CS57300
PURDUE UNIVERSITY

FEBRUARY 5, 2019

DATA MINING

ANNOUNCEMENT

- Values on some attributes is out of the range for the Assignment 2 data
  - e.g., a value of 14 on "gaming"
  - Please replace these anomalies with the maximum possible value for that attribute

NAIVE BAYES CLASSIFIER

NBC LEARNING

- Model space
  - Parametric model with specific form (i.e., based on Bayes rule and assumption of conditional independence)
  - Models vary based on parameter estimates in CPDs
- Search algorithm
  - MLE optimization of parameters (convex optimization results in exact solution)
- Scoring function
  - Likelihood of data given NBC model form

NBC: MAP ESTIMATION

- Consider a simplified scenario: binary classification (i.e., L=2) and each attribute is binary (i.e., $K(j)$=2)
- Priors: $p_1 \sim Beta(a, b), q_l^{j1} \sim Beta(\alpha_l^j, \beta_l^j)$
- MAP estimate:
  - Maximize $\dfrac{P(D\,|\,\theta)P(\theta)}{P(D)}$

NBC: MAP ESTIMATION

$$P(D\,|\,\theta)P(\theta) \propto (\prod_{i=1}^{n}\prod_{j=1}^{m} P(x_{ij}\,|\,c_i)P(c_i)) \times P(p_1) \times \prod_{l=0}^{1}\prod_{j=1}^{m} P(q_l^{j1})$$

$$= \prod_{l=0}^{1} p_l^{N_l} \prod_{l=0}^{1}\prod_{j=1}^{m}\prod_{k=0}^{1} (q_l^{jk})^{N_l^{jk}} \times P(p_1) \times \prod_{l=0}^{1}\prod_{j=1}^{m} P(q_l^{j1})$$

$$\Downarrow$$

$$p_1 \sim Beta(a + N_1, b + N_0), q_l^{j1} \sim Beta(\alpha_l^{j1} + N_l^{j1}, \beta_l^{j1} + N_l^{j0})$$

