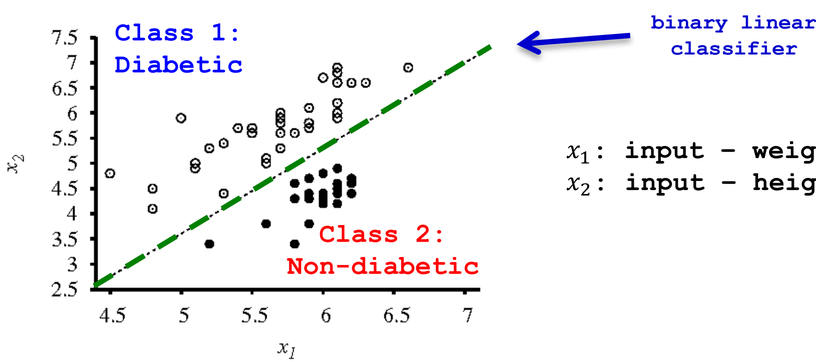
- Module 8 -
Decision Trees

Outline

- Classification
- Decision Trees

Supervised Learning: Classification

- **Classification** is another subcategory of supervised learning where the goal is the prediction of categorical labels. In classification, given a number of features, p, and a categorical outcome, the objective is to find a relationship (a function $f\colon \mathbb{R}^p \to \{1, \ldots, C\}$) to predict which of C **classes** (or categories) a sample belongs to



x_1: input – weight
x_2: input – height

Recap

Supervised Learning: Classification Applications

- Classification Applications:
  - Spam filtering
    - a binary classification problem
    - Input: email subject and message content
      (extracting keywords to form "bag of words",
      i.e., number of occurrences of 'buy', 'viagra', etc.)
    - Output: spam or non-spam
  - Movie genre classification
    - a multiclass classification problem (E.g., a genre is a class)
    - Input: plot summary
      (extracting keywords from the summary to form "bag of words",
      i.e., number of occurrences of 'love', 'laugh', etc.)
    - Output: movie genre (romance, comedy, thriller, etc.)
  - Object recognition
    - a multiclass classification problem (can be lots of classes)
    - Input: image (pixel RGB values)
    - Output: object class (car, traffic light, motorcycle, pedestrian, bicycle, etc.)

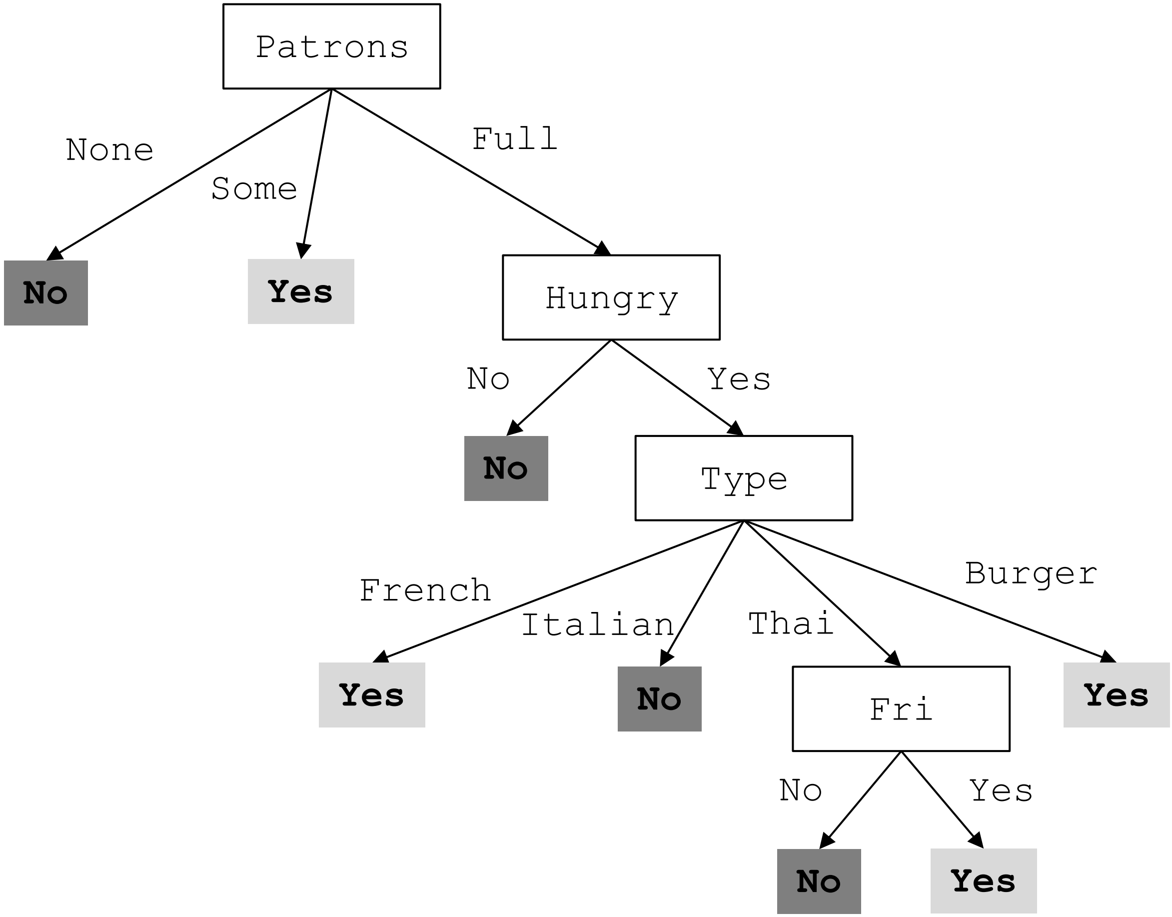
Decision Trees (1)

