slides originally by
Dr. Richard Burns,
modified by
Dr. Stephanie Schwartz

# DECISION TREES

CSCI 452: Data Mining

# Today

- Decision Trees
  - Structure
  - Information Theory: Entropy
  - Information Gain, Gain Ratio, Gini
  - ID3 Algorithm
    - Efficiently Handling Continuous Features
  - Overfitting / Underfitting
    - Bias-Variance Tradeoff
  - Pruning
  - Regression Trees

# Motivation: Guess Who Game

- I'm thinking of one of you.
- Figure out who I'm thinking of by asking a series of *binary* questions.

# Decision Trees

- Simple, yet widely used *classification* technique
  - For nominal target variables
    - There also are *Regression* trees, for continuous target variables
  - *Predictor Features*: binary, nominal, ordinal, discrete, continuous
  - *Evaluating the model:*
    - One metric: error rate in predictions

# Decision Trees

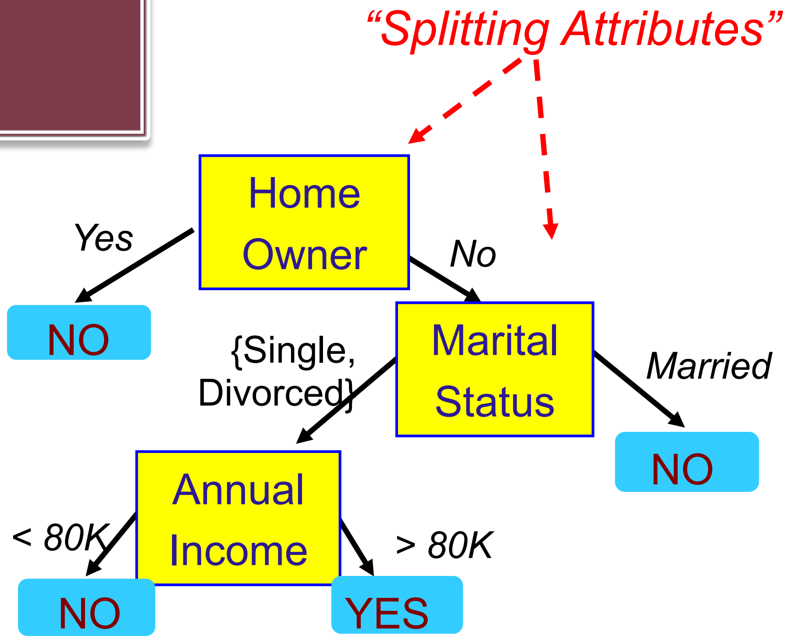
1. Root node
2. Internal nodes
3. Leaf / terminal nodes

Tree is **consistent** with training dataset.

binary categorical continuous class

| *Tid* | **Home Owner** | **Marital Status** | **Annual Income** | **Defaulted Borrower** |
|---|---|---|---|---|
| 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




Decision Tree Model #1

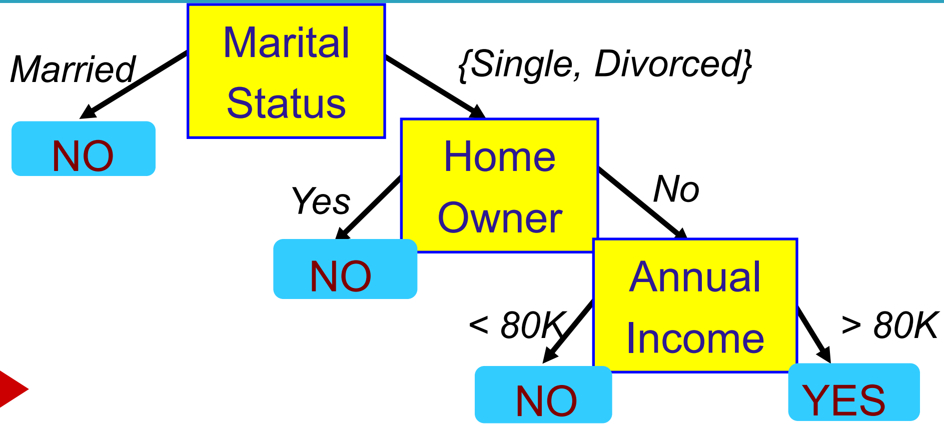
# Decision Trees

Tree is **consistent** with training dataset.

binary categorical continuous class

| *Tid* | **Home Owner** | **Marital Status** | **Annual Income** | **Defaulted Borrower** |
|---|---|---|---|---|
| 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




Decision Tree Model #2

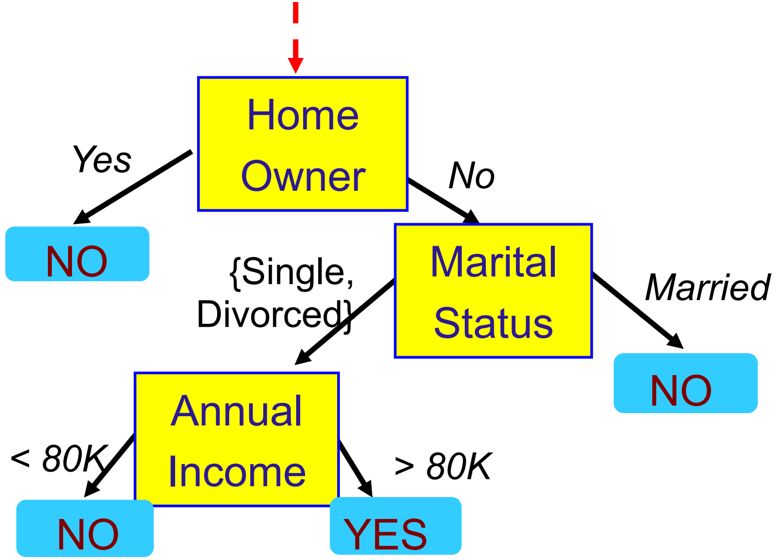
There could be more than one tree that fits the same data!

# Decision Tree Classifier

- Decision tree models are relatively *more descriptive* than other types of classifier models
  1. Easier interpretation
  2. Easier to explain predicted values
- Exponentially many decision trees can be built
  - Which is best?
    - Some trees will be more accurate than others
  - How to construct the tree?
    - Computationally infeasible to try every possible tree.

# Apply Model to Test Data

Start from the root of tree.

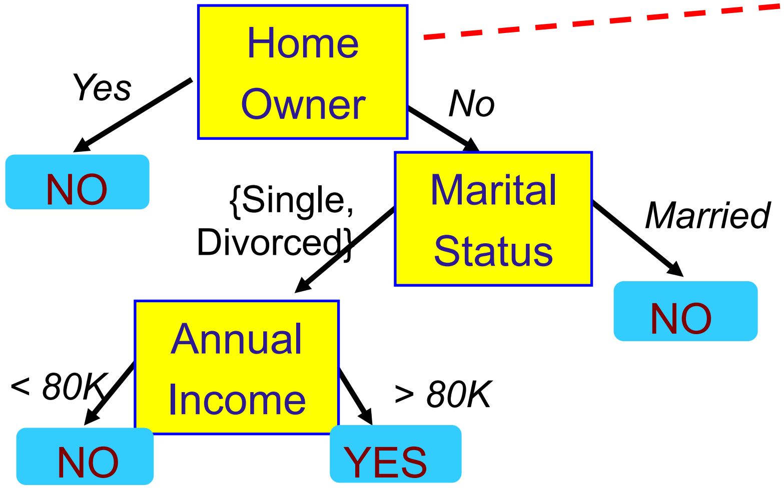



Test Data

| Home Owner | Marital Status | Annual Income | Defaulted Barrower |
|---|---|---|---|
| No | Married | 80K | ? |

# Apply Model to Test Data

Start from the root of tree.

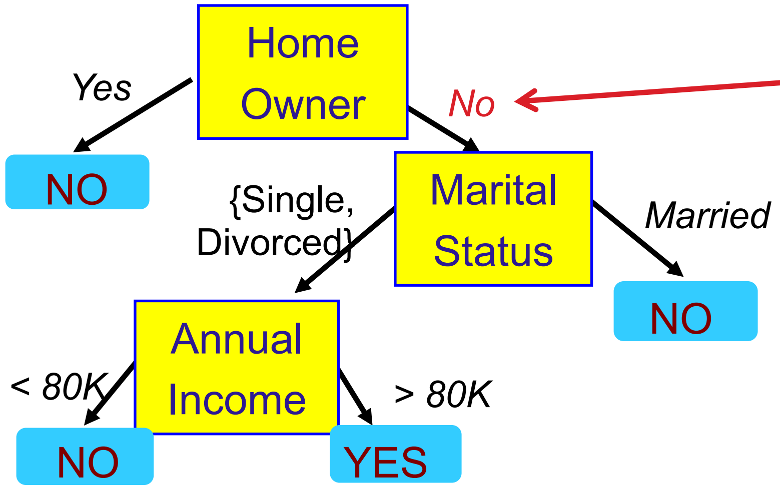



Test Data

| Home Owner | Marital Status | Annual Income | Defaulted Barrower |
|---|---|---|---|
| No | Married | 80K | ? |

# Apply Model to Test Data

Start from the root of tree.

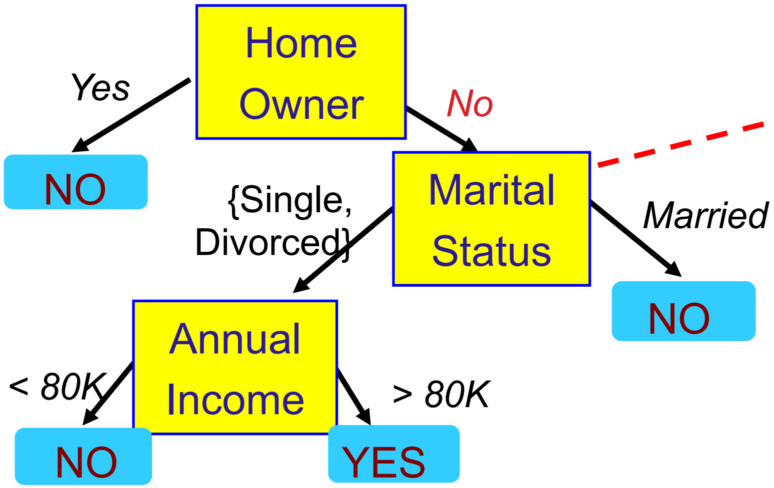



Test Data

| Home Owner | Marital Status | Annual Income | Defaulted Barrower |
|---|---|---|---|
| No | Married | 80K | ? |

# Apply Model to Test Data

Start from the root of tree.

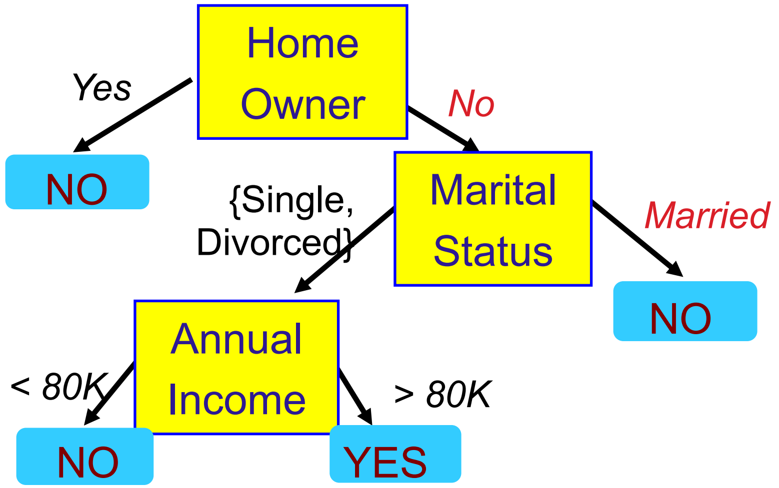



Test Data

| Home Owner | Marital Status | Annual Income | Defaulted Barrower |
|---|---|---|---|
| No | Married | 80K | ? |

# Apply Model to Test Data

Start from the root of tree.

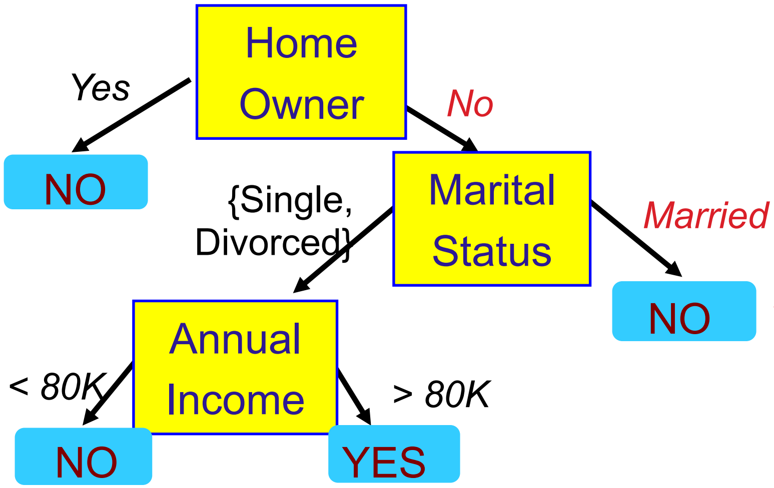



Test Data

| Home Owner | Marital Status | Annual Income | Defaulted Barrower |
|---|---|---|---|
| No | Married | 80K | ? |

# Apply Model to Test Data

Start from the root of tree.




Test Data

| Home Owner | Marital Status | Annual Income | Defaulted Barrower |
|---|---|---|---|
| No | Married | 80K | ? |

- Predict that this person will <u>not</u> default.

# Formally…

- A decision tree has three types of nodes:
  1. A root node that has no incoming edges and zero or mode outgoing edges
  2. Internal nodes, each of which has exactly one incoming edge and two or more outgoing edges
  3. Leaf nodes (or terminal nodes), each of which has exactly one incoming edge and no outgoing edges
- Each leaf node is assigned a class label
- Non-terminal nodes contain attribute test conditions to separate records that have different characteristics

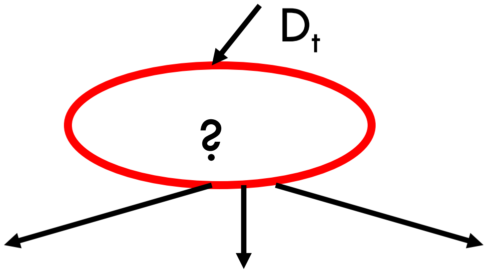
# How to Build a Decision Tree?