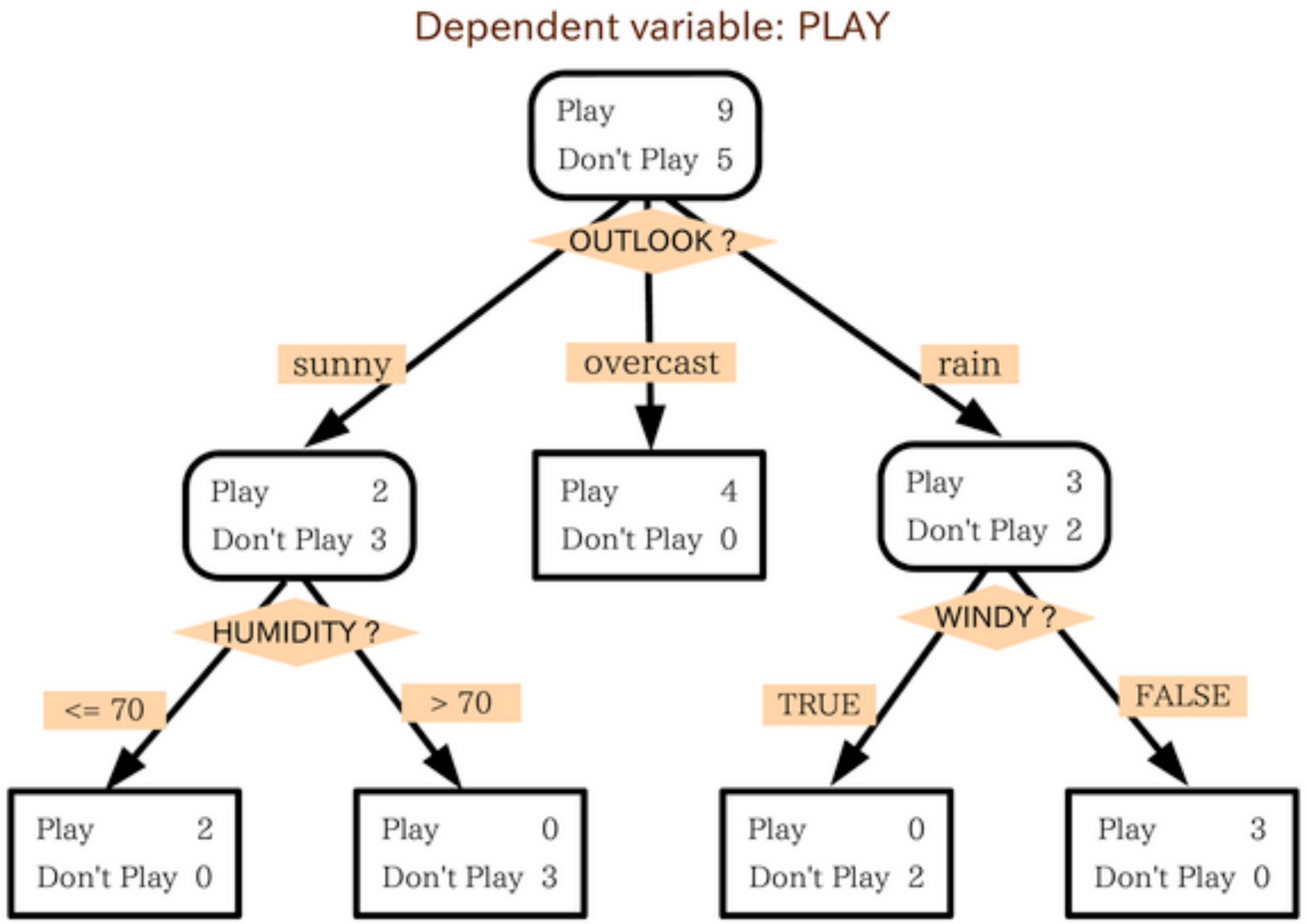
$$\Downarrow$$

$$[p_1]_{MAP} = \frac{a + N_1 - 1}{a + b + n - 2}, [q_l^{l1}]_{MAP} = \frac{\alpha_l^{j1} + N_l^{j1} - 1}{\alpha_l^{j1} + \beta_l^{j1} + N_l - 2}$$

