Decision Trees

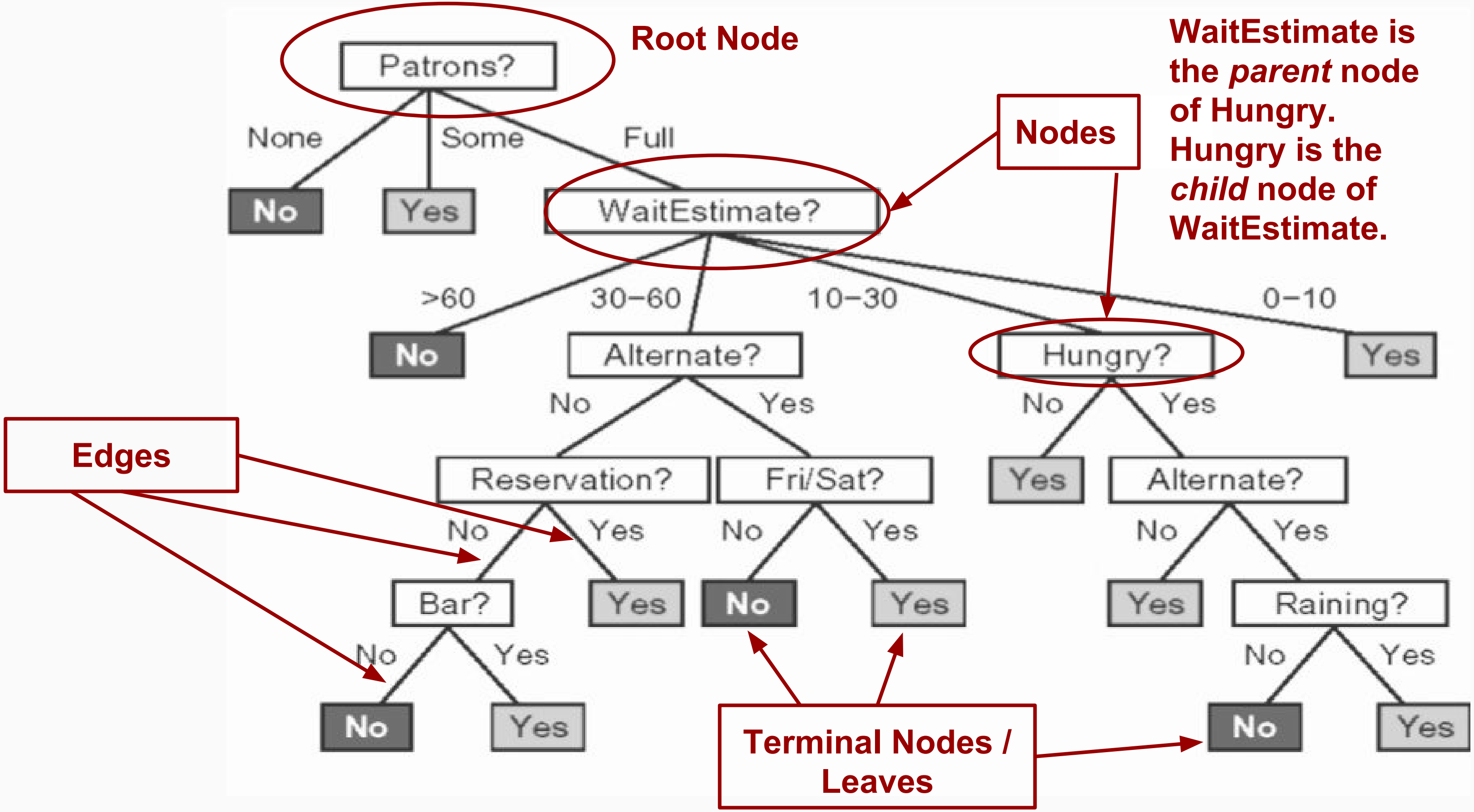
Kristie Wirth
Jon Courtney
Frank Burkholder
Ryan Henning