- Referred to as <u>decision tree induction</u>.
- Exponentially many decision trees can be constructed from a given set of attributes
  - Infeasible to try them all to find the optimal tree
- Different "decision tree building" algorithms:
  - Hunt's algorithm, CART, ID3, C4.5, …
- Usually a <u>greedy strategy</u> is employed

# Hunt's Algorithm

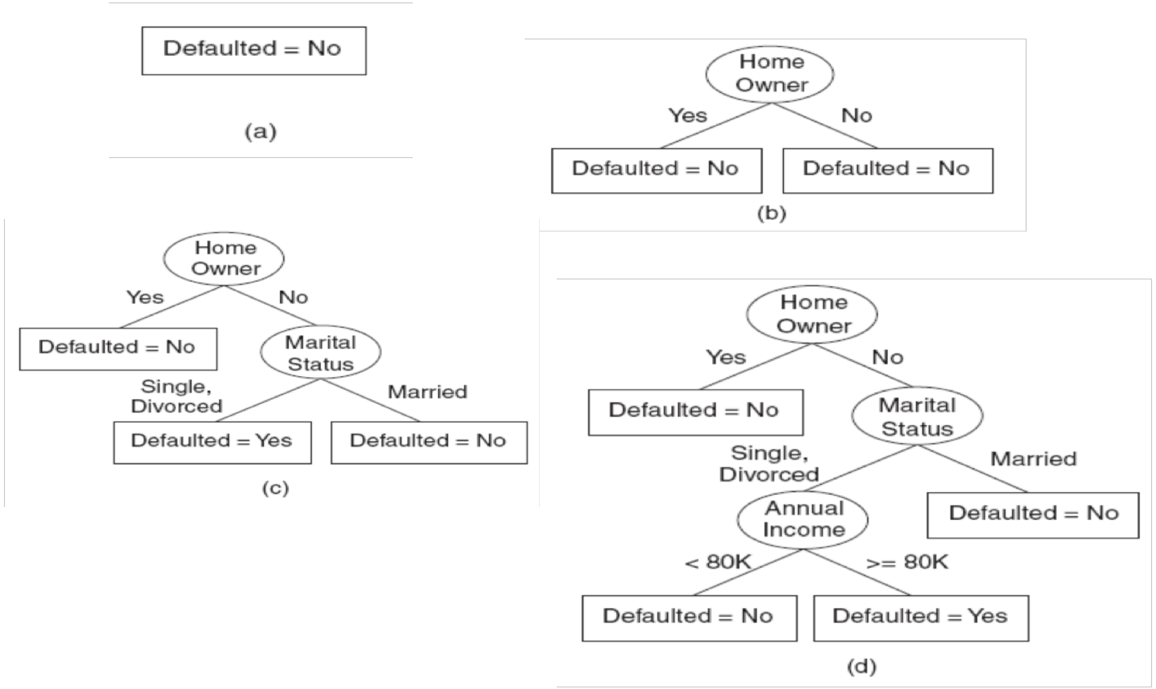
- $D_t$ = set of training records that reach a node t
- Recursive Procedure:
  1. If all records in $D_t$ belong to the same class:
     - then $t$ is a leaf node with class $y_t$
  2. If $D_t$ is an empty set:
     - then $t$ is a leaf node, class determined by the majority class of records in $D_t$'s parent
  3. If $D_t$ contains records that belong to more than one class:
     - use an attribute test to split the data into smaller subsets. Recursively apply the procedure to each subset.

| *Tid* | **Home Owner** | **Marital Status** | **Annual Income** | **Defaulted Borrower** |
|---|---|---|---|---|
| 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 |



# Hunt's Algorithm

| *Tid* | Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|---|
| 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 |




- Tree begins with single node whose label reflects the majority class value
- Tree needs to be refined because root node contains records from both classes
- Divide records recursively into smaller subsets

# Hunt's Algorithm

| Tid | Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|---|
| 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 |

- Hunt's Algorithm will work if:
  - Every combination of attribute values is present
    - *Question*: realistic or unrealistic?
    - Unrealistic: at least $2^n$ records necessary for binary attributes
      - *Examples:* no record for {HomeOwner=Yes, Marrital=Single, Income=60K}
  - Each combination of attribute values has unique class label
    - *Question:* realistic or unrealistic?
    - Unrealistic.
      - *Example:* Suppose we have two individuals, each with the properties {HomeOwner=Yes, Marrital=Single, Income=125K}, but one person defaulted and the other did not.

# Hunt's Algorithm

- Scenarios the algorithm may run into:
  1. All records associated with $D_t$ have identical attributes except for the class label (not possible to split anymore)
     - *Solution:* declare a leaf node with the same class label as the majority class of $D_t$
  2. Some child nodes are empty (no records associated, no combination of attribute values leading to this node)
     - *Solution:* declare a leaf node with the same class label as the majority class of the empty node's parent

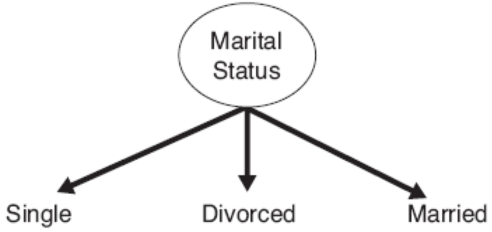
# Design Issues of Decision Tree Induction

1. How should the training records be split?
   - *Greedy strategy*: split the records based on some attribute test (always choose immediate best option)
   - Need to evaluate the "goodness" of each attribute test and select the best one.
2. How should the splitting procedure stop?
   - For now, we'll keep splitting until we can't split anymore.

# Different Split Ideas…

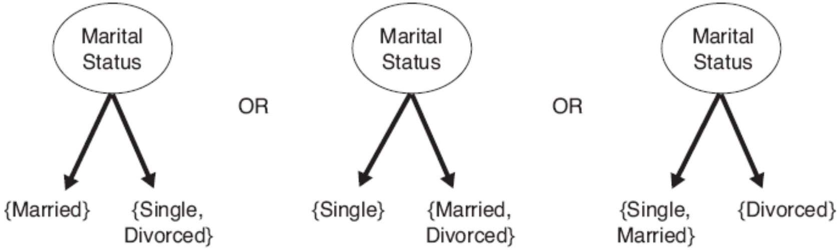
# Splitting Based on Nominal Attributes

- <u>Multiway Split</u>: Use as many partitions as distinct values



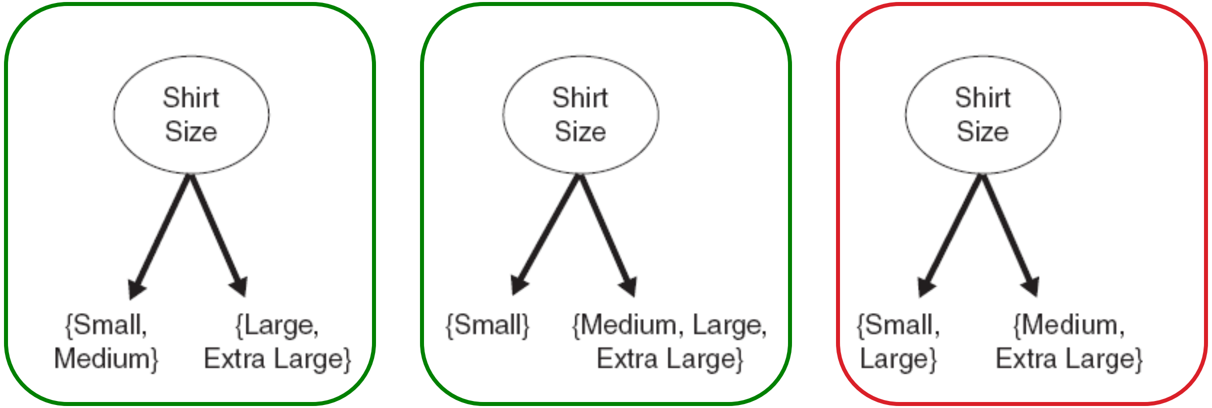

- <u>Binary Split</u>: Grouping attribute values
  - Need to find optimal partitioning

CART decision tree algorithm only creates binary splits.

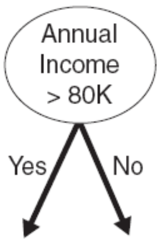


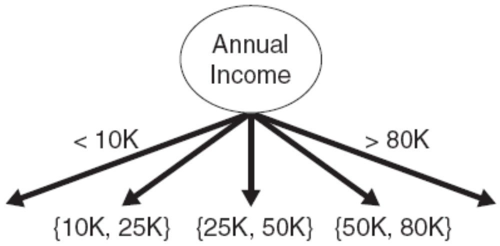
# Splitting Based on Ordinal Attributes

- Ordinal attributes can also produce binary or multiway splits.
  - Grouping should not violate ordering in the ordinal set



# Splitting Based on Continuous Attributes

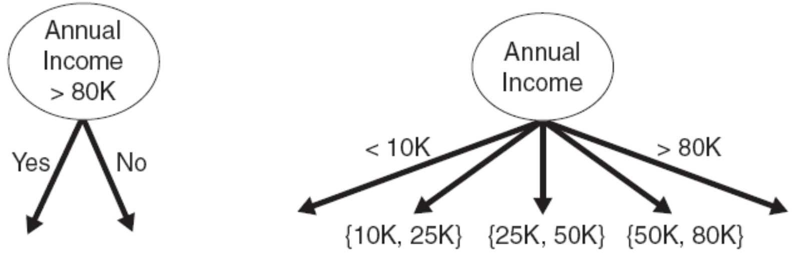



| *Tid* | **Home Owner** | **Marital Status** | **Annual Income** | **Defaulted Borrower** |
|---|---|---|---|---|
| 1 | Yes | Single | 125K | **No** |
| 2 | No | Married | 100K | **No** |
| 3 | No | Single | 70K | **No** |
| 4 | Yes | Married | 120K | **No** |
| 5 | No | Divorced | 95K | **Yes** |
| 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** |

- Continuous attributes can also have a binary or multiway split.
  - *Binary*: decision tree algorithm must consider all possible split positions *v*, and it selects the best one
    - *Comparison test: (A <= v) or (A > v)*, where *v=80K*
  - Computationally intensive

# Splitting Based on Continuous Attributes




- Continuous attributes can also have a <u>binary</u> or <u>multiway split</u>.
  - *Multiway*: outcomes of the form $v_i \leq A < v_{i+1},\ \text{for } i = 1, \ldots, k$
    - Consider all possible ranges of continuous variable?
    - Use same binning strategies as discussed for preprocessing a continuous attribute into a discrete one
  - Note: adjacent intervals/“bins” can always be aggregated into wider ones

# Tree Induction

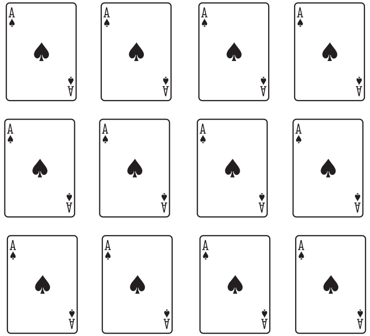
- What to split on?
  - Home Owner
  - Marital Status
    - Multiway or binary?
  - Annual Income
    - Multiway or binary?

| *Tid* | **Home Owner** | **Marital Status** | **Annual Income** | **Defaulted Borrower** |
|---|---|---|---|---|
| 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 |


(a)

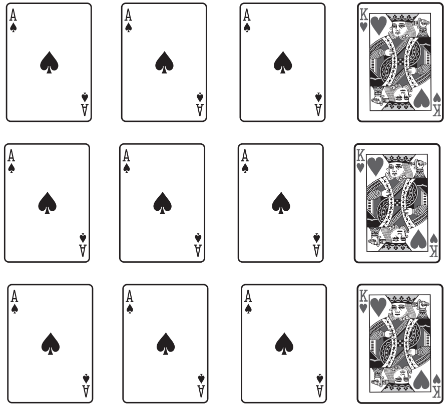
# Entropy

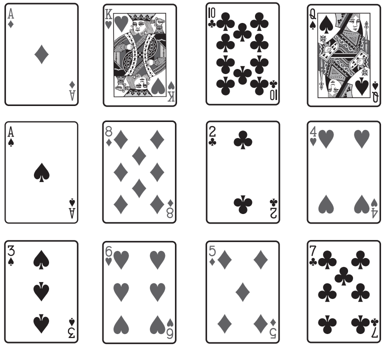
- Defined by mathematician Claude Shannon
- Measures the impurity (heterogeneity) of the elements of a set
- *“what is the uncertainty of guessing the result of the random selection from a set?”*



Very pure. Not impure.

Little impure.





Completely impure.

# Entropy

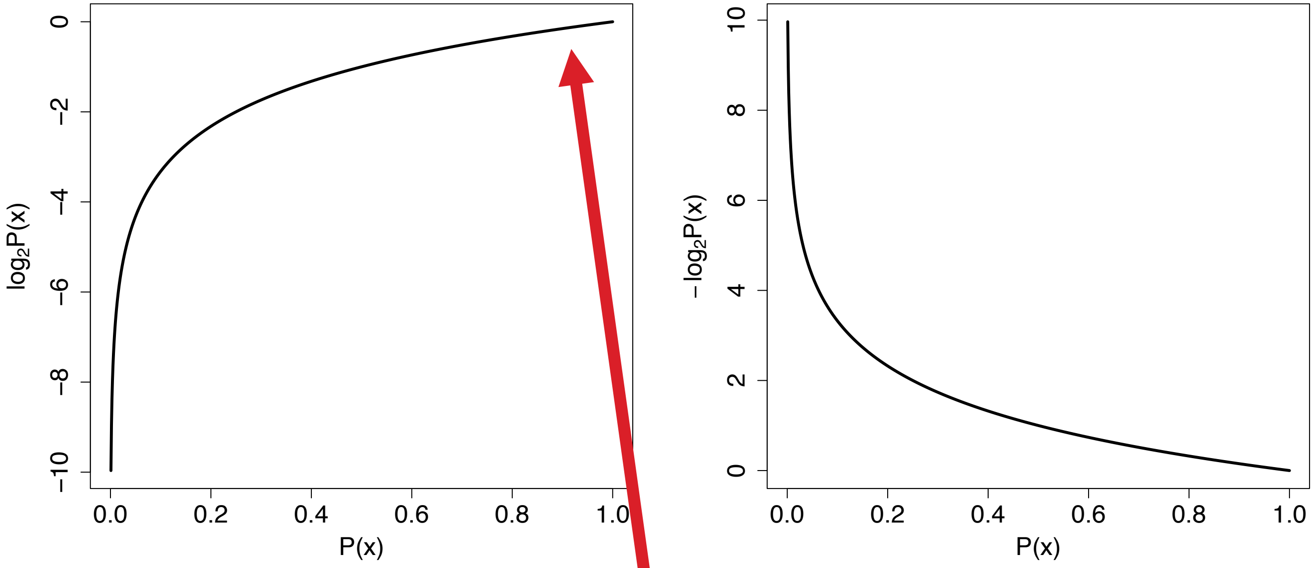
- $Entropy(n) = -\sum_{i=1}^{c} p_i \log_2 p_i$
- Weighted sum of the logs of the probabilities of each of the possible outcomes.