DECISION TREES

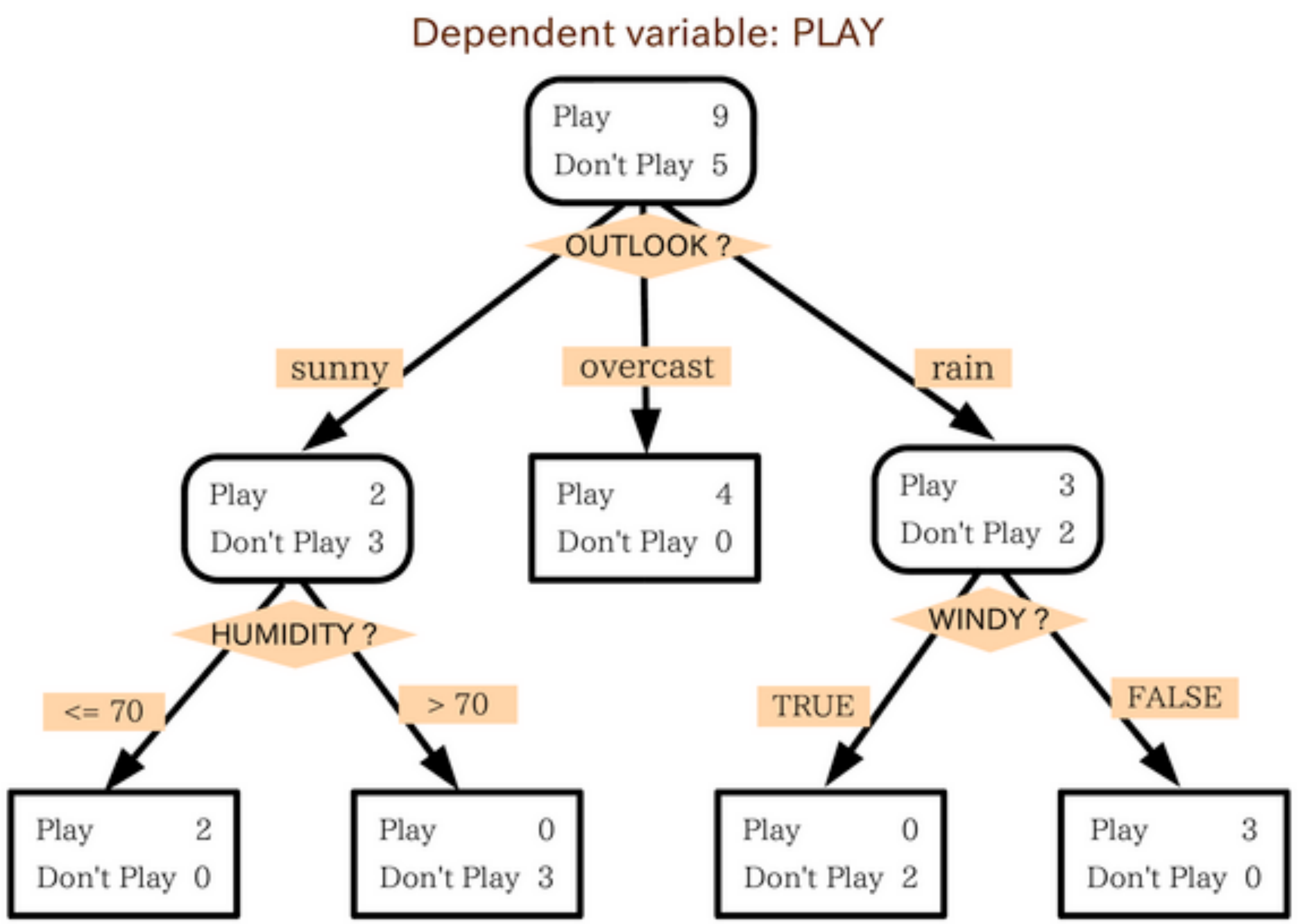
TREE MODELS: KNOWLEDGE REPRESENTATION

- A decision tree has 2 kinds of nodes
  - Each internal node is a question on features. It branches out according to the answers
  - Each leaf node has a class label, determined by the majority vote of training examples reaching that leaf
- Advantages
  - Easy inference
  - Can handle mixed variables
  - Easy for humans to understand



TREE LEARNING

- Model space: All possible decision trees
  - Each layer can include different attributes
  - Each attribute can split on different values
  - Can have different number of layers
- Scoring function: Misclassification rate
- Search process: Heuristic search
  - Greedy, recursive divide and conquer