- **Decision tree** induction is one of the simplest and yet most successful forms of machine learning
- A decision tree represents a **function** that takes as input a vector of features and returns a decision (single output value)
  - the input and output values can be continuous or discrete
    - continuous output: **regression trees**
    - discrete output: **classification trees**

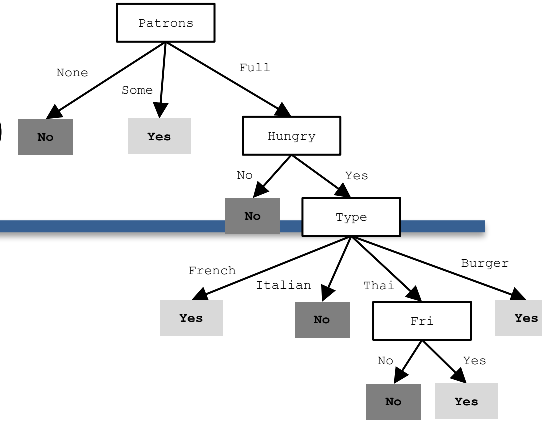
➔ For this course, focus on classification trees where input features have discrete values and the output has exactly two (2) possible values, i.e., a positive example or a false example (**binary classification**)

Decision Trees: Example (2)

- **Example:** Decision Tree



Decision Trees (3)



- A decision tree reaches its decision by performing a sequence of tests where
  - each **internal node** (white box in the example) corresponds to one of the input features, X_j for $j = 1, \dots, p$
  - the **branches** (arrow) from the node are labeled with the possible values of the feature, v_k for $k = 1, \dots, K$
  - each **leaf node** (shaded box: **light grey** shading $y_i =$ Yes, **dark grey** shading $y_i =$ No) specifies a value to be returned by the function, i.e., classification output

- After a decision tree has been induced, a new sample is classified by
  - starting at the root node of the tree
  - testing the feature specified by this node
  - moving down the branch corresponding to the value of the feature
  - ➔ process is then repeated for the subtree rooted at the new node until a leaf node is reached

Decision Trees (4)

- Advantages of decision trees:
  - easy to explain (compared to linear regression)
  - can be displayed graphically and easier to interpret
  - closely mirror human decision-making process (i.e., more intuitive)

- Limitations of decision trees:
  - can be non-robust: a small change in the training data can lead to a significant change in the tree and consequently the prediction outcomes, i.e., high variance
  - handling of continuous input features require finding the split points that give accurate classification

Decision Trees: Example (5)