# Entropy Examples

1. Entropy of the set of 52 playing cards:
   - Randomly selecting any specific card *i* is 1/52.
   - $Entropy(n) = -\sum_{i=1}^{52} 0.019 \log_2 0.019 = 5.7$
2. Entropy if only the 4 suits matter:
   - Randomly selecting any suit is ¼
   - $Entropy(n) = -\sum_{i=1}^{4} 0.25 \log_2 0.25 = 2$

The higher the impurity, the higher the entropy.

# That's the Reason for Using the *log* function



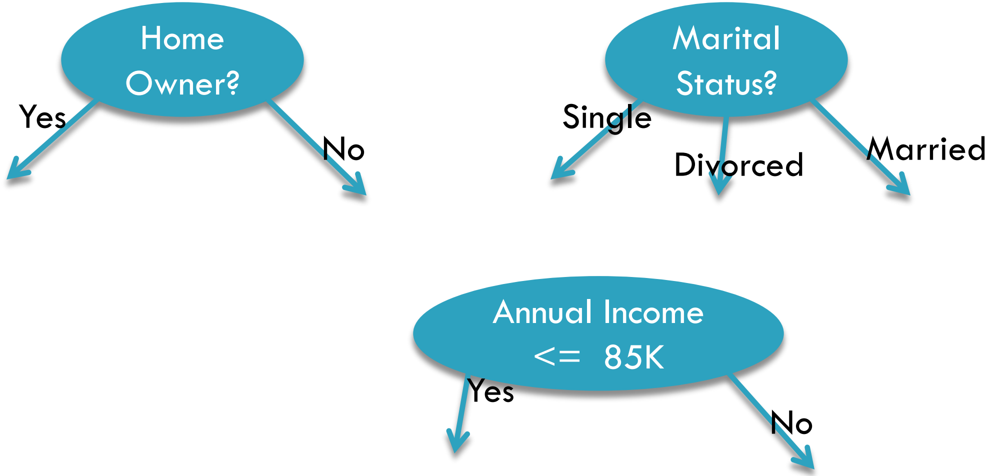

(a) (b)

Want a low "score" when something is highly probable or certain.

# How to determine the Best Split?

How does entropy help us?

- We can calculate the entropy (impureness) of *Default Borrower*




| *Tid* | Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|---|
| 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 |

# Information Gain

- Try to split on each possible feature in a dataset. See which split works “best”.
- Measure the *reduction* in the overall entropy of a set of instances
- $InformationGain = Entropy(S) - \sum_i \frac{|S_i|}{|S|} E(S_i)$

Weighting Term

# Information Gain Example

$$Entropy_{START} = -\sum_{i=1}^{c} p_i \log_2 p_i = -(\frac{3}{6}\log_2\frac{3}{6} + \frac{3}{6}\log_2\frac{3}{6}) = 1$$

$$InformationGain = Entropy(S) - \sum_i \frac{|S_i|}{|S|} E(S_i)$$

$$IG_{CI} = 1 - \left(\frac{2}{6} \times \left(-\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right)\right) + \frac{4}{6} \times \left(-\left(\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}\right)\right)\right) = 0$$

| ID | Suspicious Words | Unknown Sender | Contains Images | Class |
|---|---|---|---|---|
| 376 | true | false | true | spam |
| 489 | true | true | false | spam |
| 541 | true | true | false | spam |
| 693 | false | true | true | ham |
| 782 | false | false | false | ham |
| 976 | false | false | false | ham |

- The 0 means there was no "information gain".
- Nothing was learned by splitting on "Contains Images".

# Calculate the Information Gain on Each Feature

| ID | Suspicious Words | Unknown Sender | Contains Images | Class |
|---|---|---|---|---|
| 376 | true | false | true | spam |
| 489 | true | true | false | spam |
| 541 | true | true | false | spam |
| 693 | false | true | true | ham |
| 782 | false | false | false | ham |
| 976 | false | false | false | ham |

- $IG_{SW} = 1 - 0 = 1$
- $IG_{US} = 1 - .9183 = .0817$
- $IG_{CI} = 1 - 1 = 0$

"Suspicious Words" is the best split.

# ID3 Algorithm

- Attempts to create the *shallowest* tree that is *consistent* with the training dataset
- Builds the tree in a recursive, depth-first manner
  - beginning at the root node and working down to the leaf nodes

# ID3 Algorithm

1. Figure out the best feature to split on based on by using information gain
2. Add this root note to the tree; label it with the selected test feature
3. Partition the dataset using this test
4. For each partition, grow a branch from this node
5. Recursively repeat the process for each of these branches using the remaining partition of the dataset

# ID3 Algorithm: Stopping Condition

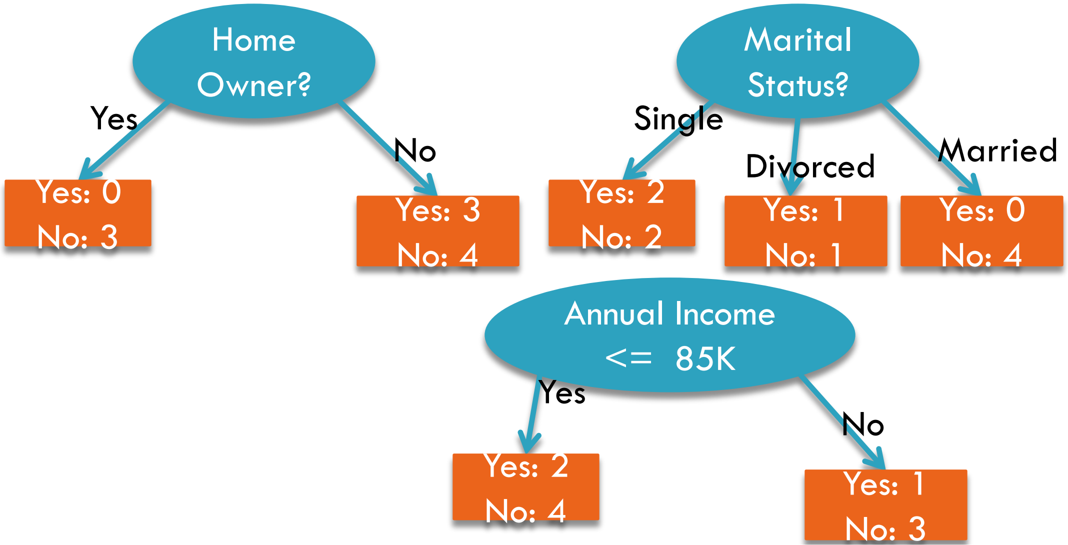
Stop the recursion and construct a leaf node when:

1. All of the instances in the remaining dataset have the same classification (target feature value).
   - Create a leaf node with that classification as its label
2. The set of features left to test is empty.
   - Create a leaf node with the majority class of the dataset as its classification.
3. The remaining dataset is empty.
   - Create a leaf note one level up (parent node), with the majority class.

# Determine the Best Split

- Before:
  - 7 records of class No
  - 3 records of class Yes

| *Tid* | Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|---|
| 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 |




Which test condition is best?

# Other Measures of Node Impurity

1. Gini Index ("*genie*")
2. Entropy
3. Misclassification Error

$$Entropy(n) = -\sum_{i=0}^{c-1} p_i \log_2 p_i$$

$$Gini(n) = 1 - \sum_{i=0}^{c-1} [p_i]^2$$

$$MisclassificationError(n) = 1 - \max p_i$$

- c = # of classes
- 0 log 0 = 0
- $p_i$ = fraction of records belonging to class *i* at a given node.

# Example Calculations

| Node $N_1$ | Count |
|---|---|
| Class = 0 | 0 |
| Class = 1 | 6 |

$$Gini = 1 - \left(\frac{0}{6}\right)^2 - \left(\frac{6}{6}\right)^2 = 0$$

$$Entropy = -\left(\frac{0}{6}\right)\log_2\left(\frac{0}{6}\right) - \left(\frac{6}{6}\right)\log_2\left(\frac{6}{6}\right) = 0$$

$$Misclassification = 1 - \max\left[\frac{0}{6}, \frac{6}{6}\right] = 0$$

| Node $N_2$ | Count |
|---|---|
| Class = 0 | 1 |
| Class = 1 | 5 |

$$Gini = 1 - \left(\frac{1}{6}\right)^2 - \left(\frac{5}{6}\right)^2 = 0.278$$

$$Entropy = -\left(\frac{1}{6}\right)\log_2\left(\frac{1}{6}\right) - \left(\frac{5}{6}\right)\log_2\left(\frac{5}{6}\right) = 0.650$$

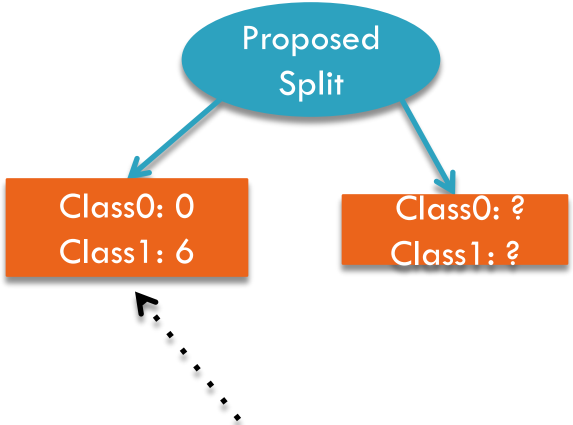
$$Misclassification = 1 - \max\left[\frac{1}{6}, \frac{5}{6}\right] = 0.167$$

| Node $N_3$ | Count |
|---|---|
| Class = 0 | 3 |
| Class = 1 | 3 |

$$Gini = 1 - \left(\frac{3}{6}\right)^2 - \left(\frac{3}{6}\right)^2 = 0.5$$

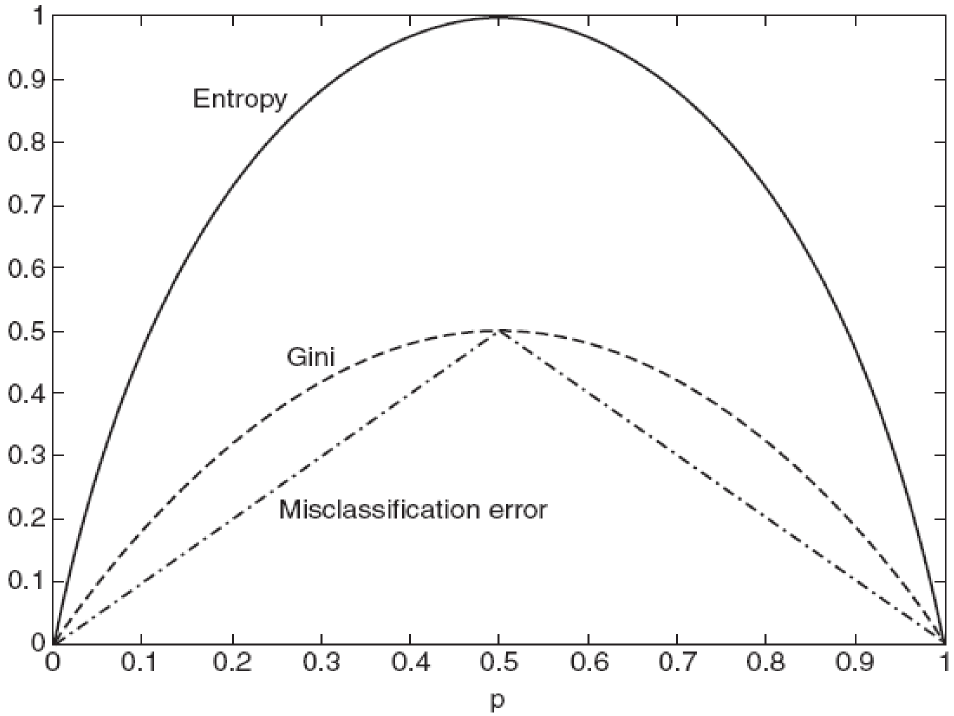
$$Entropy = -\left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right) - \left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right) = 1$$

$$Misclassification = 1 - \max\left[\frac{3}{6}, \frac{3}{6}\right] = 0.5$$




How “impure” is this node that would be created if we do the split?