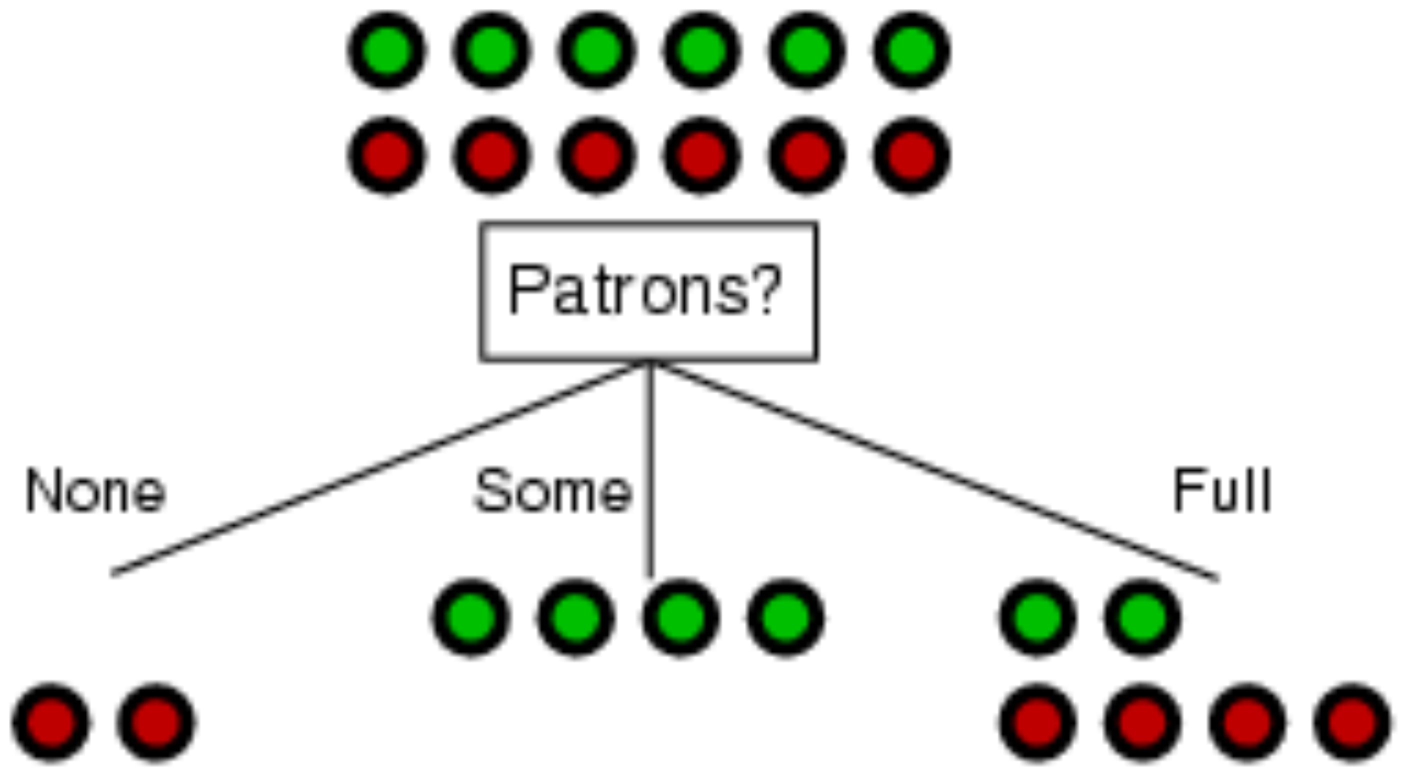


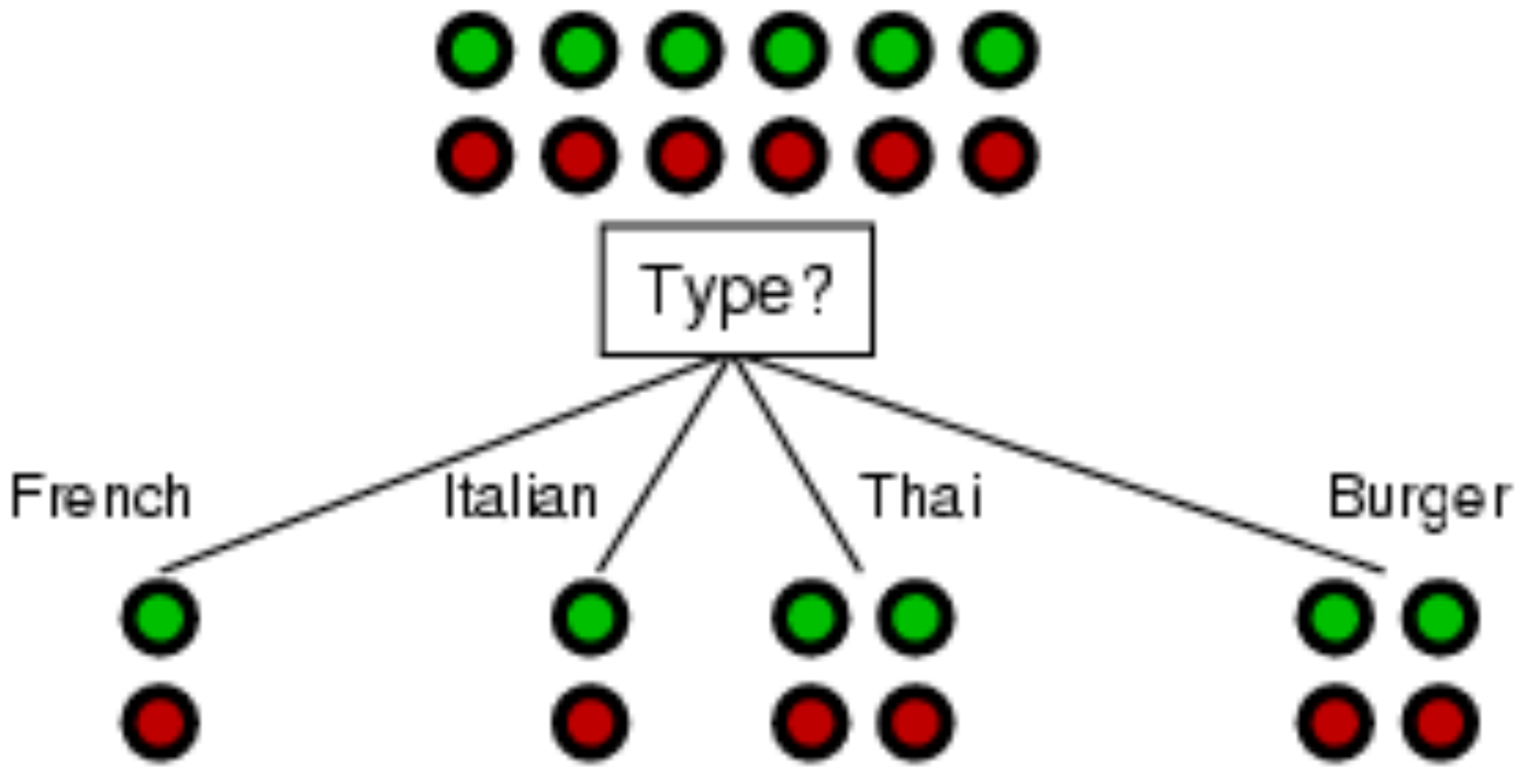
TREE LEARNING

- Top-down recursive divide and conquer algorithm
  - Start with all training examples at root
  - Select **best** attribute/feature: Take a greedy view to decide how "good" an attribute is
  - Partition examples by selected attribute
  - Recurse and repeat
- Other issues:
  - When to stop growing
  - Pruning irrelevant parts of the tree

CHOOSING AN ATTRIBUTE/FEATURE

- Be greedy: choose an attribute that can immediately minimize the misclassification rate (i.e., as if no further subtree will grow)
- A good feature splits the examples into subsets that distinguish among the class labels as much as possible… ideally into pure sets of "all positive" or "all negative"