galvanize

Learning Objectives

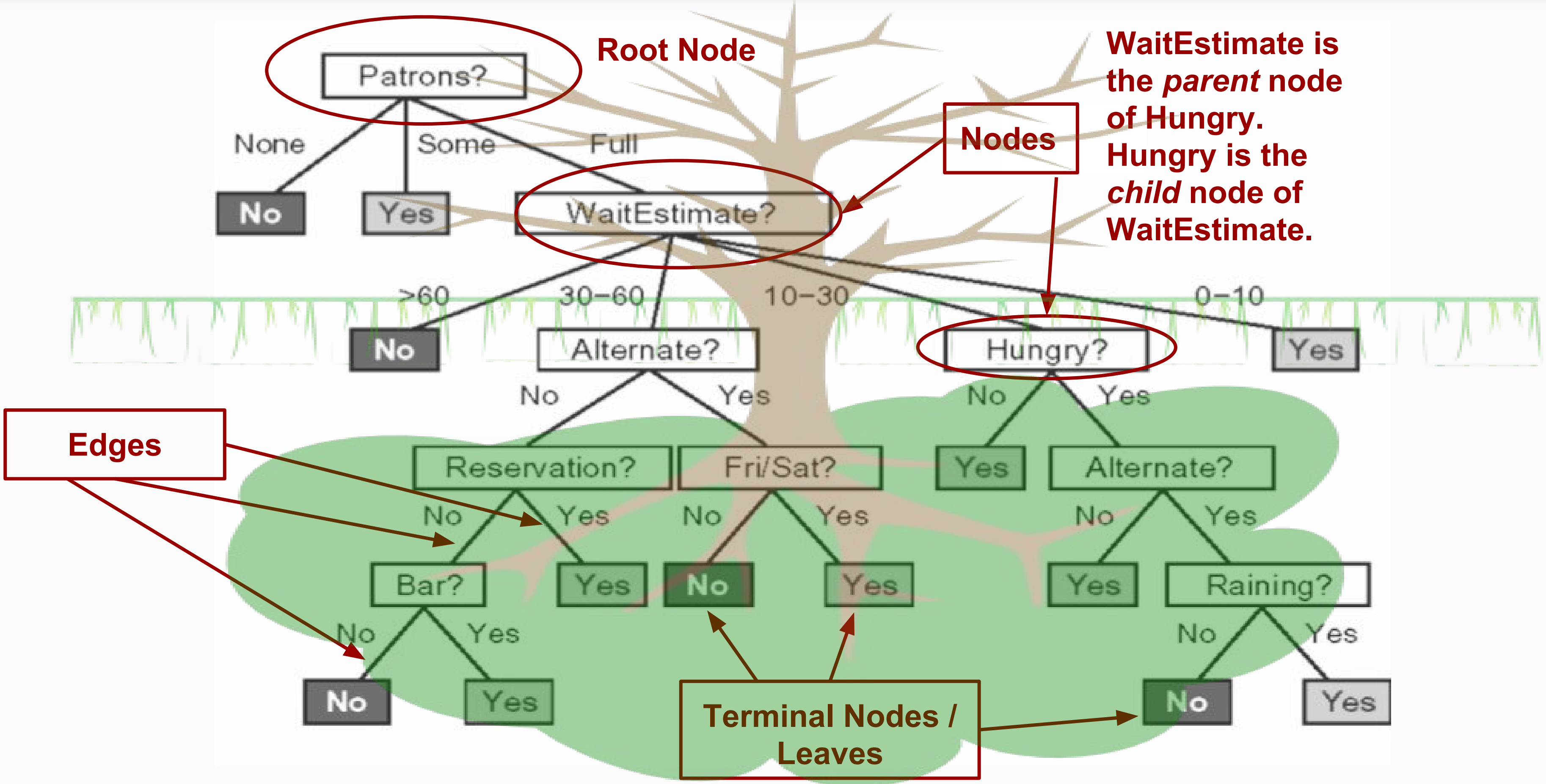
- Understand how the algorithm for decision trees works
- Be able to calculate Shannon entropy and information gain
- Explain the concept of recursion and how it relates to decision trees
- Know advantages & disadvantages of decision trees

Decision/Target Variable: Whether to wait for a table at a restaurant



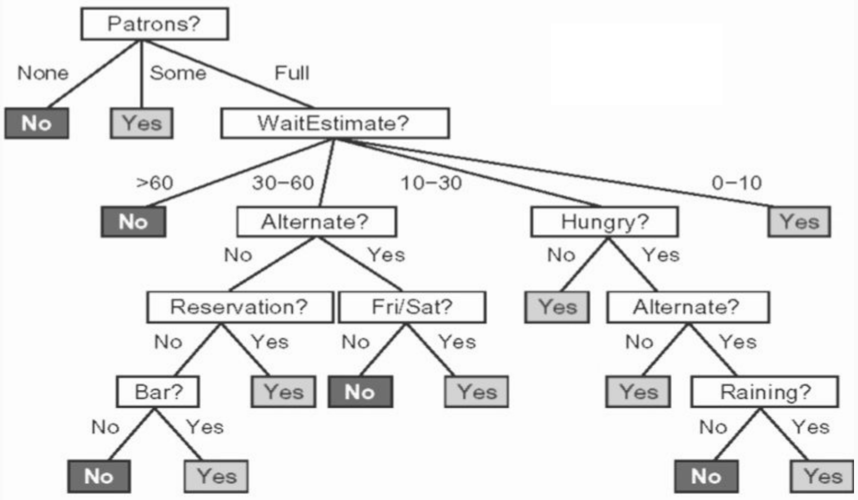
Decision Trees are Upside-Down...

galvanize



Decision Trees Overview

- Supervised learning
- Non-parametric model
- A series of sequential splits
- Each split based on a single feature
  - E.g., split by the wait estimate
- Splits are chosen to best separate the target variable
  - E.g., being hungry vs not being hungry is a good way to differentiate between whether you should go to the restaurant or not
- Target variable can be either categorical (classification tree) or numerical (regression tree)
- Overall goal: minimize error in your predictions of your target variable