# Comparing Impurity Measures for Binary Classification Problems

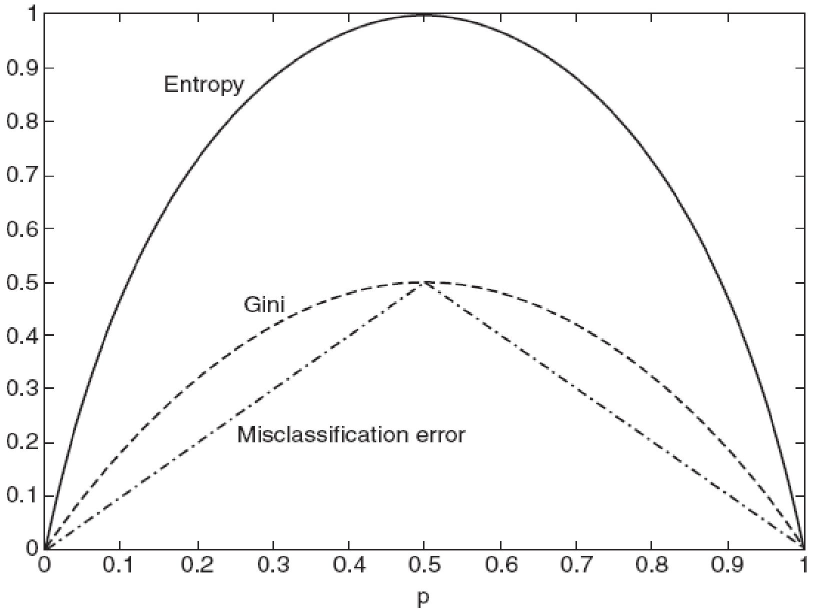



- Assuming only <u>two</u> classes.
- $p$ = fraction of records that belong to one of the two classes

| | |
|---|---|
| Class0: 3 | |
| Class1: 3 | |

$$p = \frac{3}{6} = .5$$

# Comparing Impurity Measures




- Consistency among different impurity measures
- But attribute chosen as the test condition *may vary* depending on impurity measure choice

# Gain

- Gain: "goodness of the split"
- Comparing:
  - degree of impurity of parent node (before splitting)
  - degree of impurity of the child nodes (after splitting), *weighted*
- larger gain => better split (better test condition)

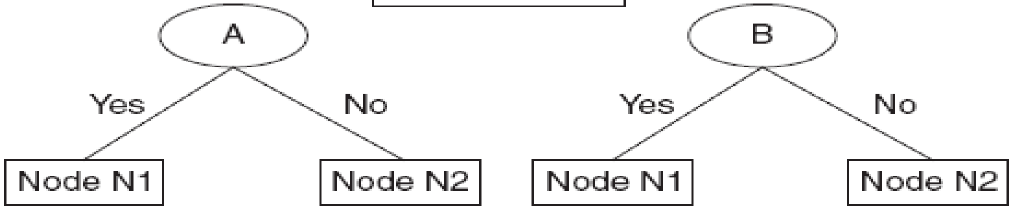
$$\Delta(gain) = I(parent) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$

- $I(n)$ = impurity measure at node $n$
- $I(v_j)$ = impurity measure at child node $v_j$
- $k$ = number of attribute values
- $N(v_j)$ = total number of records at child node $v_j$
- $N$ = total number of records at parent node

*Footnote:* (Information Gain: term used when *entropy* is used as the impurity measure)

# Gain Example

| | Parent |
|---|---|
| C0 | 6 |
| C1 | 6 |
| Gini = 0.500 | |

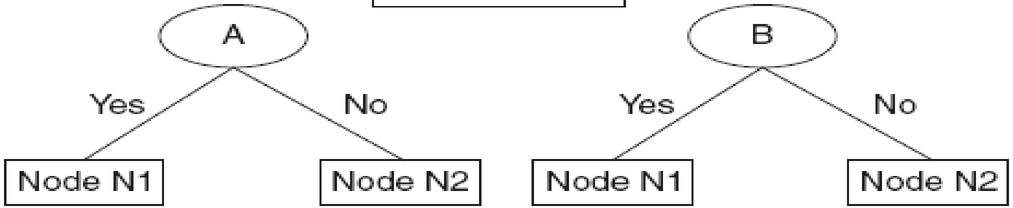



| | N1 | N2 |
|---|---|---|
| C0 | 4 | 2 |
| C1 | 3 | 3 |
| Gini = ? | | |

| | N1 | N2 |
|---|---|---|
| C0 | 1 | 5 |
| C1 | 4 | 2 |
| Gini = ? | | |

- What is the Gini (index) of the child nodes?
- Is gain greater for split A or split B?

| | Parent |
|---|---|
| C0 | 6 |
| C1 | 6 |
| Gini = 0.500 | |




| | N1 | N2 |
|---|---|---|
| C0 | 4 | 2 |
| C1 | 3 | 3 |
| Gini = 0.486 | | |

| | N1 | N2 |
|---|---|---|
| C0 | 1 | 5 |
| C1 | 4 | 2 |
| Gini = 0.371 | | |

$$Gini(N_1) = 1 - \left(\frac{3}{7}\right)^2 - \left(\frac{4}{7}\right)^2 = 0.4898 \quad Gini(N_2) = 1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2 = 0.480$$

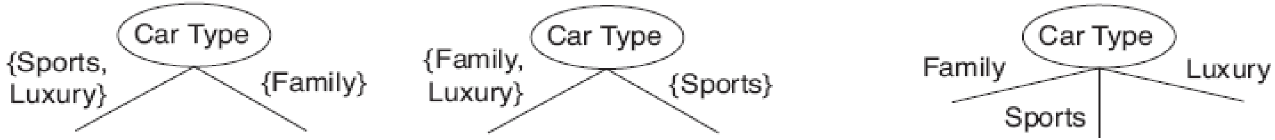
$$Gini\,WeightedAvg = \left(\frac{7}{12}\right) \times 0.4898 + \left(\frac{5}{12}\right) \times 0.480 = 0.486$$

$$Gain_A = .500 - .486 = .014$$

$$Gain_B = .500 - .375 = .125$$

Since descendent nodes after splitting with Attribute B have a smaller Gini index than after splitting with Attribute A, splitting with Attribute B is preferred. (The gain is greater.)

# Computing Multiway Gini index




| | Car Type | |
|---|---|---|
| | {Sports, Luxury} | {Family} |
| C0 | 9 | 1 |
| C1 | 7 | 3 |
| Gini | 0.468 | |

| | Car Type | |
|---|---|---|
| | {Sports} | {Family, Luxury} |
| C0 | 8 | 2 |
| C1 | 0 | 10 |
| Gini | 0.167 | |

(a) Binary split

| | Car Type | | |
|---|---|---|---|
| | Family | Sports | Luxury |
| C0 | 1 | 8 | 1 |
| C1 | 3 | 0 | 7 |
| Gini | 0.163 | | |

(b) Multiway split

- Computed for every attribute value.

$$Gini(\{Family\}) = 0.375$$
$$Gini(\{Sports\}) = 0$$
$$Gini(\{Luxury\}) = 0.219$$
$$OverallGiniIndex = \frac{4}{20}\times 0.375 + \frac{8}{20}\times 0 + \frac{8}{20}\times 0.219 = 0.163$$

# Binary Splitting of Continuous Attributes

- Need to find best value $v$ to split against
- Brute-force method:
  - Consider every attribute value $v$ as a split candidate
    - $O(n)$ possible candidates
    - For each candidate, need to iterate through all records again to determine the count of records with attribute $< v$ or $> v$
    - $O(n^2)$ *complexity*
- By sorting records by the continuous attribute, this improves to $O(n \log n)$

# Splitting of Continuous Attributes

- Need to find best value v to split against
- Brute-force method:
  - Consider every attribute value v as a split candidate

| Tid | Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|---|
| 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 |

- Need to find a split value for *AnnualIncome* predictor
- Try 125K?
  - Compute Gini for <= 125K and > 125K.
  - *Complexity:* O(n)
- Try 100K?
  - *Complexity:* O(n)
- Try 70K?, etc.
- *Overall complexity:* $O(n^2)$

# Splitting of Continuous Attributes

- By sorting records by the continuous attribute, this improves to O(*n log n*)
  - Candidate Split position are midpoints between two adjacent, different, class values
  - Only need to consider split positions: 80 and 97

| *Tid* | Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|---|---|---|---|---|
| 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 |

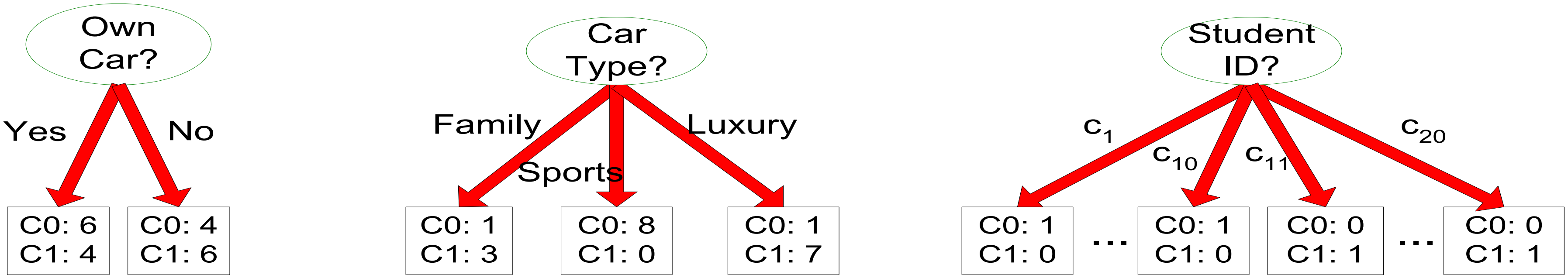
| Class | No | No | No | Yes | Yes | Yes | No | No | No | No |
|---|---|---|---|---|---|---|---|---|---|---|
| | Annual Income | | | | | | | | | |
| Sorted Values → | 60 | 70 | 75 | 85 | 90 | 95 | 100 | 120 | 125 | 220 |

| Split Positions → | 55 | | 65 | | 72 | | 80 | | 87 | | 92 | | 97 | | 110 | | 122 | | 172 | | 230 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > | <= | > |
| Yes | 0 | 3 | 0 | 3 | 0 | 3 | 0 | 3 | 1 | 2 | 2 | 1 | 3 | 0 | 3 | 0 | 3 | 0 | 3 | 0 | 3 | 0 |
| No | 0 | 7 | 1 | 6 | 2 | 5 | 3 | 4 | 3 | 4 | 3 | 4 | 3 | 4 | 4 | 3 | 5 | 2 | 6 | 1 | 7 | 0 |
| Gini | 0.420 | | 0.400 | | 0.375 | | 0.343 | | 0.417 | | 0.400 | | *0.300* | | 0.343 | | 0.375 | | 0.400 | | 0.420 | |

# Bias: Favoring attributes with large number of distinct values

- Entropy and Gini Impurity measures favor attributes with large number of distinct values

Possible nodes to split on:




- StudentId will result in perfectly pure children.
- Will have the greatest gain.
- Should have been removed as a predictor variable.

# Gain Ratio

C4.5 algorithm uses Gain Ratio to determine the goodness of a split.

- To avoid bias of favoring attributes with large number of distinct values:
  1. Restrict test conditions to *only* binary splits
     - CART decision tree algorithm
  2. Gain Ratio: Take into account the number of outcomes produced by attribute split condition
     - Adjusts information gain by the entropy of the partitioning

$$GainRatio = \frac{\Delta_{\text{info}}}{Split_{Info}}$$

$$Split_{Info} = -\sum_{i=1}^{k} P(v_i)\log_2 P(v_i)$$

$k$ is the total number of splits

- Large number of splits make Split $_{\text{Info}}$ larger
- will reduce the Gain Ratio

# Complete Example

Initial: 2 Yes, 8 No

$$Gini = 1 - \left(\frac{2}{10}\right)^2 - \left(\frac{8}{10}\right)^2 = .32$$

| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

# Complete Example

Split on Gives Birth?

$$Gini_{GivesBirth=Yes} = 1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2 = .48$$

$$Gini_{GivesBirth=No} = 1 - \left(\frac{0}{5}\right)^2 - \left(\frac{5}{5}\right)^2 = 0$$

$$\text{Weighted Average: } \frac{5}{10} \times .48 + \frac{5}{10} \times 0 = .24$$

$$Gini_{GivesBirth} = .32 - .24 = .08$$

| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

# Complete Example

Split on 4-legged?

$$Gini_{4Legged=Yes} = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = .5$$

$$Gini_{4Legged=No} = 0$$

$$\text{Weighted Average: } \frac{4}{10} \times .5 + \frac{6}{10} \times 0 = .2$$

$$Gini_{4Legged} = .32 - .2 = .12$$

| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

# Complete Example

| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
| --- | --- | --- | --- | --- | --- |
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

Split on Hibernates?

$$Gini_{Hibernates=Yes} = 1 - \left(\frac{1}{4}\right)^2 - \left(\frac{3}{4}\right)^2 = .375$$

$$Gini_{Hibernates=No} = 1 - \left(\frac{1}{6}\right)^2 - \left(\frac{5}{6}\right)^2 = .278$$

$$\text{Weighted Average: } \frac{4}{10} \times .375 + \frac{6}{10} \times .278 = .3168$$

$$Gini_{Hibernates} = .32 - .3168 = .0032$$

# Complete Example

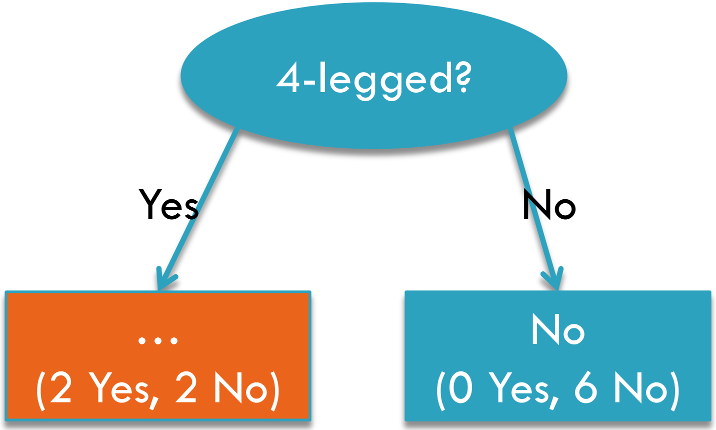
Split on Body Temperature?

$$Gini_{BodyTemperature=Warm} = .48$$
$$Gini_{BodyTemperature=Cold} = 0$$
Weighted Average: .24
$$Gini_{BodyTemperature} = .08$$

| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

# Complete Example

Splitting on 4-legged would yield the largest Gain.




$$Gini = 1-\left(\frac{2}{4}\right)^2-\left(\frac{2}{4}\right)^2 = .5$$

| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

# Complete Example

Split on Gives Birth?

$$Gini_{GivesBirth=Yes} = 1 - \left(\frac{2}{2}\right)^2 - \left(\frac{0}{2}\right)^2 = 0$$

$$Gini_{GivesBirth=No} = 0$$

Weighted Average: 0

$$Gini_{GivesBirth} = .5 - 0 = .5$$

| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

# Complete Example

Split on Hibernates?

$$Gini_{Hibernates=Yes} = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2 = .5$$

$$Gini_{Hibernates=No} = .5$$

$$\text{Weighted Average: } \frac{2}{4} \times .5 + \frac{2}{4} \times .5 = .5$$

$$Gini_{Hibernates} = .5 - .5 = 0$$

| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

# Complete Example

Split on Body Temperature?