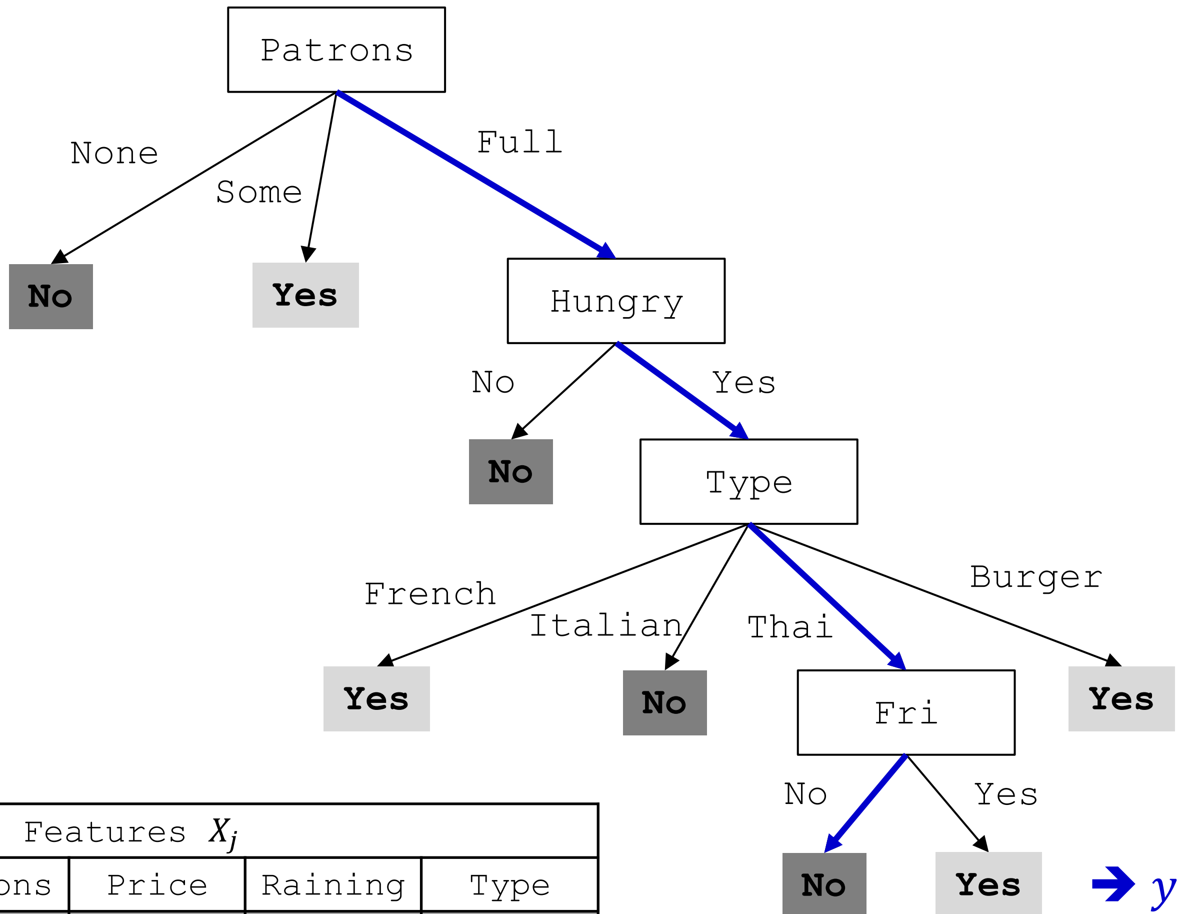
- **Example**: Classification Problem

  Given the following input features and possible values of each feature:
  - X_1: **Friday = {Yes, No}**
  - X_2: **Hungry = {Yes, No}**
  - X_3: **Patrons = {None, Some, Full}**
    (indicates how many people are in the restaurant)
  - X_4: **Price = {$, $$, $$$}**
    (indicates the restaurant's price range)
  - X_5: **Raining = {Yes, No}**
    (indicates whether it is raining outside)
  - X_6: **Type = {French, Italian, Thai, Burger}**
    (the kind of restaurant)

  Question: Determine if the person, **{No, Yes, Full, $, No, Thai}**, will wait, y ={Yes, No}, for a table at a given restaurant

Decision Trees: Example (6)