Defining Concepts

- *Entropy* - a measurement of the diversity in a sample
    - High entropy - a sample that is partly made up of dogs and partly made up of horses
    - Low entropy - a sample that is 100% dogs
- Decision trees split the data on features to decrease entropy as much as possible
    - We are trying to separate the classes, e.g., split the dogs from the horses
- *Information gain* - a way to measure how much we reduced the entropy by splitting the data in a particular way
    - If we decrease the entropy by a large amount, then we have a large information gain
    - Information gain = parent_entropy - mean_child_entropy

Entropy Equation:

Proportion of class *i* in the sample

$$H(X) = -\sum_i p_i log_2(p_i)$$

Information Gain Equation:

Information gain from this split

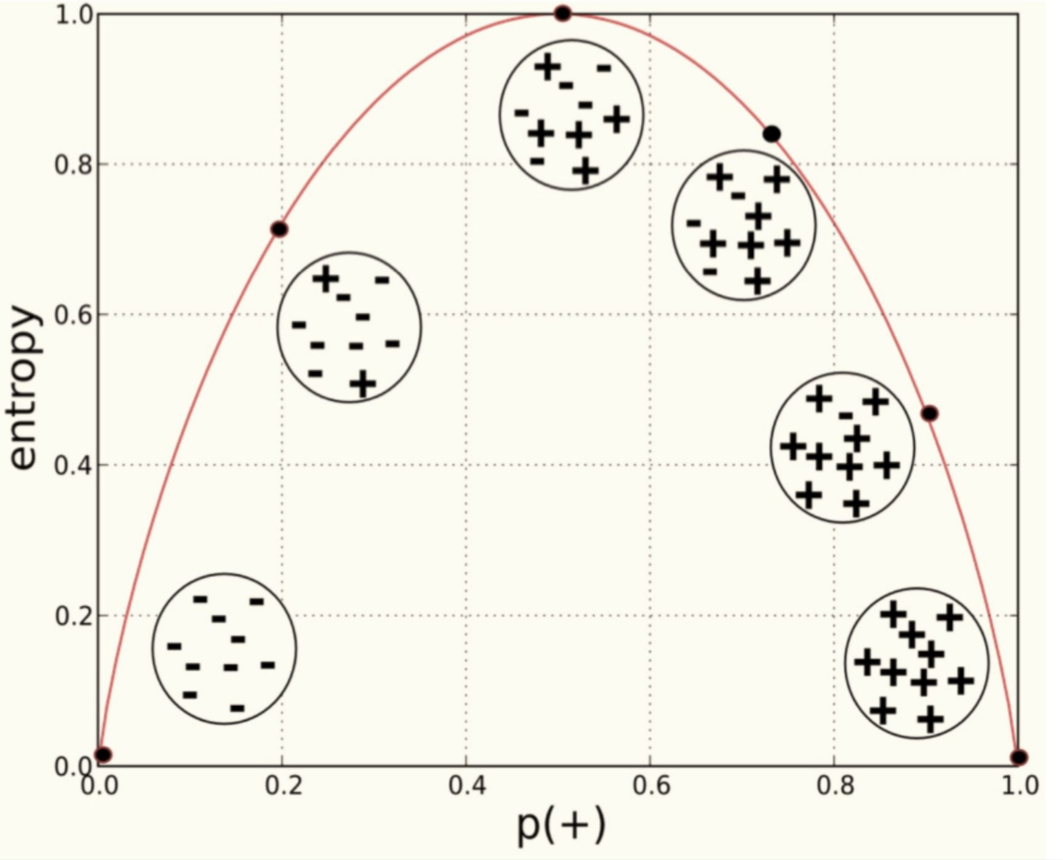
Entropy of the parent

Entropy of child *i*

$$\mathrm{IG}(S, C) = H(S) - \sum_{C_i \in C} \frac{|C_i|}{|S|} H(C_i)$$

Size of child node divided by size of parent node

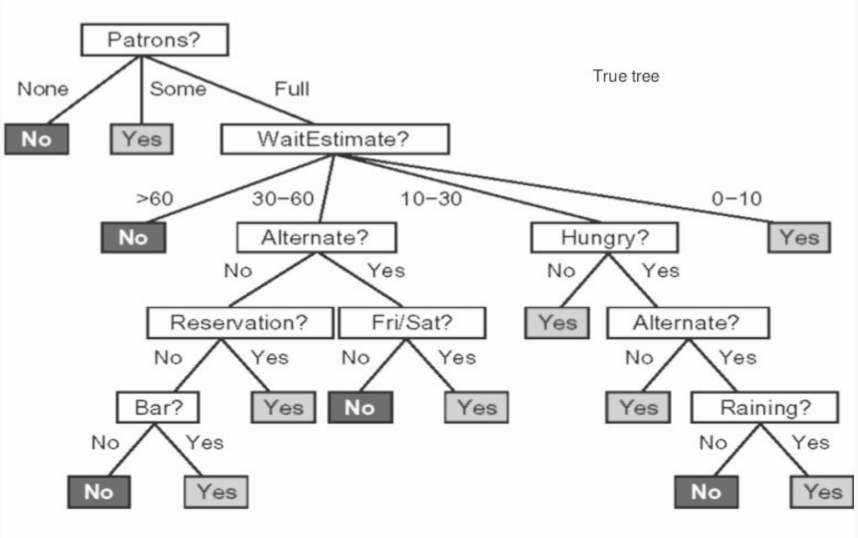
Entropy Graph of a Bernoulli Random Variable X



- If the probability of the positive class is 0, entropy is also 0.
- If the probability of the positive class is 1, entropy is 0.
- If the probability of the positive class is 0.5, entropy is at its highest value, 1.0.