$$Gini_{BodyTemperature=Warm} = 0$$

$$Gini_{BodyTemperature=Cold} = 0$$

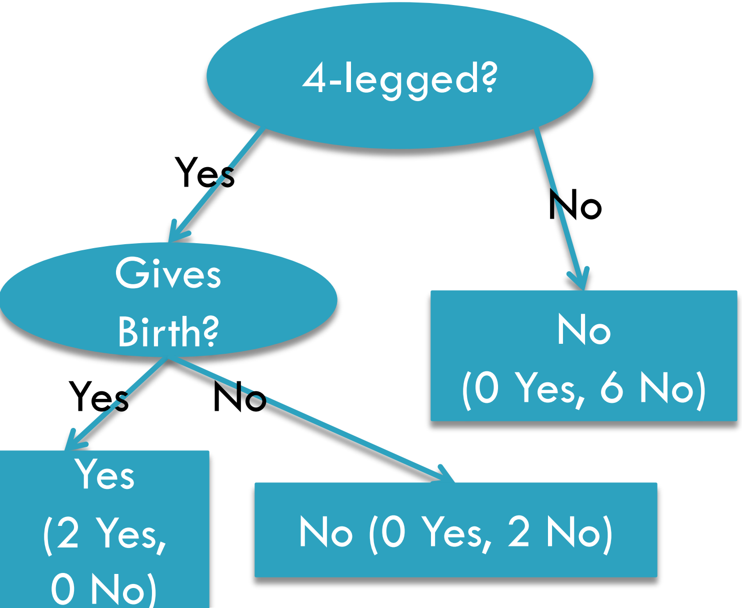
Weighted Average: 0

$$Gini_{BodyTemperature} = .5 - 0 = .5$$

| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

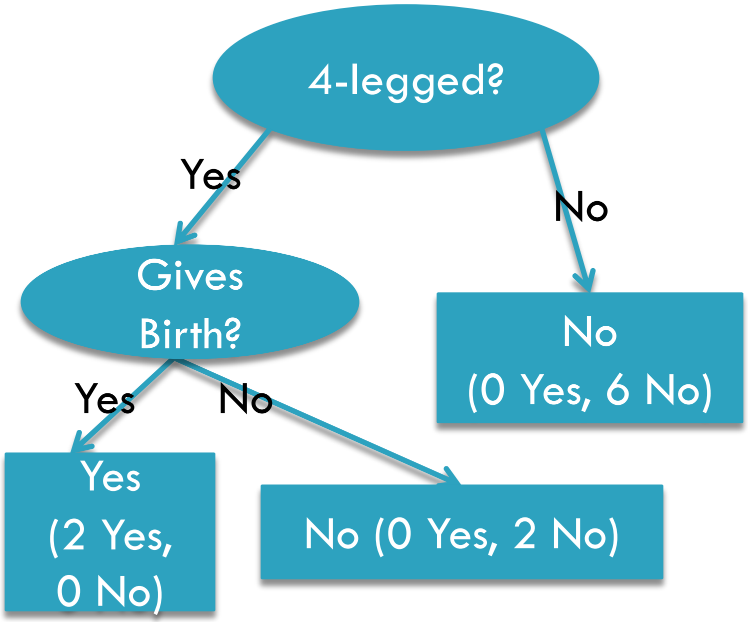
# Complete Example

Splitting on either Gives Birth or Body Temperature will fit training data perfectly.




| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

# Decision Tree Rules




- If *4-legged* And *GivesBirth* Then Yes
- If *4-legged* And Not *GivesBirth* Then No
- If Not *4-legged* Then No

# Training Set vs. Test Set

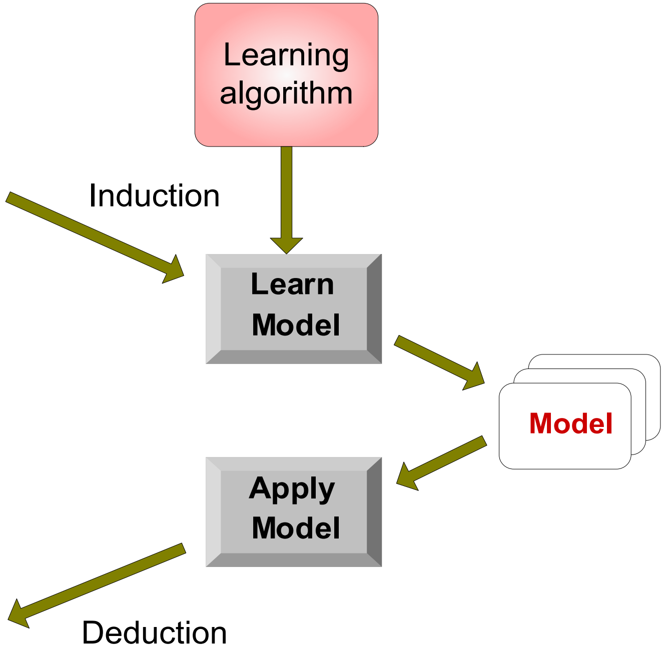
- Overall dataset is divided into:
  1. Training set – used to build model
  2. Test set – evaluates model
  3. (sometimes a Validation set is also used; more later)

| *Tid* | Attrib1 | Attrib2 | Attrib3 | Class |
|---|---|---|---|---|
| 1 | Yes | Large | 125K | No |
| 2 | No | Medium | 100K | No |
| 3 | No | Small | 70K | No |
| 4 | Yes | Medium | 120K | No |
| 5 | No | Large | 95K | Yes |
| 6 | No | Medium | 60K | No |
| 7 | Yes | Large | 220K | No |
| 8 | No | Small | 85K | Yes |
| 9 | No | Medium | 75K | No |
| 10 | No | Small | 90K | Yes |

Training Set

| *Tid* | Attrib1 | Attrib2 | Attrib3 | Class |
|---|---|---|---|---|
| 11 | No | Small | 55K | ? |
| 12 | Yes | Medium | 80K | ? |
| 13 | Yes | Large | 110K | ? |
| 14 | No | Small | 95K | ? |
| 15 | No | Large | 67K | ? |

Test Set



# Review: Model Evaluation on Test Set (Classification) – Error Rate

- <u>Error Rate:</u> proportion of mistakes that are made by applying our $\hat{f}$ model to the testing observations:

$$\frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

Observations in test set: $\{(x_1, y_1), \ldots, (x_n, y_n)\}$

$\hat{y}_i$ is the predicted class for the $i$th record

$I(y_i \neq \hat{y}_i)$ is an indicator variable: equals 1 if $y_i \neq \hat{y}_i$ and 0 if $y_i = \hat{y}_i$

# Review: Model Evaluation on Test Set (Classification) – Confusion Matrix

- <u>Confusion Matrix:</u> tabulation of counts of test records correctly and incorrectly predicted by model

| | | Predicted Class | |
|---|---|---|---|
| | | Class = 1 | Class = 0 |
| Actual Class | Class = 1 | $f_{11}$ | $f_{10}$ |
| | Class = 0 | $f_{01}$ | $f_{00}$ |

(Confusion matrix for a 2-class problem.)

# Review: Model Evaluation on Test Set (Classification) – Confusion Matrix

| | | Predicted Class | |
|---|---|---|---|
| | | *Class* = 1 | *Class* = 0 |
| Actual Class | *Class* = 1 | $f_{11}$ | $f_{10}$ |
| | *Class* = 0 | $f_{01}$ | $f_{00}$ |

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

$$\text{Error rate} = \frac{\text{Number of wrong predictions}}{\text{Total number of predictions}} = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

Most classification tasks seek models that attain the highest accuracy when applied to the test set.

# Review: Model Evaluation on Test Set (Regression) – Mean Squared Error

- <u>Mean Squared Error:</u> measuring the "quality of fit"
  - will be small if the predicted responses are very close to the true responses

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

Observations in test set: $\{(x_1,y_1), \ldots, (x_n,y_n)\}$

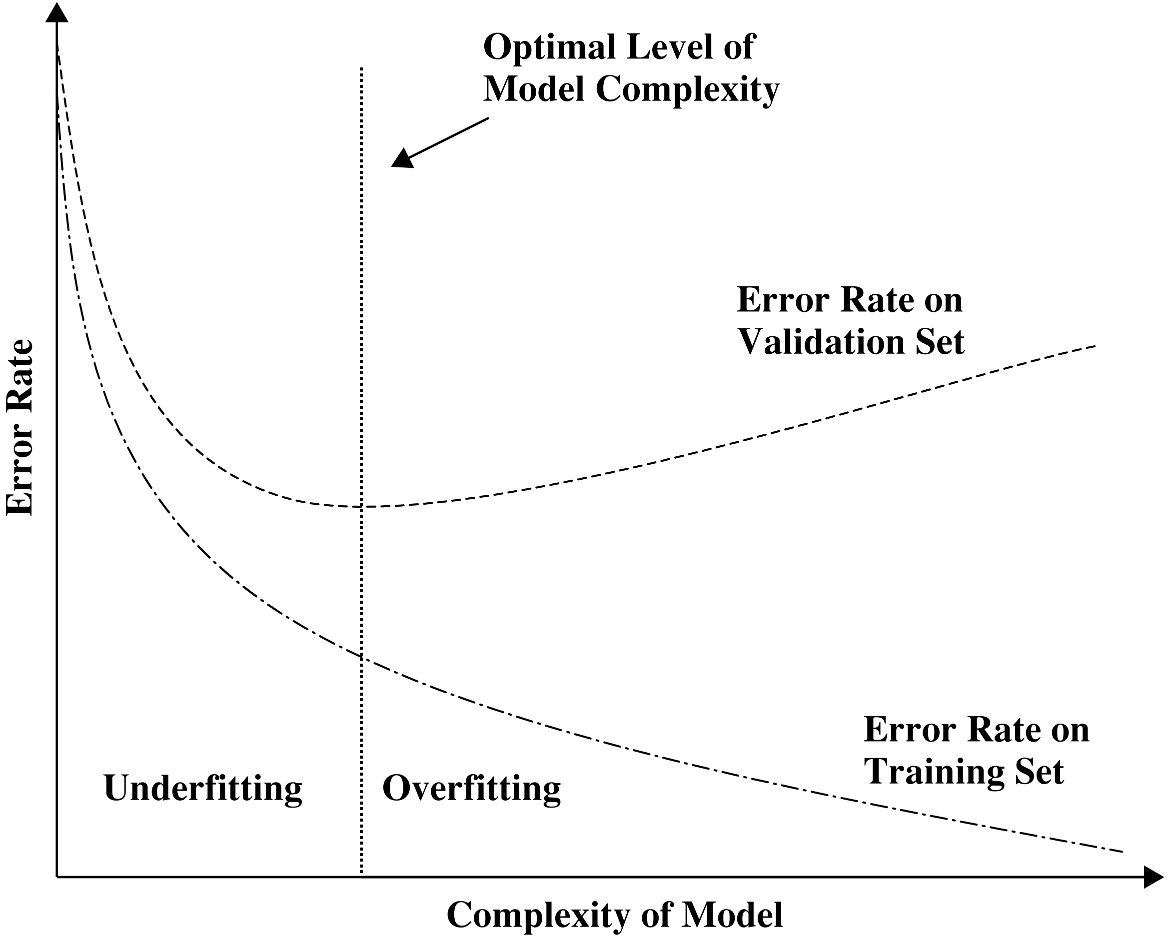
$\hat{f}(x_i)$ is the predicted value for the $i$th record

# Review: Problem

- Error rates on training set vs. testing set might be drastically different.
- No guarantee that the model with the smallest training error rate will have the smallest testing error rate

# Review: Overfitting

- Overfitting: occurs when model "memorizes" the training set data
  - very low error rate on training data
  - yet, high error rate on test data
- Model *does not generalize* to the overall problem
- This is bad! We wish to avoid overfitting.

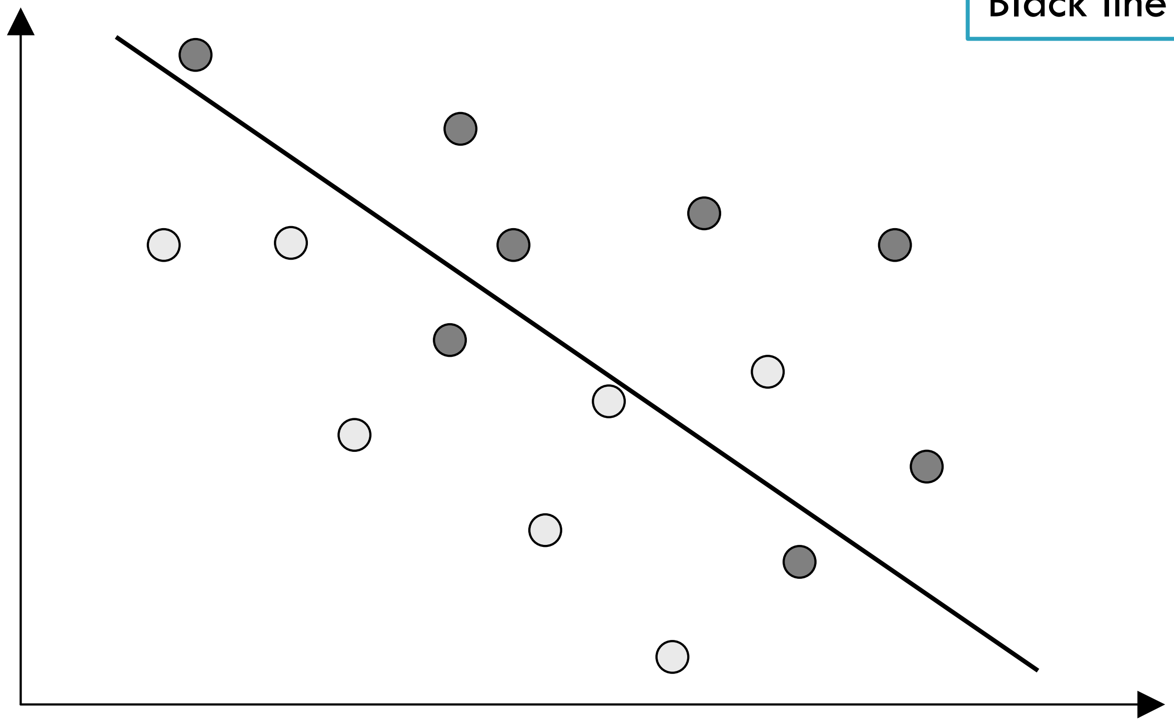

Optimal Level of
Model Complexity
Error Rate on
Validation Set
Error Rate
Error Rate on
Training Set
Underfitting
Overfitting
Complexity of Model

# Review: Bias and Variance

- Bias: the error introduced by modeling a real-life problem (usually extremely complicated) by a much simpler problem
  - The more flexible (complex) a method is, the less bias it will generally have.
- Variance: how much the learned model will change if the training set was different
  - *Does changing a few observations in the training set, dramatically affect the model?*
  - Generally, the more flexible (complex) a method is, the more variance it has.