- **Solution:** Prediction using a decision tree



Input Features X_j					
Fri	Hungry	Patrons	Price	Raining	Type
No	**Yes**	**Full**	**$**	**No**	**Thai**

➔ $y = \text{No}$

Decision Trees (7)

- <u>Steps</u> to use a <u>decision tree</u> for <u>prediction</u>:
  - Step 1: **Induce:** <u>construct</u> a <u>decision tree</u> using <u>training samples</u>
  - Step 2: **Prediction:** for a <u>new sample</u> x, its <u>predicted classification output</u> $\hat{y}$, is given by the <u>value</u> corresponding to the <u>leaf node</u>

Decision Trees: Induction (8)

- The goal of decision tree induction is to construct a decision tree that is consistent with training samples and is as small as possible
  - however, finding the smallest consistent tree is generally computationally infeasible (i.e., intractable)

- A **greedy divide-and-conquer** approach is used
  - greedy: always test the most important feature first ➔ one that makes the most difference to the classification of a sample
  - divide-and-conquer: divide the problem into smaller subproblems that can then be solved recursively

Decision Trees: Example - Induction (9)

Example: A training set of 12 samples is shown below

Sample	Input Features X_j						**WillWait** y_i
	Fri	**Hungry**	**Patrons**	**Price**	**Raining**	**Type**	
x_1	No	Yes	Some	$$$	No	French	$y_1 =$ Yes
x_2	No	Yes	Full	$	No	Thai	$y_2 =$ No
x_3	No	No	Some	$	No	Burger	$y_3 =$ Yes
x_4	Yes	Yes	Full	$	Yes	Thai	$y_4 =$ Yes
x_5	Yes	No	Full	$$$	No	French	$y_5 =$ No
x_6	No	Yes	Some	$$	Yes	Italian	$y_6 =$ Yes
x_7	No	No	None	$	Yes	Burger	$y_7 =$ No
x_8	No	Yes	Some	$$	Yes	Thai	$y_8 =$ Yes
x_9	Yes	No	Full	$	Yes	Burger	$y_9 =$ No
x_{10}	Yes	Yes	Full	$$$	No	Italian	$y_{10} =$ No
x_{11}	No	No	None	$	No	Thai	$y_{11} =$ No
x_{12}	Yes	Yes	Full	$	No	Burger	$y_{12} =$ Yes

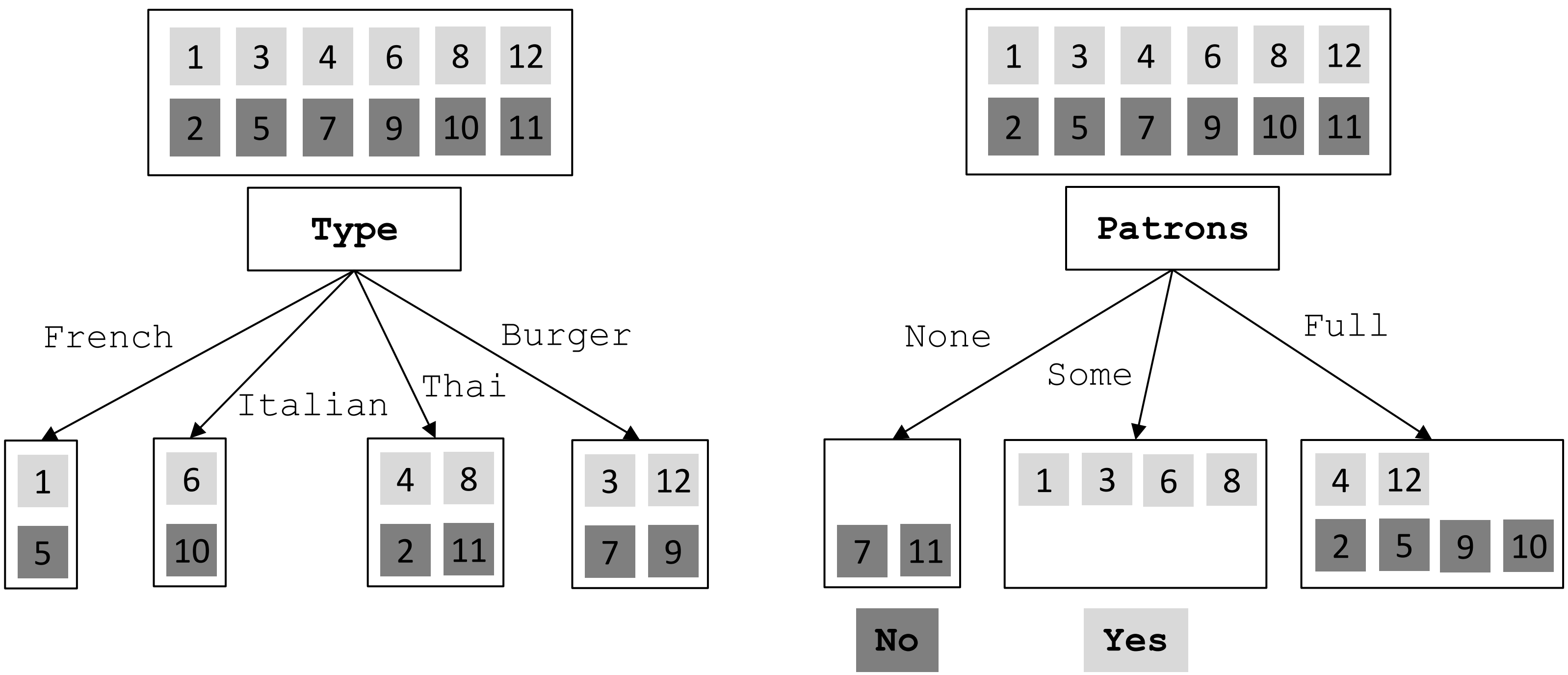
Legend: **No** Dark-colored **Yes** Light-colored