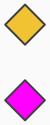
The Decision Tree Algorithm

1. Consider all possible splits on all features
    a. If a feature is categorical, split on value or not value (for a binary split).
    b. If a feature is numeric, split at a threshold: >threshold or <=threshold
2. Calculate & choose the "best" split
    a. Classification trees - the best split is the split that has the highest information gain when moving from parent to child nodes
    b. Regression trees - the best split is the split that has the largest reduction in variance when moving from parent to child nodes

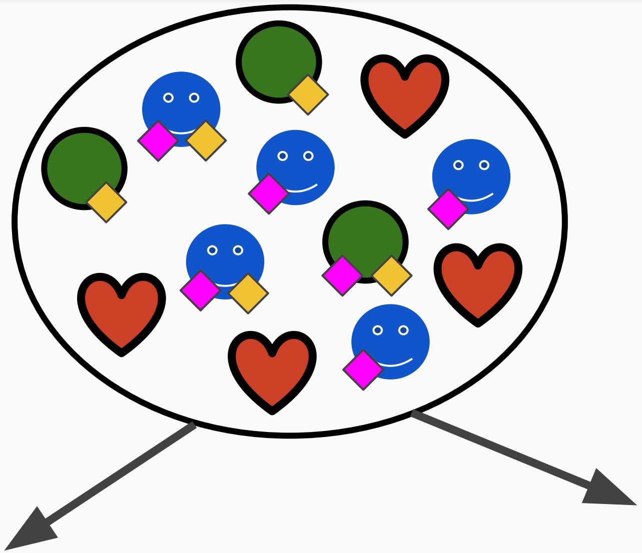


Classification Tree: The Splitting Process

Features:



Target Variable Categories:



What feature can we split on to maximize information gain?

Calculating Entropy

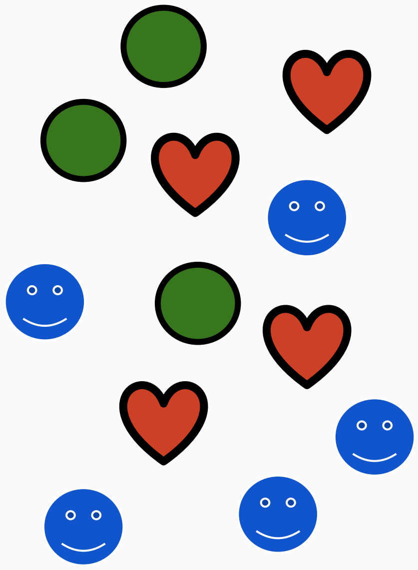
galvanize

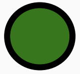
Entropy Equation:

Proportion of class *i* in the sample

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

Estimate:



P() = 3/12 = 0.25

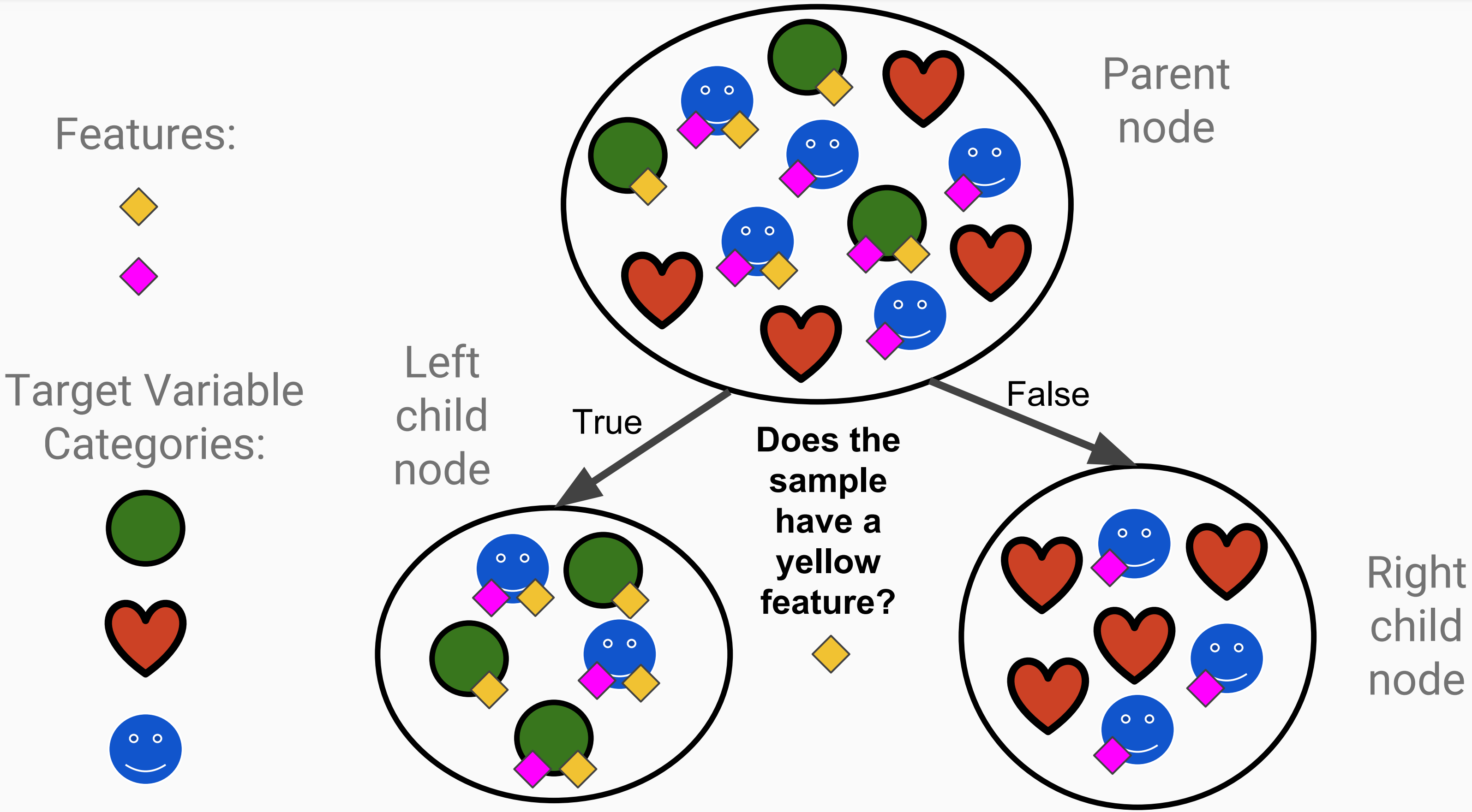
P() = 4/12 = 0.33

P() = 5/12 = 0.42

$$H = -0.25*\log_2(0.25) + -0.33*\log_2(0.33) + -0.42*\log_2(0.42)$$

H = 1.55

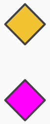
Splitting on Yellow



Calculating Entropy for each Node

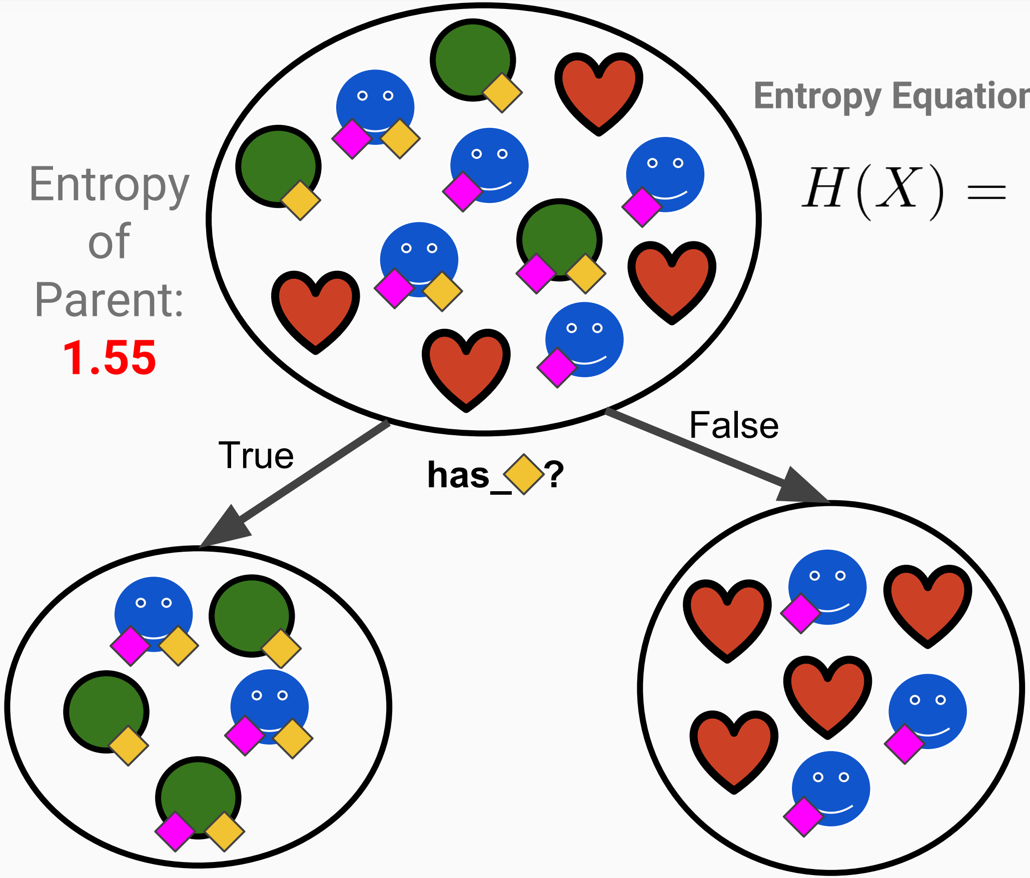
galvanize

Features:



Target Variable Categories:

Entropy of Parent: **1.55**



Entropy of left child: **0.97**

Entropy of right child: **0.985**

Entropy Equation:

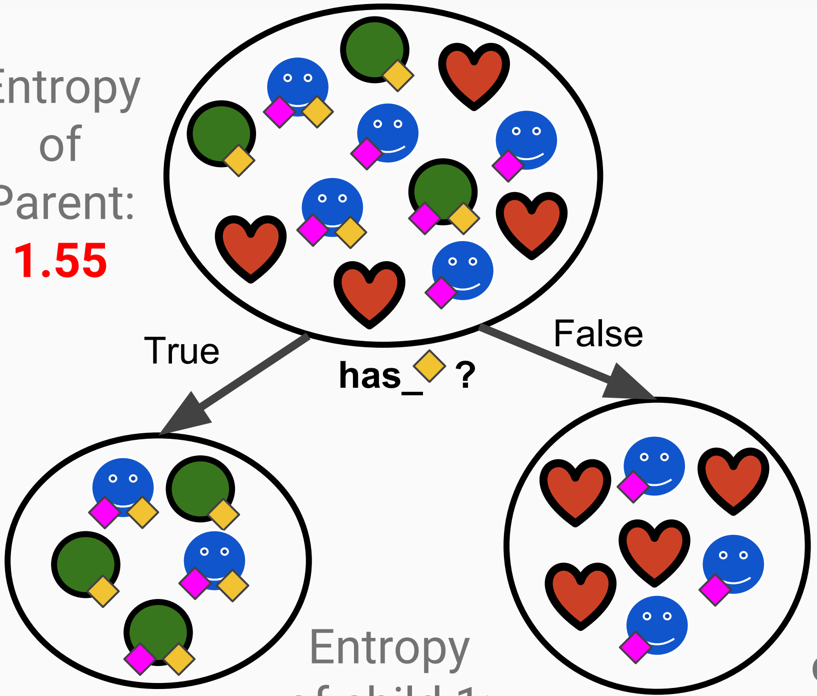
Proportion of class *i* in the sample

$$H(X) = -\sum_i p_i log_2(p_i)$$

Determining Information Gain from a Split

galvanize

Entropy of Parent: **1.55**



Entropy of child 1: **0.97**

Entropy of child 2: **0.985**

Information Gain Equation:

Information gain from this split

Entropy of the parent

Entropy of child *i*

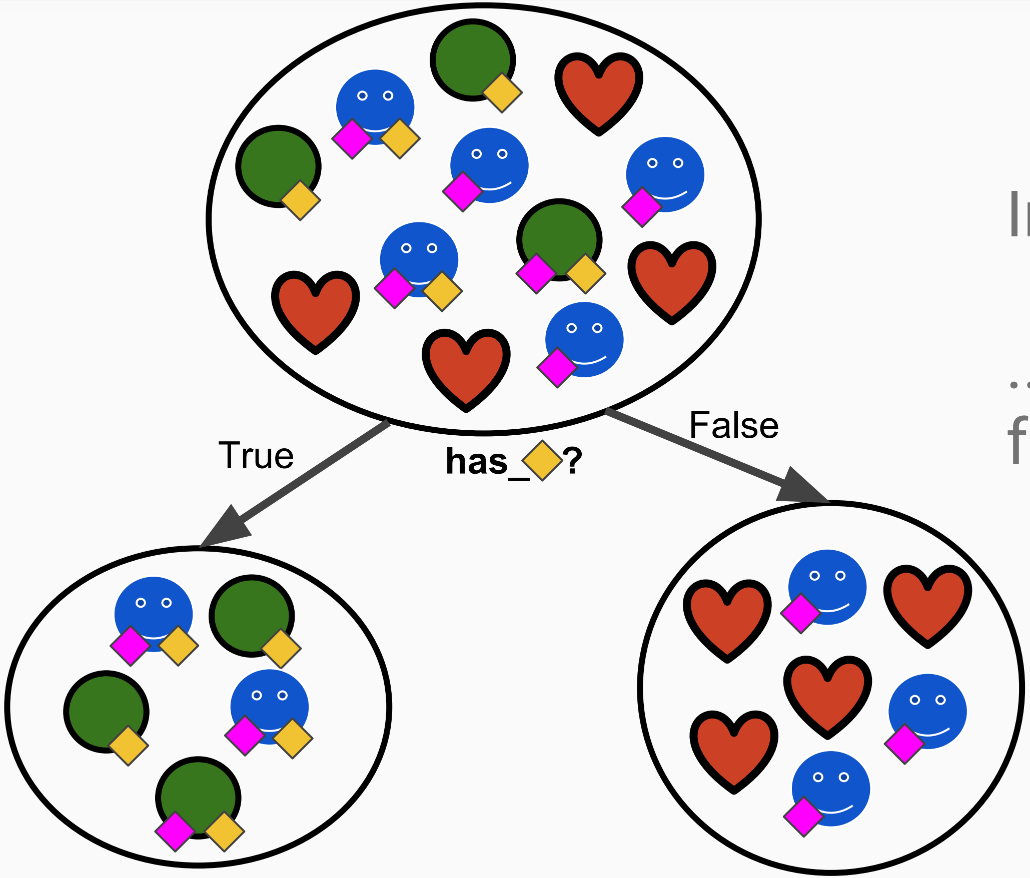
$$\mathrm{IG}(S, C) = H(S) - \sum_{C_i \in C} \frac{|C_i|}{|S|} H(C_i)$$