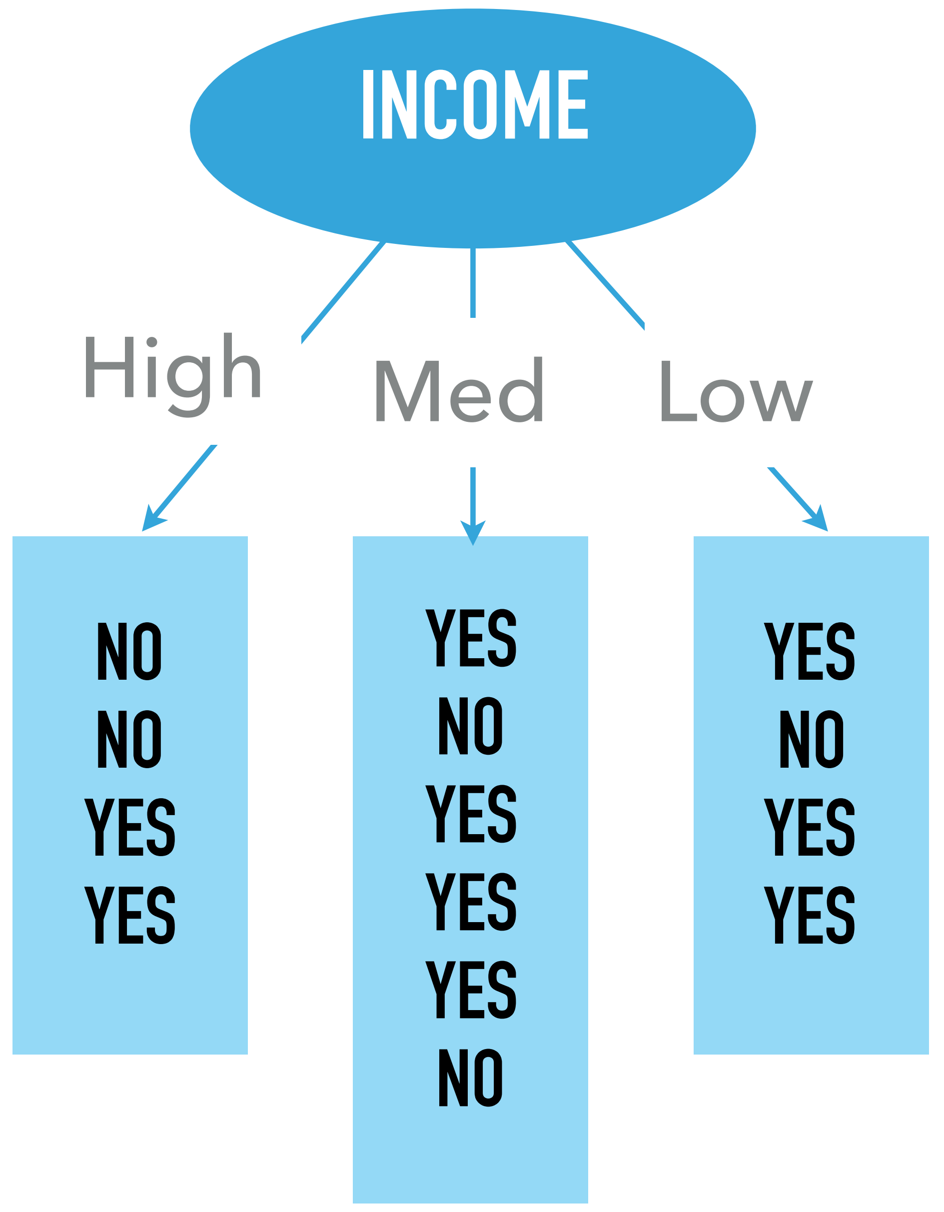




ASSOCIATION BETWEEN ATTRIBUTE AND CLASS LABEL

Data

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no



Contingency table

Class label value

Attribute value	Buy	No buy
High	2	2
Med	4	2
Low	3	1

A good attribute leads to **highly certain** prediction for training examples sharing the same value on that attribute!

MEASURING UNCERTAINTY: ENTROPY

- Used to quantify the amount of randomness of a probability distribution.
- Definition: Suppose a discrete random variable X has a distribution of P(X). The **entropy** H(X) of X is defined by:

$$H(X) = -\sum_{x} p(x) \log_2 p(x)$$

ENTROPY OF A RANDOM VARIABLE

A completely random binary variable with P(X)=[0.5,0.5] has entropy:

$H(X) = -(0.5 \log_2 0.5 + 0.5 \log_2 0.5) = -(-0.5 + -0.5) = 1$

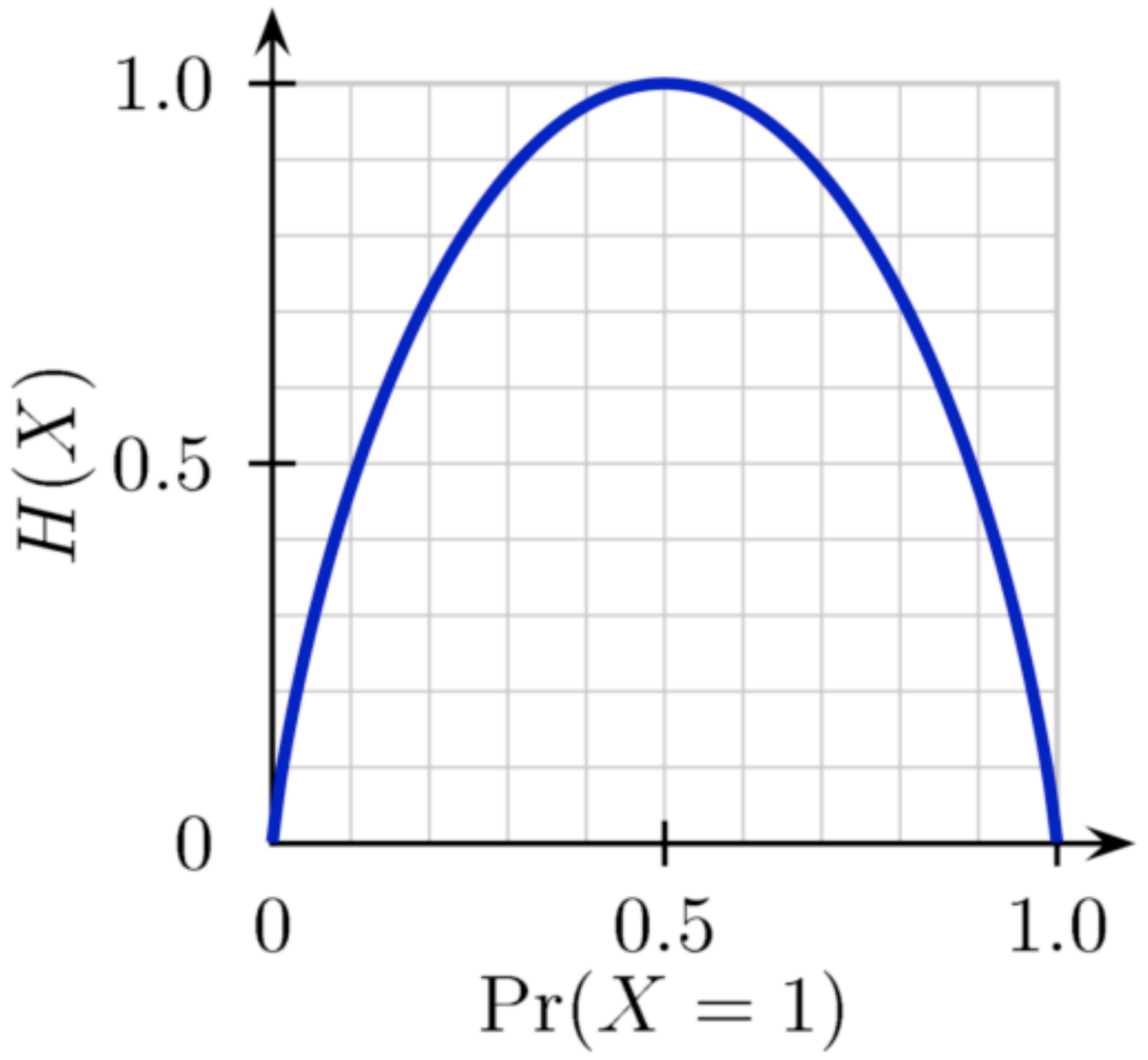
A deterministic variable with P(X)=[1,0] has entropy:

$H(X) = -(1 \log_2 1 + 0 \log_2 0) = -(0+0) = 0$

A biased variable with P(X)=[0.75,0.25] has entropy:

$H(X) = 0.8113$

The entropy of a probability distribution **p** expresses the **amount of uncertainty** that we have about the values of X



MEASURING CHANGE OF UNCERTAINTY: INFORMATION GAIN

- How much does a feature split decrease the entropy?

$$Gain(S, A) = \underline{Entropy(S)} - \sum_{v \in values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