*Example:* we wish to build a model that separates the dark-colored points from the light-colored points.

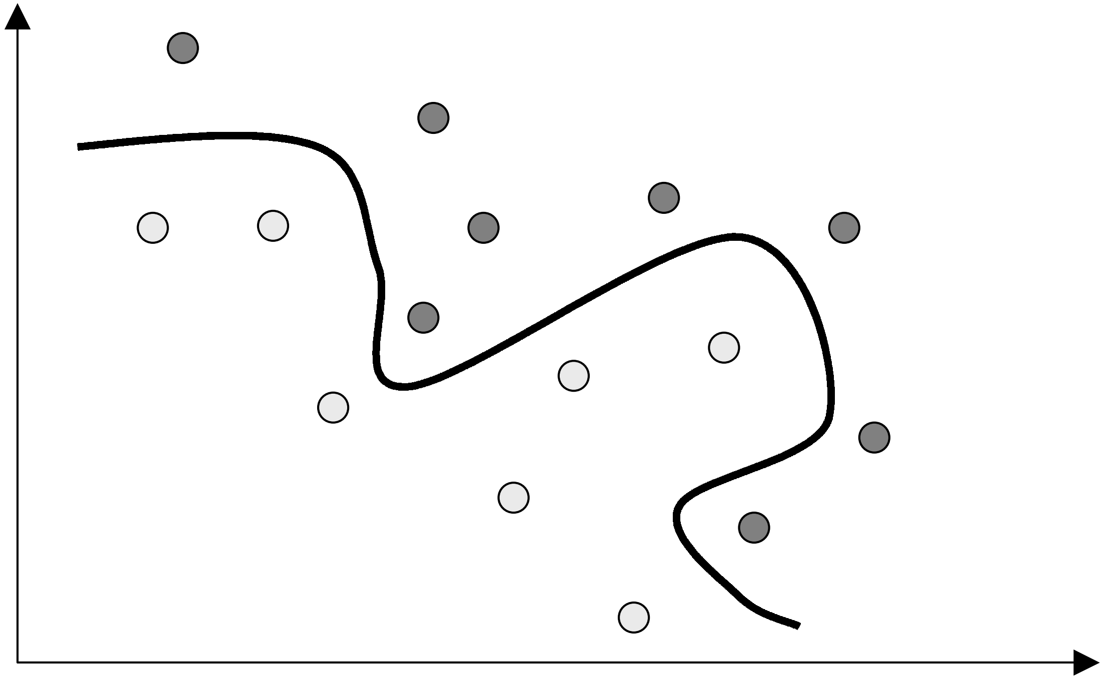
Black line is *simple, linear* model



Currently, some classification error

- Low variance
- Bias present

More complex model (curvy line instead of linear)

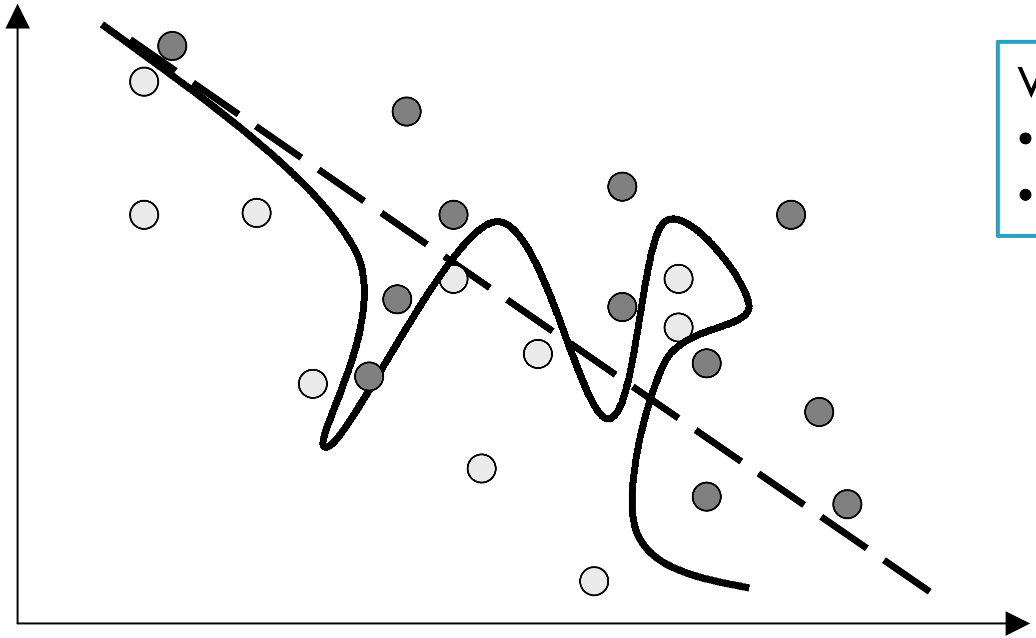


Zero classification error for these data points

- No linear model bias
- Higher Variance?

More data has been added.

Re-train both models (linear line, and curvy line) in order to minimize error rate



Variance:

- Linear model doesn't change much
- Curvy line significantly changes

Which model is better?

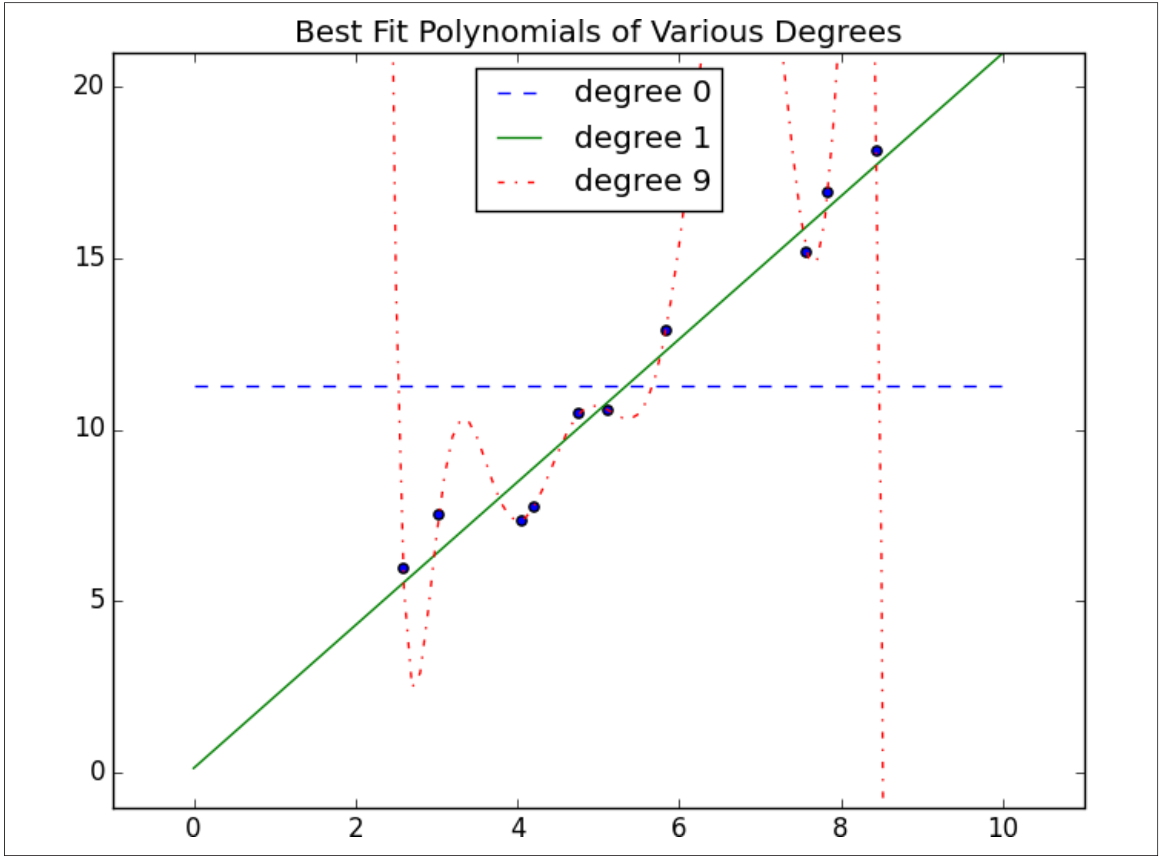
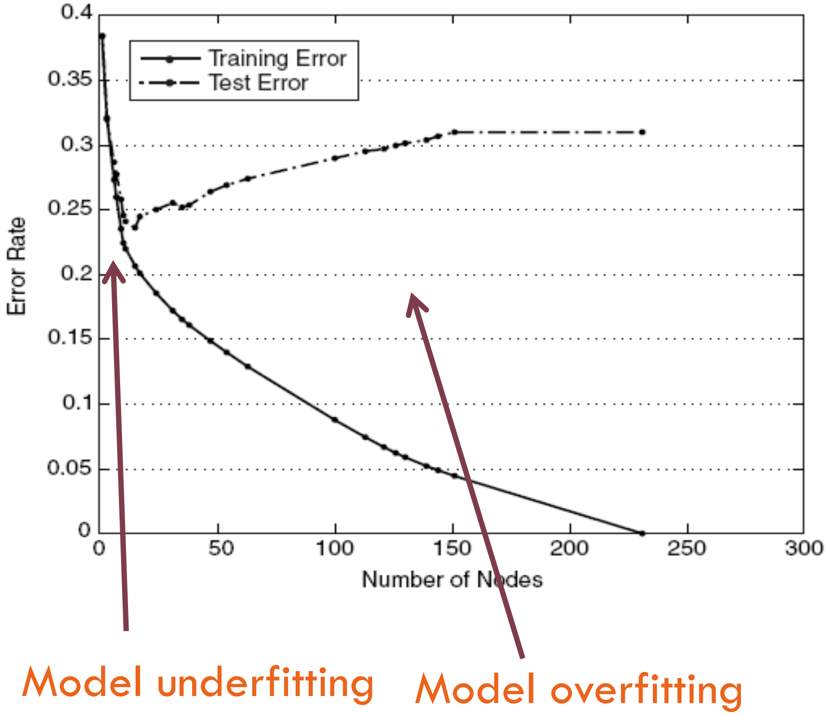



*Figure 11-1. Overfitting and underfitting*

# Model Overfitting

- Errors committed by a classification model are generally divided into:
  1. Training errors: misclassification on training set records
  2. Generalization errors (testing errors): errors made on testing set / previously unseen instances
- Good model has low training error and low generalization error.
- Overfitting: model has low training error rate, but high generalization errors

# Model Underfitting and Overfitting



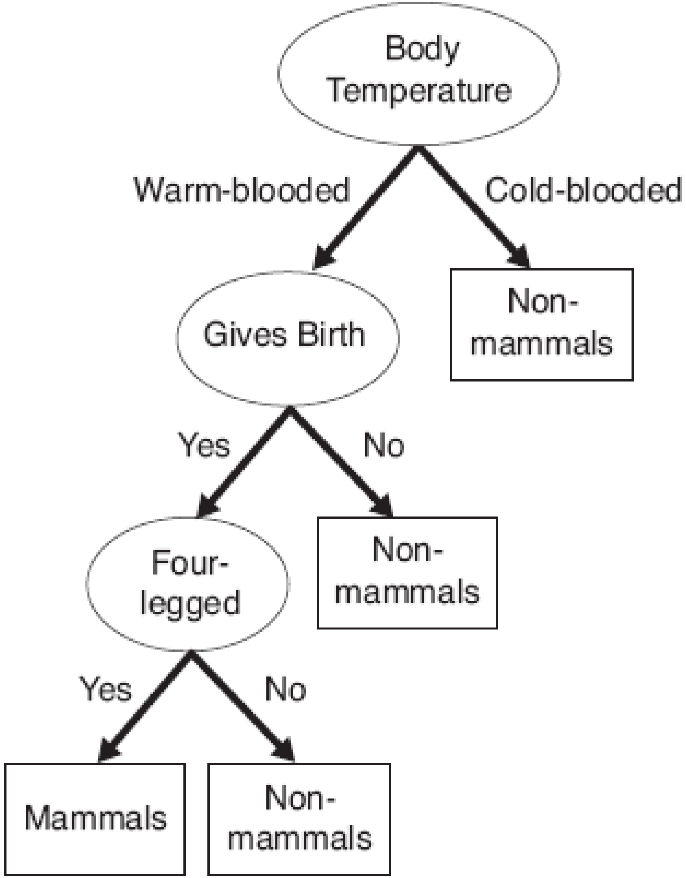

- *When tree is small:*
  - Underfitting
  - Large training error rate
  - Large testing error rate
  - Structure of data isn't yet learned
- *When tree gets too large:*
  - Beware of overfitting
  - Training error rate decreases while testing error rate increases
  - Tree is too complex
  - Tree "almost perfectly fits" training data, but doesn't generalize to testing examples

# Reasons for Overfitting

1. Presence of noise
2. Lack of representative samples

## Training Set

| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Porcupine | Warm | Yes | Yes | Yes | Yes |
| Cat | Warm | Yes | Yes | No | Yes |
| Bat | Warm | Yes | No | Yes | No |
| Whale | Warm | Yes | No | No | No |
| Salamander | Cold | No | Yes | Yes | No |
| Komodo Dragon | Cold | No | Yes | No | No |
| Python | Cold | No | No | Yes | No |
| Salmon | Cold | No | No | No | No |
| Eagle | Warm | No | No | No | No |
| Guppy | Cold | Yes | No | No | No |

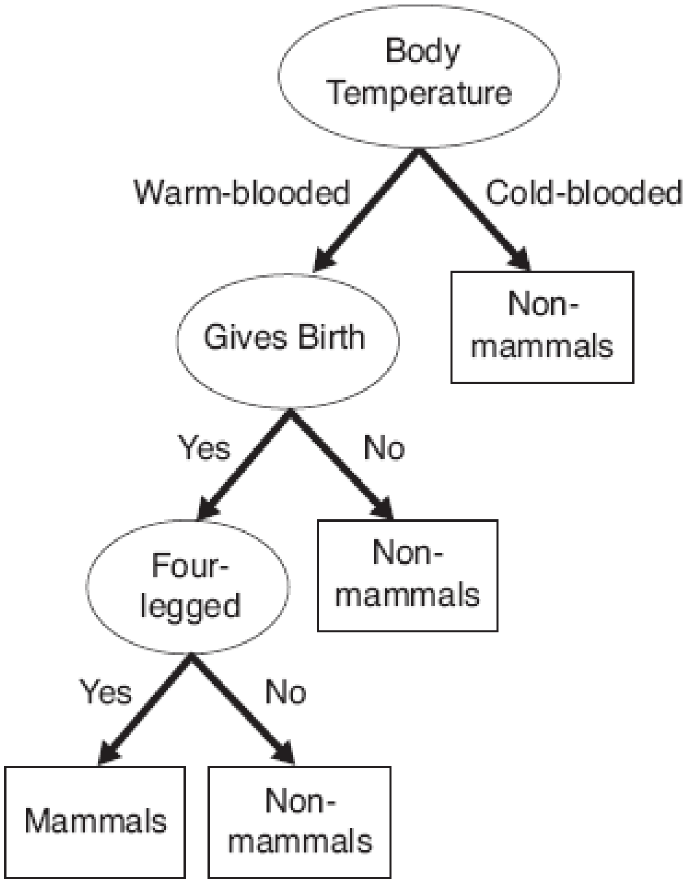



Two training records are mislabeled.