Size of child node divided by size of parent node

$$\mathrm{IG}(\text{parent}, \{\text{child\_1}, \text{child\_2}\}) = 1.55 - 5/12 * 0.97 - 7/12 * 0.985 = 0.57$$

Determining Information Gain from a Split

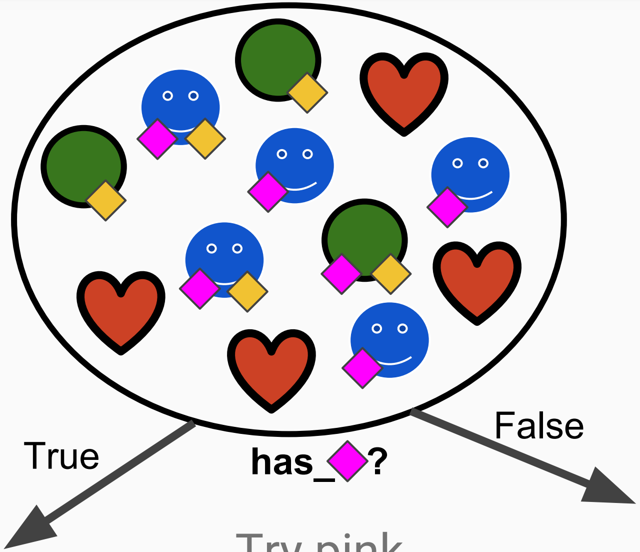
galvanize



Information Gain = 0.57

...for splitting on yellow feature.

Determining Information Gain from a Split



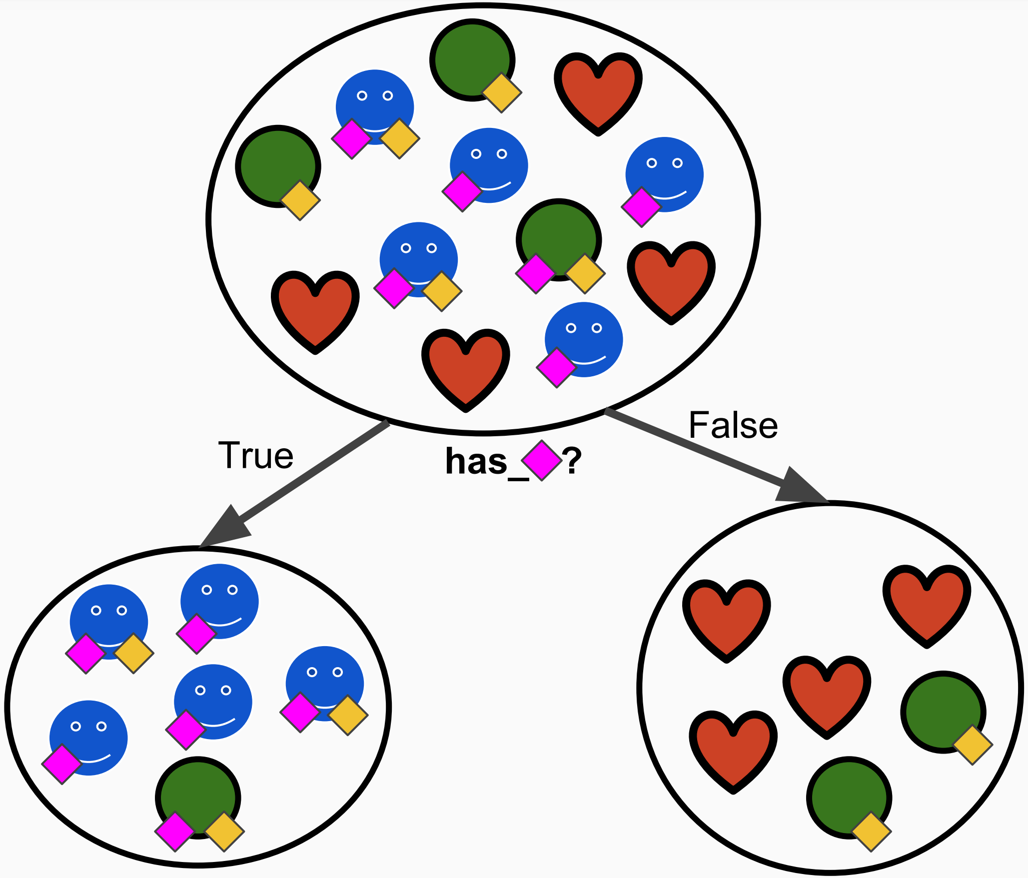
Try pink feature. Better or worse?

