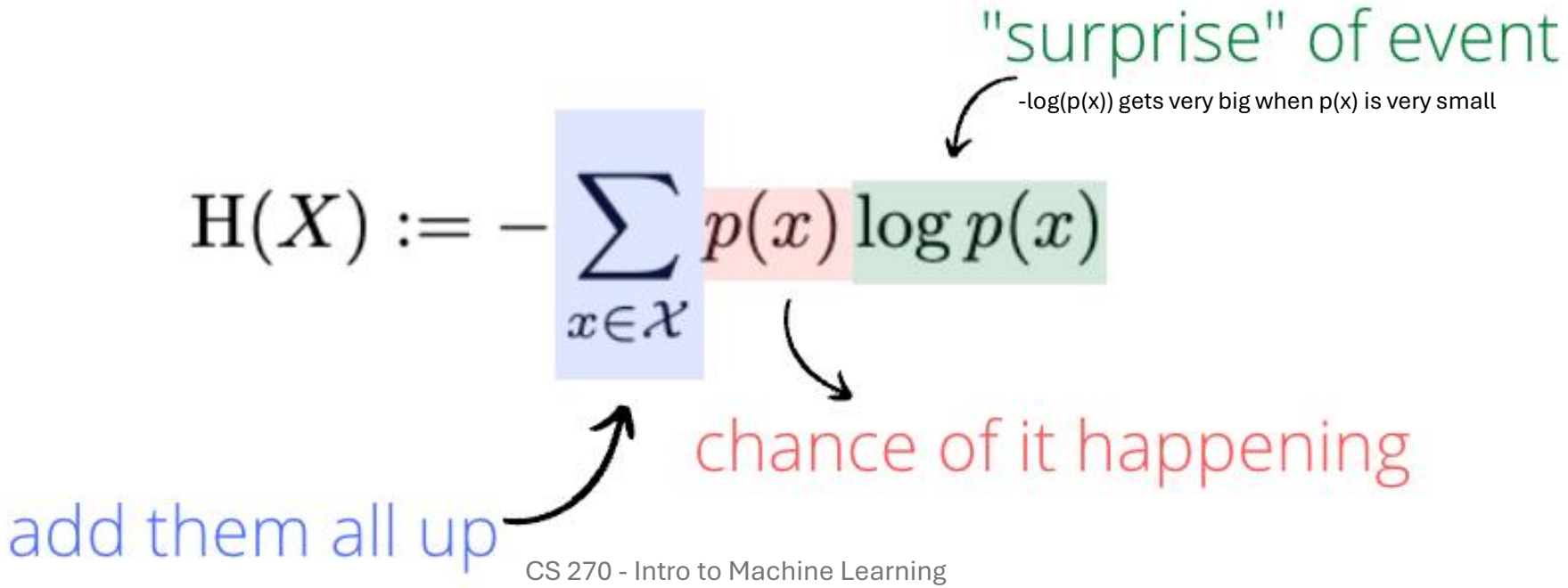


Entropy

Entropy at a **given node** t:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

(where $p(j \mid t)$ is the relative frequency of class j at node t



Examples of Computing Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Entropy = – 0 $\log_2$ 0 + –1 $\log_2$ 1 = – 0 – 0 = 0

log(0) = ∞ but multiplied by 0 log(1) = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (5/6) = 0.65

C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

Questions About Entropy?

Gini Impurity

Gini Impurity

- Entropy involves calculating lots of logs.
- Can you do a log by hand?
- Logs are expensive to compute.
- Can we approximate surprise with a simpler calculation?

Gini Impurity

- Simpler calculation – ‘pick two’ game from a bag of items.
- Reach into the bag and pull out two random items.
- **The Question:** "What are the odds that these two items satisfy different classes?"
  - **Pure Bag (All Blue):** You will *always* pull two Blue balls. They match. **Impurity = 0.**
  - **Mixed Bag (50 Blue / 50 Red):** There is a 50% chance you pull one Blue and one Red. They mismatch. **Impurity = 0.5.**