Tree perfectly fits training data.

## Testing Set

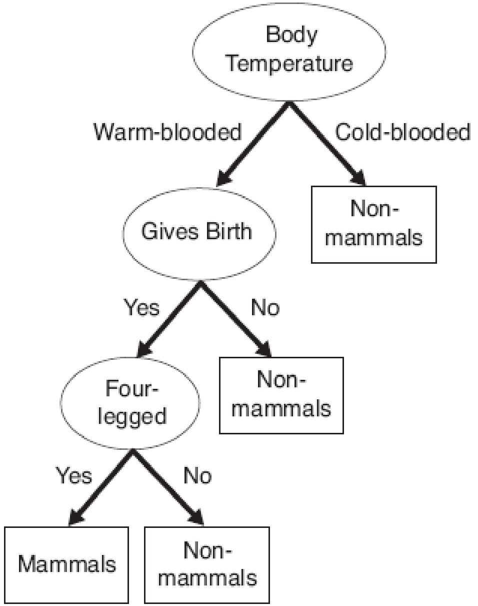
| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Human | Warm | Yes | No | No | Yes |
| Pigeon | Warm | No | No | No | No |
| Elephant | Warm | Yes | Yes | No | Yes |
| Leopard Shark | Cold | Yes | No | No | No |
| Turtle | Cold | No | Yes | No | No |
| Penguin | Cold | No | No | No | No |
| Eel | Cold | No | No | No | No |
| Dolphin | Warm | Yes | No | No | Yes |
| Spiny Anteater | Warm | No | Yes | Yes | Yes |
| Gila Monster | Cold | No | Yes | Yes | No |




Test error rate: 30%

## Testing Set

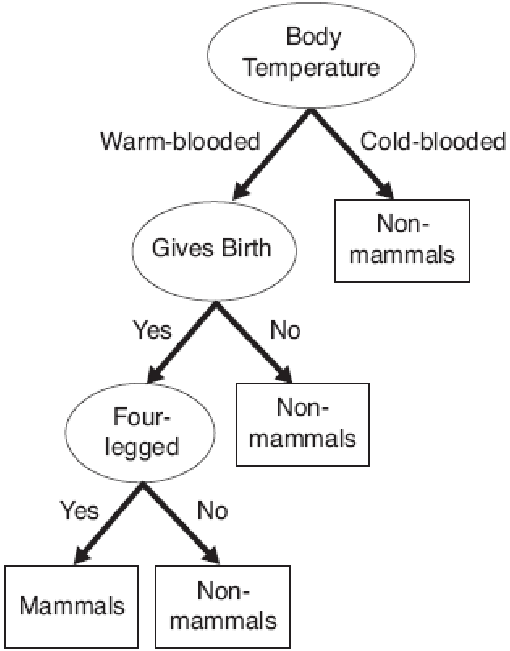
| Name | Body Temp. | Gives Birth | 4 legged | Hibernates | Class (Mammal?) |
|---|---|---|---|---|---|
| Human | Warm | Yes | No | No | Yes |
| Pigeon | Warm | No | No | No | No |
| Elephant | Warm | Yes | Yes | No | Yes |
| Leopard Shark | Cold | Yes | No | No | No |
| Turtle | Cold | No | Yes | No | No |
| Penguin | Cold | No | No | No | No |
| Eel | Cold | No | No | No | No |
| Dolphin | Warm | Yes | No | No | Yes |
| Spiny Anteater | Warm | No | Yes | Yes | Yes |
| Gila Monster | Cold | No | Yes | Yes | No |



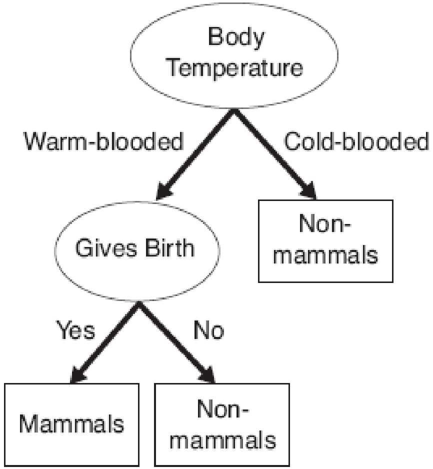

Test error rate: 30%

Reasons for misclassifications:

- Mislabeled records in training data
- "Exceptional case"
  - Unavoidable
  - Minimal error rate achievable by any classifier




(a) Model M1




(b) Model M2

- Training error rate: 0%
- Test error rate: 30%
- *overfitting*

- Training error rate: 20%
- Test error rate: 10%

# Overfitting and Decision Trees

- The likelihood of overfitting occurring increases as a tree gets deeper
  - the resulting classifications are based on smaller subsets of the full training dataset
- Overfitting involves splitting the data on an irrelevant feature.

# Pruning: Handling Overfitting in Decision Trees

- Tree pruning identifies and removes subtrees within a decision tree that are likely to be due to noise and sample variance in the training set used to induce it.
- Pruning will result in decision trees being created that are *not consistent* with the training set used to build them.
- But we are more interested in created prediction models that generalize well to new data!

1. Pre-pruning (Early Stopping)
2. Post-pruning

# Pre-pruning Techniques

1. Stop creating subtrees when the number of instances in a partition falls below a threshold
2. Information gain measured at a node is not deemed to be sufficient to make partitioning the data worthwhile
3. Depth of the tree goes beyond a predefined limit
4. … other more advanced approaches

Benefits: Computationally efficient; works well for small datasets.
Downsides: Stopping too early will fail to create the most effective trees.

# Post-pruning

1. Decision tree initially grown to its maximum size
2. Then examine each branch
3. Branches that are deemed likely to be due to overfitting are pruned.

- Post-pruning tends to give better results than prepruning
- *Which is faster?*
  - Post-pruning is more computationally expensive than prepruning because entire tree is grown

# Post-pruning Techniques

1. Reduced Error Pruning
2. Cost Complexity Pruning

# Reduced Error Pruning

- Starting at the leaves, each node is replaced with its most popular class.
- If the accuracy is not affected, then the change is kept.
  - Evaluate accuracy on a validation set
  - Set aside some of the training set as a validation set
- *Advantages*: simplicity and speed

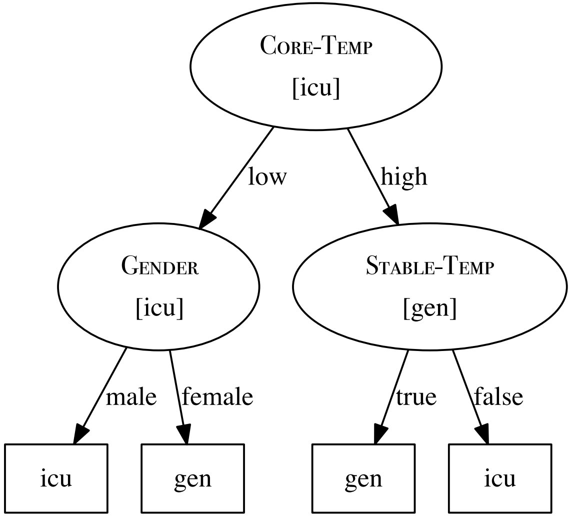
# Cost Complexity Pruning

- Nonnegative tuning parameter: α
  - “Penalizing cost” / “complexity parameter”
- Will look at different pruned subtrees and compare their performance on a test sample
- α determines the trade-off between misclassification error and the model complexity
  - Small α: penalty for larger tree is small
  - Larger α: smaller trees preferred depending on # of errors

# Post-Pruning Example

## Example validation set:

| ID | CORE-TEMP | STABLE-TEMP | GENDER | DECISION |
|---|---|---|---|---|
| 1 | high | true | male | gen |
| 2 | low | true | female | icu |
| 3 | high | false | female | icu |
| 4 | high | false | male | icu |
| 5 | low | false | female | icu |
| 6 | low | true | male | icu |

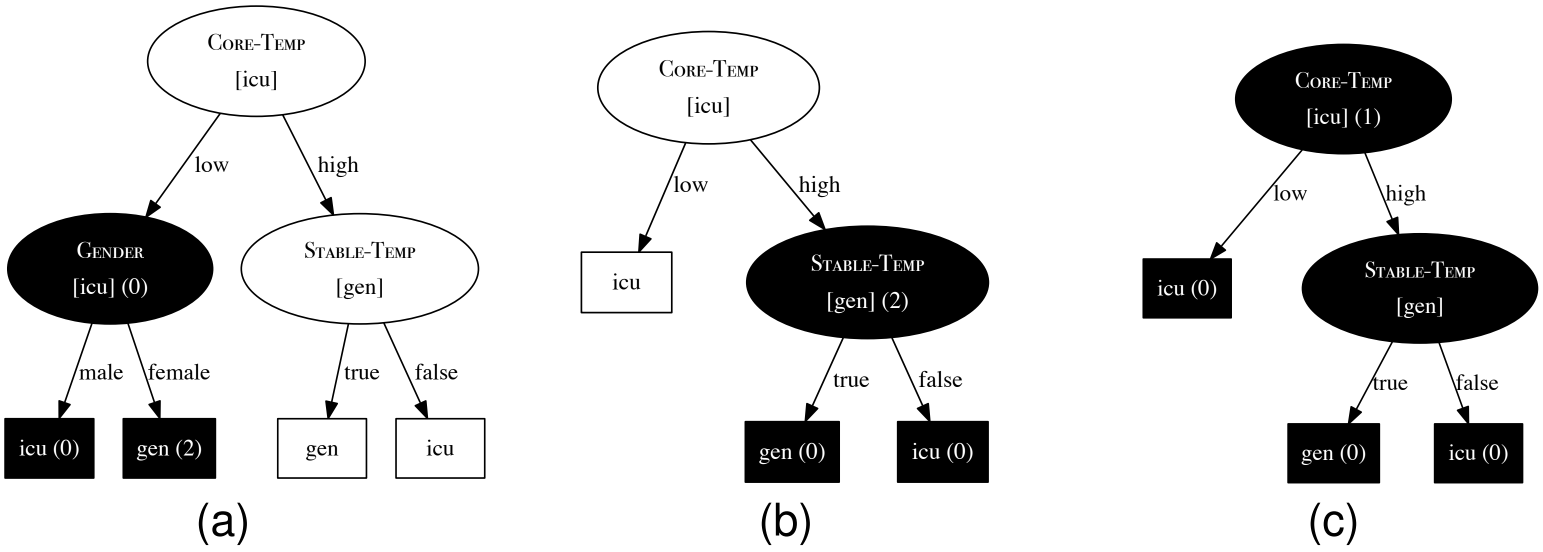



Induced decision tree from training data

- Need to prune?