Information Gain Equation:

$$\mathrm{IG}(S, C) = H(S) - \sum_{C_i \in C} \frac{|C_i|}{|S|} H(C_i)$$

What is the information gain for splitting on the pink feature?

Determining Information Gain from a Split



Information Gain = 0.766

Better! In this case we would choose to split on the pink feature (higher information gain)

Comparing Types of Information Gain

Shannon Entropy - Measures the **diversity** of a sample

$$H(X) = -\sum_i p_i log_2(p_i)$$

Information Gain Using Shannon Entropy

$$\mathrm{IG}(S, C) = H(S) - \sum_{C_i \in C} \frac{|C_i|}{|S|} H(C_i)$$

Gini Index - Measures the **probability of misclassifying** a single element if it was randomly labeled according to the distribution of classes in the sample

$$\mathrm{Gini}(S) = 1 - \sum_{i \in S} p_i^2$$

Information Gain Using Gini Index

$$\mathrm{IG}(S, C) = \mathrm{Gini}(S) - \sum_{C_i \in C} \frac{|C_i|}{|S|} \mathrm{Gini}(C_i)$$

Recursion: When a Function Calls Itself

```python
def f(x):
    '''
    This function returns x!.
    INPUT: 5
    OUTPUT: 120
    '''
    if x <= 1:
        return 1
    else:
        return x * f(x-1)
```

Base case

Recursive call

INPUT: f(5)

5 * f(4)

5 * 4 * f(3)

5 * 4 * 3 * f(2)

5 * 4 * 3 * 2 * f(1)

5 * 4 * 3 * 2 * 1

Different ways to write the exact same function:

$$f(x) = \begin{cases} 1, & \text{if } x \leq 1 \\ x f(x-1), & \text{otherwise} \end{cases}$$

$$f(x) = \prod_{i=1}^{x} i$$

Decision Tree Pseudocode

galvanize

```
function BuildTree:
    if every item in the dataset is in the same class
    or there is no feature left on which to split the data:
        return a leaf node with the class label
    else:
        find the best feature and value to split the data on
        split the dataset
        create a node
        for each split
            call BuildTree and add the result as a child of the node
        return node
```

Pruning: Preventing Overfitting

- Trees will overfit by default unless you direct them otherwise
- Pruning involves a bias-variance trade-off
- Pre-pruning ideas (pruning **while** you build the tree)
  - Leaf size: stop splitting when the number of samples left gets small enough
  - Depth: stop splitting at a certain depth (after a certain number of splits)
  - Purity: stop splitting if enough of the examples are the same class
  - Gain threshold: stop splitting when the information gain becomes too small
- Post-pruning ideas (pruning **after** you've finished building the tree)
  - Merge terminal nodes (i.e., undo your split) if doing so decreases error in your test set
  - Set the maximum number of terminal nodes; this is a form of regularization

Algorithm Options for Decision Trees

- **ID3:** category features only, information gain, multi-way splits
- **C4.5:** continuous and categorical features, information gain, missing data okay, pruning
- **CART:** continuous and categorical features and targets, gini index, binary splits only

Advantages & Disadvantages of Decision Trees

Advantages:

- Easy to interpret
- Non-parametric/more flexible model
- Can incorporate both numerical and categorical features*
- Prediction is computationally cheap
- Can handle missing values and outliers*
- Can handle irrelevant features and multicollinearity

Disadvantages:

- Computationally expensive to train
- Greedy algorithm - looks for the simplest, quickest model and may miss the best model (i.e., converges at local maxima instead of global maxima)
- Often overfits
- Deterministic - i.e., you'll get the same model every time you run it

* = some exceptions in sklearn

Decision Trees in sklearn

- Uses the CART algorithm (see http://scikit-learn.org/stable/modules/tree.html#tree, section 1.10.6)
- Uses Gini Index by default (but you can change it to entropy if you'd like)
- You can prune by varying the following hyperparameters: `max_depth`, `min_samples_split`, `min_samples_leaf`, `max_leaf_nodes`
- Must use dummy variables for categorical features
    - E.g., a column with ['Red', 'Green', 'Blue'] would need to be coded as separate variables - Is_it_red (Y/N), Is_it_green (Y/N), Is_it_blue (Y/N)
    - See 'Feature Binarization and Encoding Categorical Features' at http://scikit-learn.org/stable/modules/preprocessing.html
- Does not support missing values (even though CART typically does)
- Only supports binary splits

Check in Questions

- Explain how the algorithm for decision trees works.
- How is Shannon entropy calculated?
- How is information gain calculated?
- What is recursion? How does it relate to decision trees?
- What are some advantages of decision trees?
- What are some disadvantages of decision trees?