Decision Trees: Example - Induction (10)

Greedy Approach:

Determine the most important feature from

Fri, Hungry, Patrons, Price, Raining, Type



Decision Trees: Most Important Feature (11)

- The greedy approach is designed to approximately minimize the depth of the final tree
- The idea is to pick the feature that goes as far as possible towards providing an exact classification of the samples
- A perfect feature divides the samples into sets, each of which are all positive or all negative and thus will be leaves of the tree

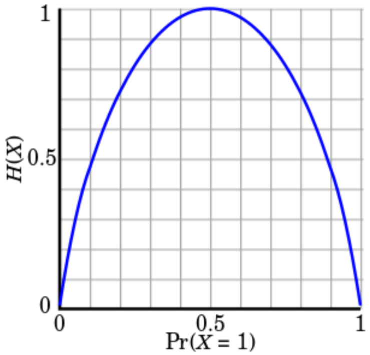
  ➔ However, features may not be perfect – use **information gain**, which is defined in terms of **entropy**

Decision Trees: Most Important Feature (12)

- The entropy of a random variable X with values v_k, each with probability $\Pr(X = v_k)$, is defined as

$$H(X) = -\sum_{k=1}^{K} \Pr(X = v_k) \log_2 \Pr(X = v_k)$$

is a measure of the uncertainty of a random variable (where acquisition of information corresponds to a reduction in entropy)



Examples:

– A coin that always comes up heads – $\Pr(X = 1) = 1$, $\Pr(X = 0) = 0$ – (where $X = 1$ represents a result of heads), has no uncertainty and thus its entropy is defined as $H(X) = -(1 \log_2 1 + 0 \log_2 0) = 0$

– A fair coin is equally likely to come up heads or tails – $\Pr(X = 1) = 0.5$, $\Pr(X = 0) = 0.5$ – has maximum uncertainty and thus its entropy is defined as $H(X) = -(0.5 \log_2 0.5 + 0.5 \log_2 0.5) = 1$

Decision Trees: Most Important Feature (13)

- Using entropy, the steps to induce the decision tree:
  1. Compute the entropy H_m for the current node, m
  2. For every feature X_j (X_j has K distinct values, $v_1, v_2, \ldots, v_K$)
     a) Compute the entropy, H_k, for each value v_k of feature X_j

$$H_k = -\sum_{c=1}^{C} p_{k,c} \log_2 p_{k,c}$$

Indicator function:
$I(condition) = \begin{cases} 1, & \text{if } condition = true \\ 0, & \text{otherwise} \end{cases}$

where

- $p_{k,c} = \frac{1}{N_k} \sum_{x_i} I(x_{i,j} = v_k \; and \; y_i = c)$ is the proportion of training samples whose feature $X_j = v_k$ and output $y_i = c$
  (N_k is the number of samples at node m with feature $X_j = v_k$)