# Post-Pruning Example

| ID | CORE-TEMP | STABLE-TEMP | GENDER | DECISION |
|---|---|---|---|---|
| 1 | high | true | male | gen |
| 2 | low | true | female | icu |
| 3 | high | false | female | icu |
| 4 | high | false | male | icu |
| 5 | low | false | female | icu |
| 6 | low | true | male | icu |



# Occam's Razor

*General Principle (Occam's Razor)*: given two models with same generalization (testing) errors, the *simpler* model is preferred over the more *complex* model

- Additional components in a more complex model have greater chance at being fitted purely by chance

Problem solving principle by philosopher William of Ockham (1287-1347)

# Advantages of Pruning

1. Smaller trees are easier to interpret
2. Increased generalization accuracy.

# Regression Trees

- Target Attribute:
  - Decision (Classification) Trees: qualitative
  - Regression Trees: continuous
- *Decision trees:* reduce the entropy in each subtree
- *Regression trees:* reduce the variance in each subtree
  - Idea: adapt ID3 algorithm measure of *Information Gain* to use variance rather than node impurity

# Regression Tree Splits

## Classification Trees

- Gain: "goodness of the split"
- larger gain => better split (better test condition)

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

- $I(n)$ = impurity measure at node $n$
- $k$ = number of attribute values
- $N(n)$ = total number of records at child node $n$
- $N$ = total number of records at parent node

## Regression Trees

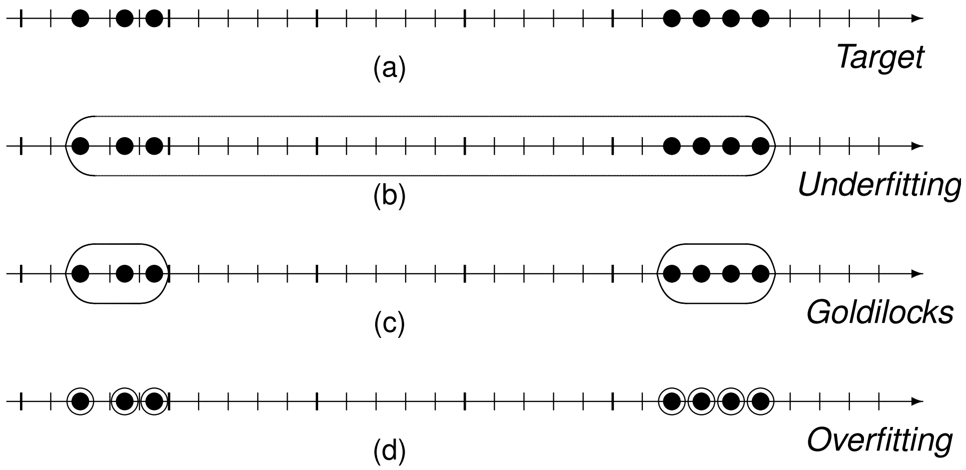
- Impurity (variance) at a node:

$$var(t, \mathcal{D}) = \frac{\sum_{i=1}^{n} (t_i - \bar{t})^2}{n - 1}$$

- Select feature to split on that minimizes the weighted variance across all resulting partitions:

$$\mathbf{d}[best] = \underset{d \in \mathbf{d}}{\operatorname{argmin}} \sum_{l \in levels(d)} \frac{|\mathcal{D}_{d=l}|}{|\mathcal{D}|} \times var(t, \mathcal{D}_{d=l})$$

# Need to watch out for Overfitting



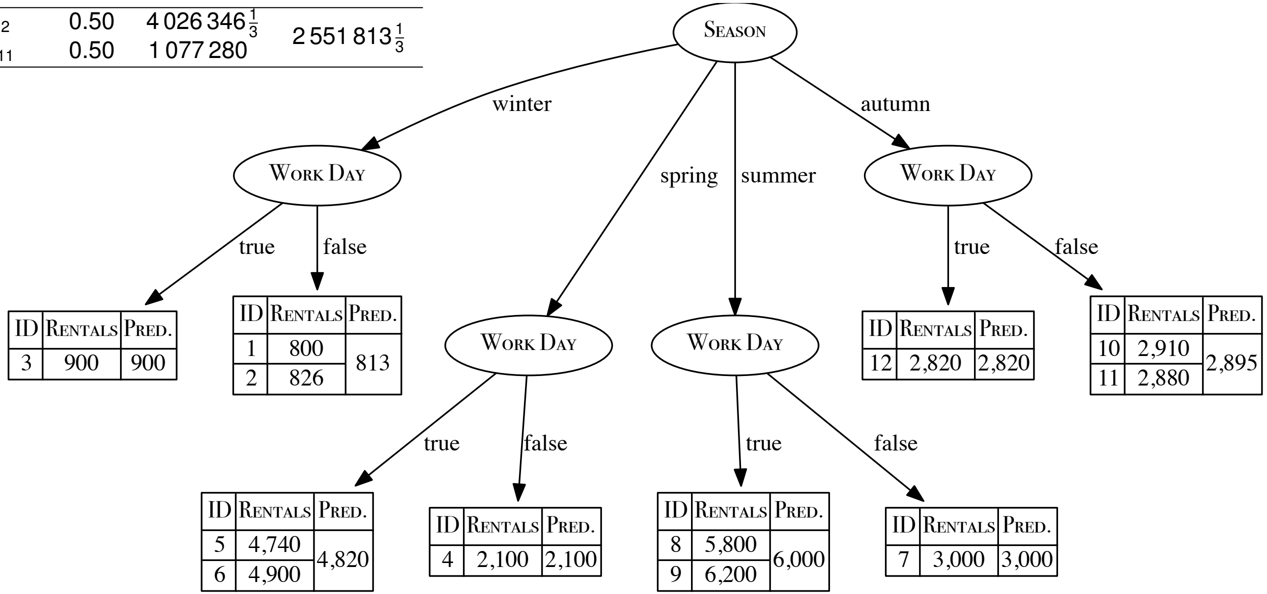

- Want to avoid overfitting:
  - Early stopping criterion
  - Stop partitioning the dataset if the number of training instances is less than some threshold
  - (5% of the dataset)

# Example

| ID | Season | Work Day | Rentals |
|---|---|---|---|
| 1 | winter | false | 800 |
| 2 | winter | false | 826 |
| 3 | winter | true | 900 |
| 4 | spring | false | 2 100 |
| 5 | spring | true | 4 740 |
| 6 | spring | true | 4 900 |

| ID | Season | Work Day | Rentals |
|---|---|---|---|
| 7 | summer | false | 3 000 |
| 8 | summer | true | 5 800 |
| 9 | summer | true | 6 200 |
| 10 | autumn | false | 2 910 |
| 11 | autumn | false | 2 880 |
| 12 | autumn | true | 2 820 |

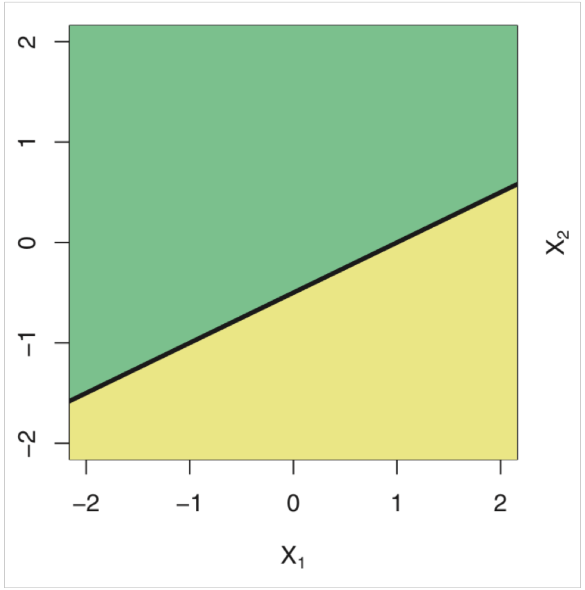
| Split by Feature | Level | Part. | Instances | $\frac{\lvert\mathcal{D}_{d=l}\rvert}{\lvert\mathcal{D}\rvert}$ | $var(t, \mathcal{D})$ | Weighted Variance |
|---|---|---|---|---|---|---|
| Season | 'winter' | $\mathcal{D}_1$ | $\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3$ | 0.25 | 2 692 | $1\,379\,331\frac{1}{3}$ |
| | 'spring' | $\mathcal{D}_2$ | $\mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6$ | 0.25 | $2\,472\,533\frac{1}{3}$ | |
| | 'summer' | $\mathcal{D}_3$ | $\mathbf{d}_7, \mathbf{d}_8, \mathbf{d}_9$ | 0.25 | 3 040 000 | |
| | 'autumn' | $\mathcal{D}_4$ | $\mathbf{d}_{10}, \mathbf{d}_{11}, \mathbf{d}_{12}$ | 0.25 | 2 100 | |
| Work Day | 'true' | $\mathcal{D}_5$ | $\mathbf{d}_3, \mathbf{d}_5, \mathbf{d}_6, \mathbf{d}_8, \mathbf{d}_9, \mathbf{d}_{12}$ | 0.50 | $4\,026\,346\frac{1}{3}$ | $2\,551\,813\frac{1}{3}$ |
| | 'false' | $\mathcal{D}_6$ | $\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_7, \mathbf{d}_{10}, \mathbf{d}_{11}$ | 0.50 | 1 077 280 | |

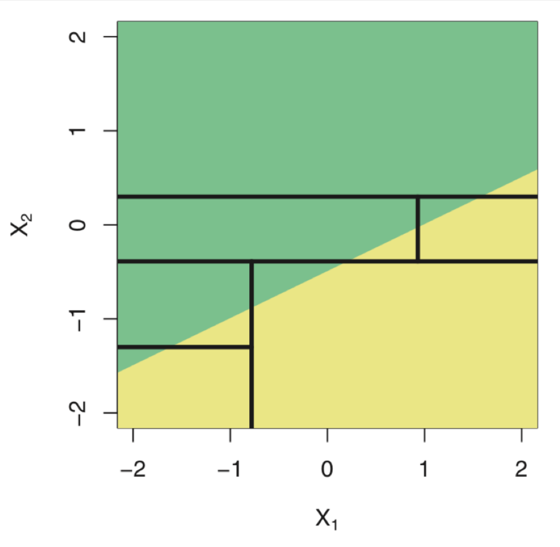


# Advantages and Disadvantages of Trees (Compared to Linear Models)

Two-classes: {green, blue}

Decision boundary: linear



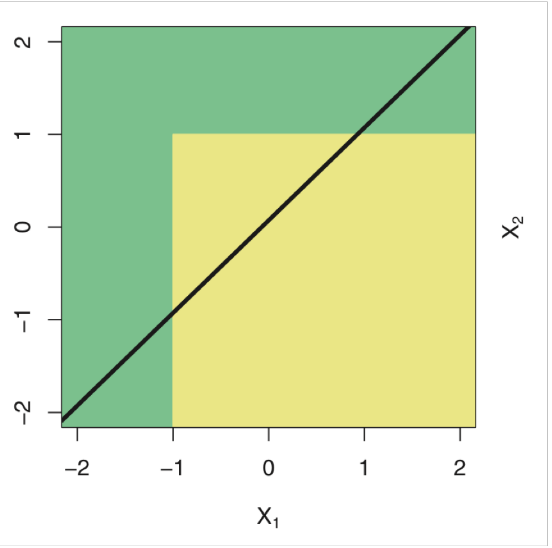

Linear model can perfectly separate the two regions.

- What about a decision tree?

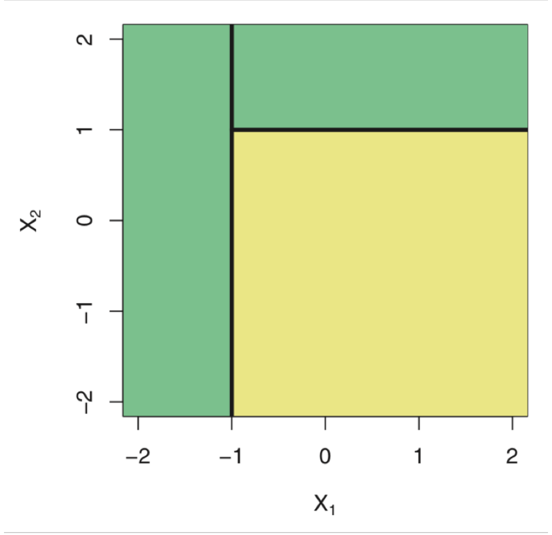
Decision tree cannot separate regions.

# Advantages and Disadvantages of Trees (Compared to Linear Models)

Two-classes: {green, blue}




Decision boundary: nonlinear

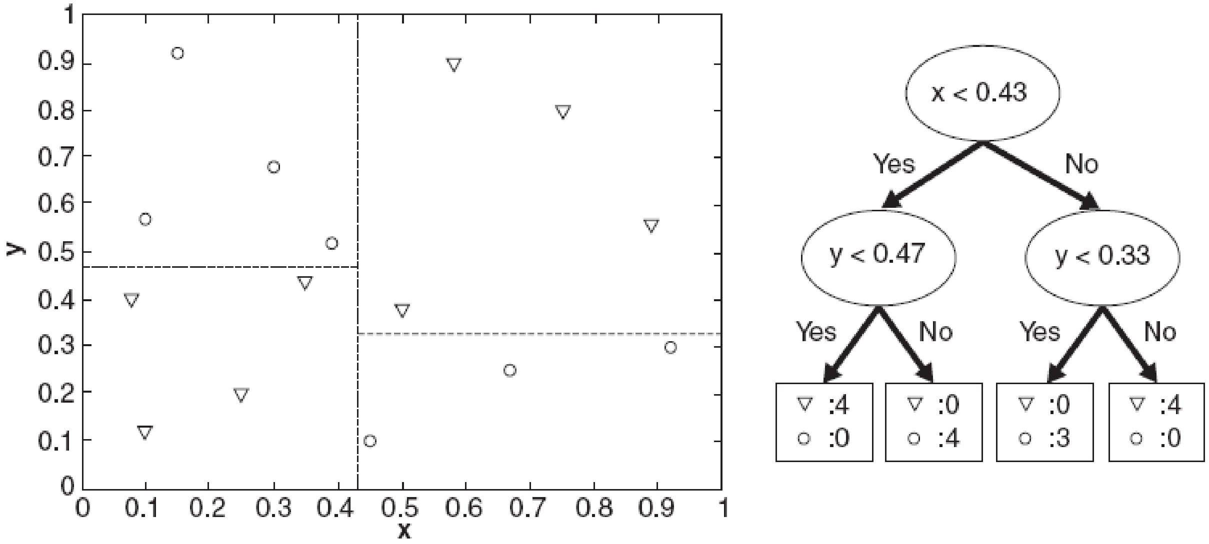



Linear model cannot perfectly separate the two regions.

- What about a decision tree?

A Decision tree can!

# Decision Tree can Separate Nonlinear Regions



# Advantages and Disadvantages of Trees

| Advantages | Disadvantages |
| --- | --- |

**Advantages**

- Trees are very easy to explain
  - Easier to explain than linear regression
- Trees can be displayed graphically and interpreted by a non-expert
- Decision trees may more closely mirror human decision-making
- Trees can easily handle qualitative predictors
  - No dummy variables

**Disadvantages**

- Trees usually do not have same level of predictive accuracy as other data mining algorithms

But, predictive performance of decision trees can be improved by aggregating trees.

- *Techniques*: bagging, boosting, random forests

# Decision Tree Advantages

- Inexpensive to construct
- Extremely fast at classifying unknown records
  - O($d$) where $d$ is the depth of the tree
- Presence of redundant attributes does not adversely affect the accuracy of decision trees
  - One of the two redundant attributes will not be used for splitting once the other attribute is chosen
- Nonparametric approach
  - Does not require any prior assumptions regarding probability distributions, means, variances, etc.

# References


- *Fundamentals of Machine Learning for Predictive Data Analytics*, 1st Edition, Kelleher et al.
- *Data Science from Scratch*, 1st Edition, Grus
- *Data Mining and Business Analytics in R*, 1st edition, Ledolter
- *An Introduction to Statistical Learning*, 1st edition, James et al.
- *Discovering Knowledge in Data*, 2nd edition, Larose et al.
- *Introduction to Data Mining*, 1st edition, Tam et al.