$$Gini = 1 - \sum (p_i)^2$$

Range of impurity measures

$$\text{Entropy}(t) = -\sum_{i=1}^{c} p(i \mid t) \log p(i \mid t)$$

$$\text{Gini}(t) = 1 - \sum_{i=1}^{c} [p(i \mid t)]^2$$

$$\text{Classification error}(t) = 1 - \max_i [p(i \mid t)]$$

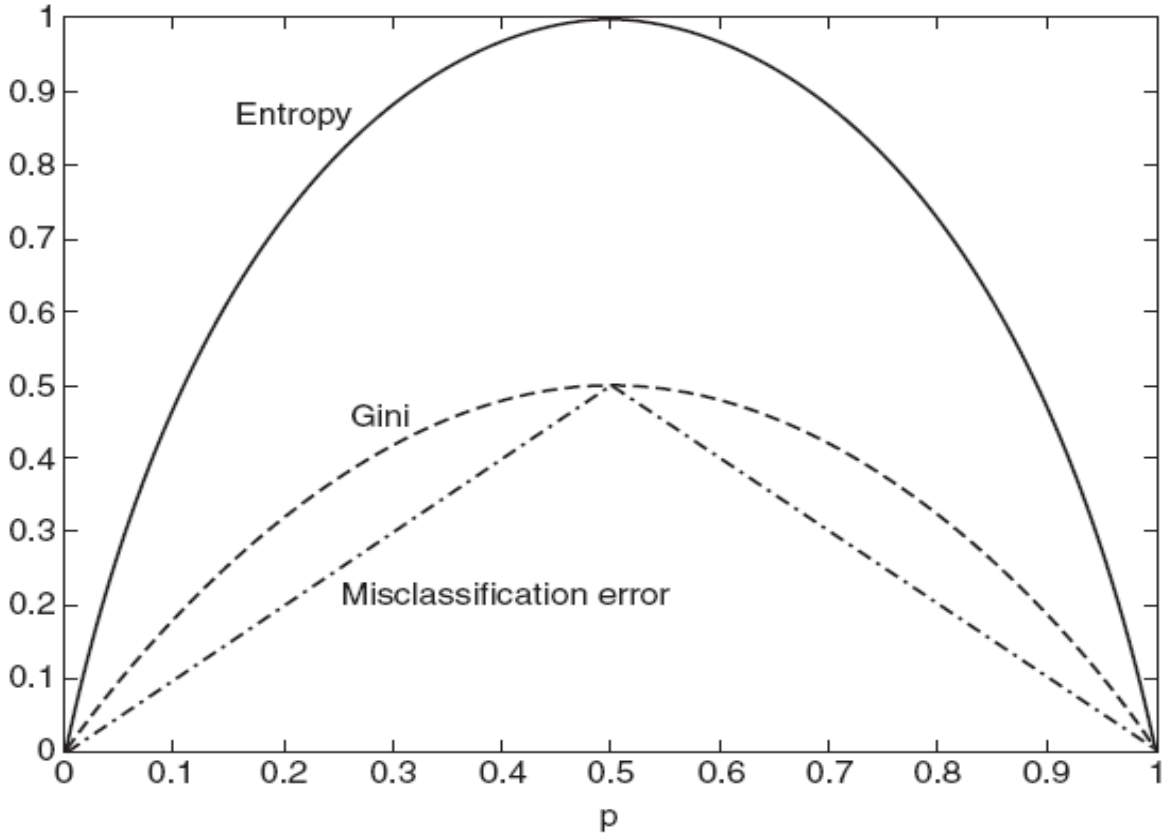


Figure 4.13. Comparison among the impurity measures for binary classification problems.

Does the Gini impurity look like a reasonable proxy for entropy?

Why don't we use misclassification error?

Impurity measures

- In general the different impurity measures are ***consistent***
- ***Gain of a test condition:*** compare the impurity of the parent node with the impurity of the child nodes

$$\Delta = I(parent) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$

- Maximizing the gain == minimizing the weighted average impurity measure of children nodes
- $Gain = I(Parent) - Average\ I(Children)$

Quiz Time!

Questions About Anything From Today?

High Level Review

- Trees are a data structure.
- We can construct *Decision Trees*, a classifier that uses trees to construct interpretable, intuitive decision processes.
- There are multiple algorithms for constructing decision trees.
  - We want to have the purest leaf nodes (cleanest classification)