Insight: H_k will take on a small value if the node contains predominantly samples from a single class

Decision Trees: Most Important Feature (14)

b) Compute the information gain, $G(m, X_j)$, for node m, feature X_j:

$$G(m, X_j) = H_m - \sum_{k=1}^{K} \frac{N_k}{N_m} H_k$$

where

- N_m is the number of samples at current node m

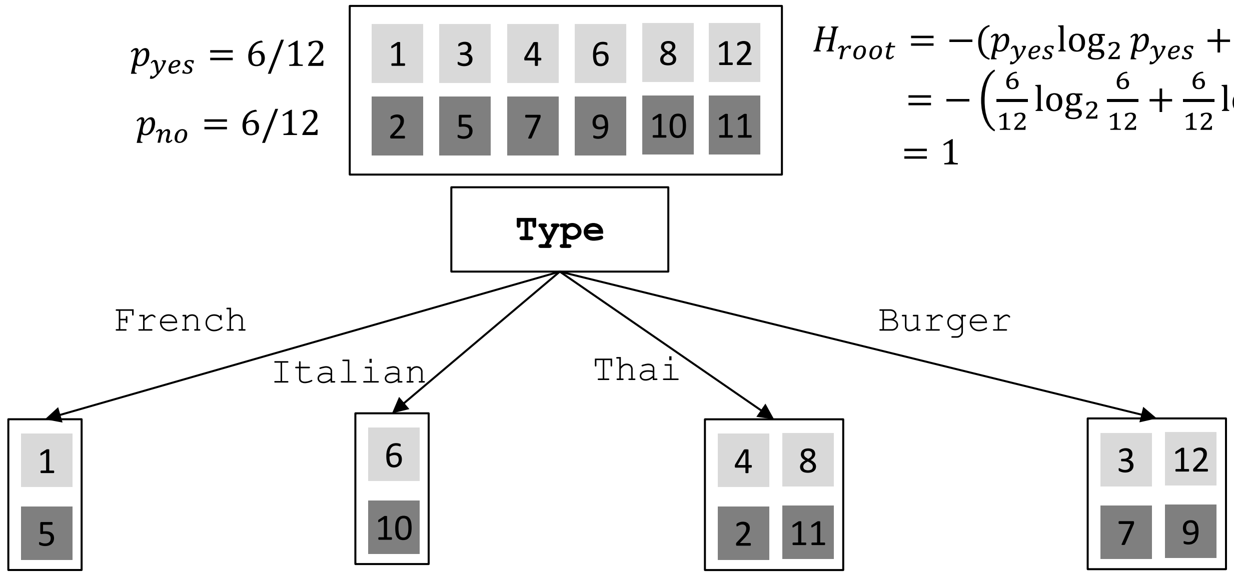
3. Select the feature with the highest information gain
4. Add a branch for each value v_k of X_j to the current node
5. Repeat steps 1 to 4 for the subtrees until **stopping criterion** is met

Decision Trees:
Example - Most Important Feature (15)

Step 1)

$p_{yes} = 6/12$

$p_{no} = 6/12$



$$\begin{aligned} H_{root} &= -(p_{yes} \log_2 p_{yes} + p_{no} \log_2 p_{no}) \\ &= -\left(\frac{6}{12}\log_2\frac{6}{12} + \frac{6}{12}\log_2\frac{6}{12}\right) \\ &= 1 \end{aligned}$$

Step 2a)

French	Italian	Thai	Burger
$p_{yes} = 1/2$	$p_{yes} = 1/2$	$p_{yes} = 2/4$	$p_{yes} = 2/4$
$p_{no} = 1/2$	$p_{no} = 1/2$	$p_{no} = 2/4$	$p_{no} = 2/4$

$$\begin{aligned} H_{French} &= -(p_{yes} \log_2 p_{yes} + p_{no} \log_2 p_{no}) \\ &= -\left(\tfrac{1}{2}\log_2\tfrac{1}{2} + \tfrac{1}{2}\log_2\tfrac{1}{2}\right) \\ &= 1 \end{aligned}$$

$H_{Italian} = 1$ $\quad H_{Thai} = 1$ $\quad H_{Burger} = 1$

Step 2b) $$Gain(root, Type) = H_{root} - \sum_k \frac{N_k}{N_{root}} H_k = 1 - \left(\frac{2}{12}(1) + \frac{2}{12}(1) + \frac{4}{12}(1) + \frac{4}{12}(1)\right) = 0$$

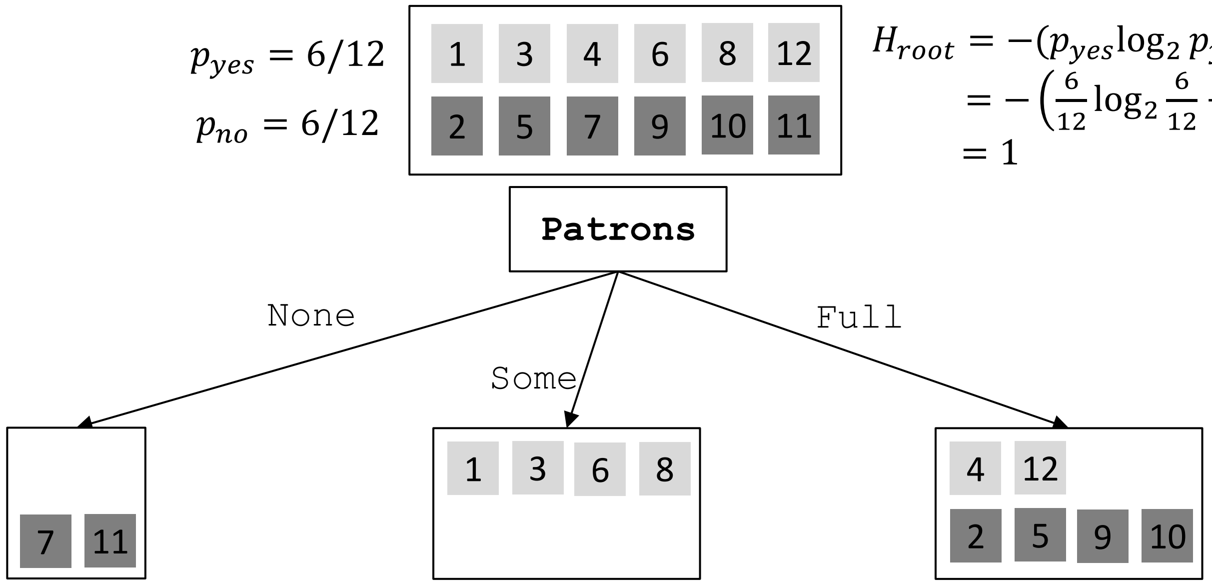
Decision Trees: Example - Most Important Feature (16)

Step 1)

$p_{yes} = 6/12$

$p_{no} = 6/12$

$$H_{root} = -(p_{yes}\log_2 p_{yes} + p_{no}\log_2 p_{no}) = -\left(\frac{6}{12}\log_2\frac{6}{12} + \frac{6}{12}\log_2\frac{6}{12}\right) = 1$$



Step 2a)

$p_{yes} = 0/2$	$p_{yes} = 4/4$	$p_{yes} = 2/6$
$p_{no} = 2/2$	$p_{no} = 0/4$	$p_{no} = 4/6$
$H_{None} = 0$	$H_{Some} = 0$	$H_{Full} = 0.918$

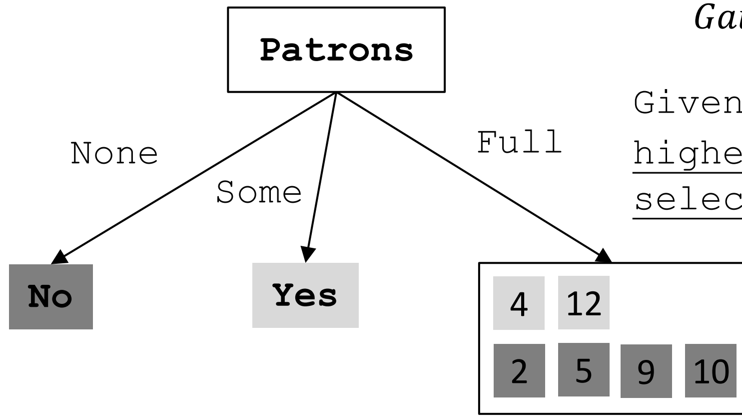
Step 2b) $$Gain(root, Patrons) = H_{root} - \sum_k \frac{N_k}{N_{root}} H_k = 1 - \left(\frac{2}{12}(0) + \frac{4}{12}(0) + \frac{6}{12}(0.918)\right) = 1 - 0.459 = 0.541$$

- Compute $Gain(root, X_j)$ for the remaining features (Fri, Hungry, Price and Raining) …

Decision Trees:
Example - Most Important Feature (17)

Step 3)

Step 4)



- Select the feature that has the highest $Gain(root, X_j)$

Given that **Patrons** has the highest $Gain(root, X_j)$, it is selected as the root node

Step 5)

Divide-and-conquer Approach:
For subtree at node m, compute $Gain(m, X_j)$ for every remaining feature X_j (Fri, Hungry, Price, Raining and Type), i.e., excluding features of parent node(s), using the remaining six (6) training samples (x_2, x_4, x_5, x_9, x_{10} and x_{12}) and select one that has the highest $Gain(m, X_j)$

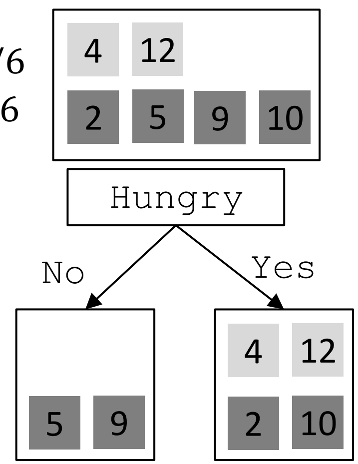
Decision Trees: Example - Most Important Feature (18)

Sample	Input Features X_j						WillWait y_i
	Fri	Hungry	Patrons	Price	Raining	Type	
x_2	No	Yes	Full	$	No	Thai	$y_2 =$ No
x_4	Yes	Yes	Full	$	Yes	Thai	$y_4 =$ Yes
x_5	Yes	No	Full	$$$	No	French	$y_5 =$ No
x_9	Yes	No	Full	$	Yes	Burger	$y_9 =$ No
x_{10}	Yes	Yes	Full	$$$	No	Italian	$y_{10} =$ No
x_{12}	Yes	Yes	Full	$	No	Burger	$y_{12} =$ Yes

Step 1)

$$p_{yes} = 2/6 \qquad p_{no} = 4/6$$

$$H_m = -(p_{yes}\log_2 p_{yes} + p_{no}\log_2 p_{no}) = -\left(\tfrac{2}{6}\log_2\tfrac{2}{6} + \tfrac{4}{6}\log_2\tfrac{4}{6}\right) = 0.918$$



Step 2a)

$$p_{yes} = 0/2 \qquad p_{no} = 2/2 \qquad H_{No} = 0$$

$$p_{yes} = 2/4 \qquad p_{no} = 2/4 \qquad H_{Yes} = 1$$

Step 2b)

$$Gain(m, Hungry) = H_m - \sum_k \frac{N_k}{N_m} H_k = 0.918 - \left(\frac{2}{6}(0) + \frac{4}{6}(1)\right) = 0.918 - 0.667 = 0.251$$

- Compute $Gain(m, X_j)$ for the remaining features (Fri, Price, Raining and Type) …
- Continue with Steps 3 through 5 (until stopping criterion is met)

Decision Trees: Stopping Criteria (19)

- Examples of stopping criteria:
  - all training samples in a node belong to the same class
    (e.g., for Patrons = None, all training samples belong to No class)
  - maximum tree depth has been reached (a root node will have a depth of 0)
  - number of training samples in a node is less than some pre-defined threshold
    (e.g., number of training samples is less than 5 in a node)

- In the case of where a leaf node contains training samples of different classes, its value is determined by **majority vote,** i.e., the most commonly occurring class of the training samples

Example:
Constraint:
Maximum Tree Depth = 1