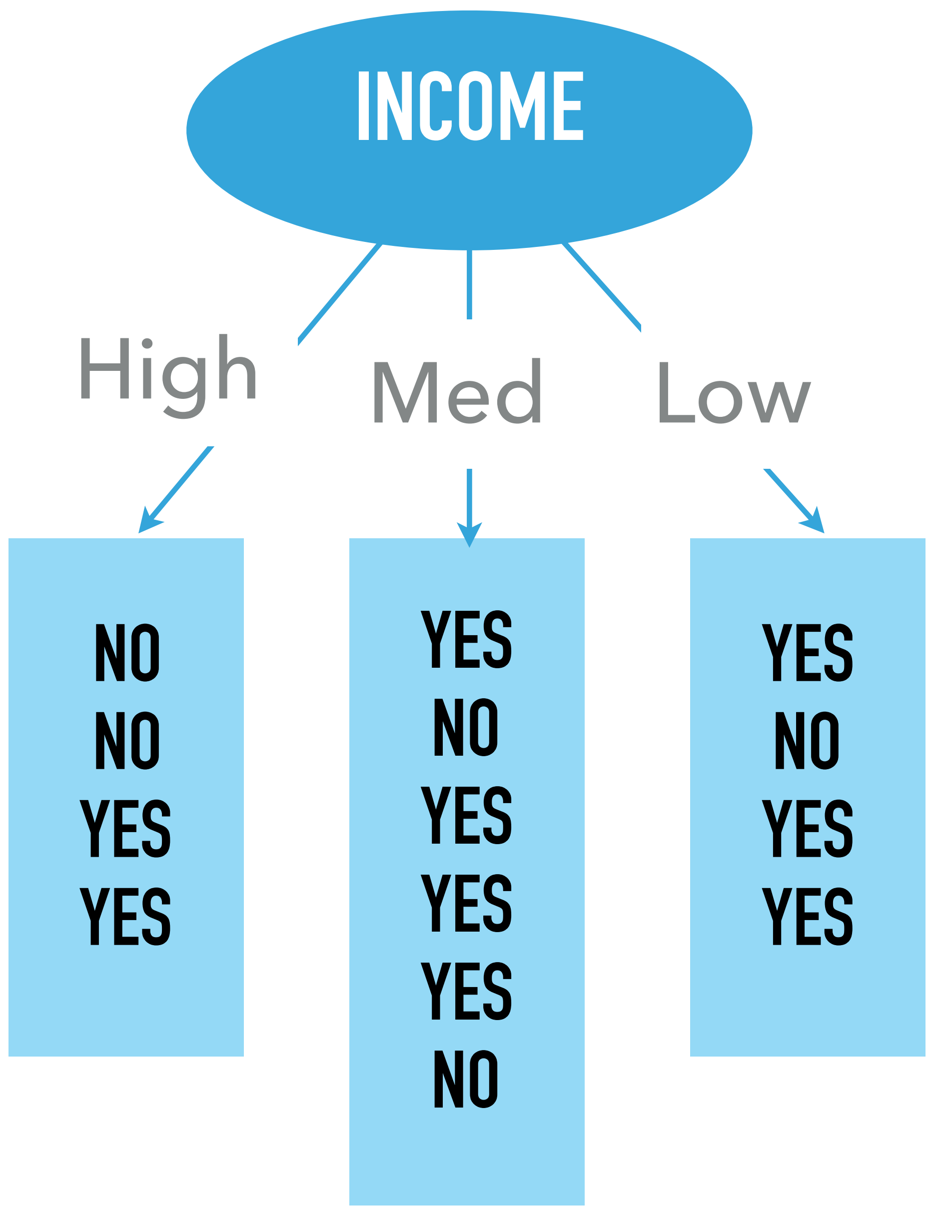
Entropy(S)

= -9/14 Log 9/14 -5/14 Log 5/14

= 0.9400

INFORMATION GAIN

$$Gain(S, A) = Entropy(S) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$



Entropy(Income=High)
= -2/4 Log 2/4 -2/4 Log 2/4 = 1

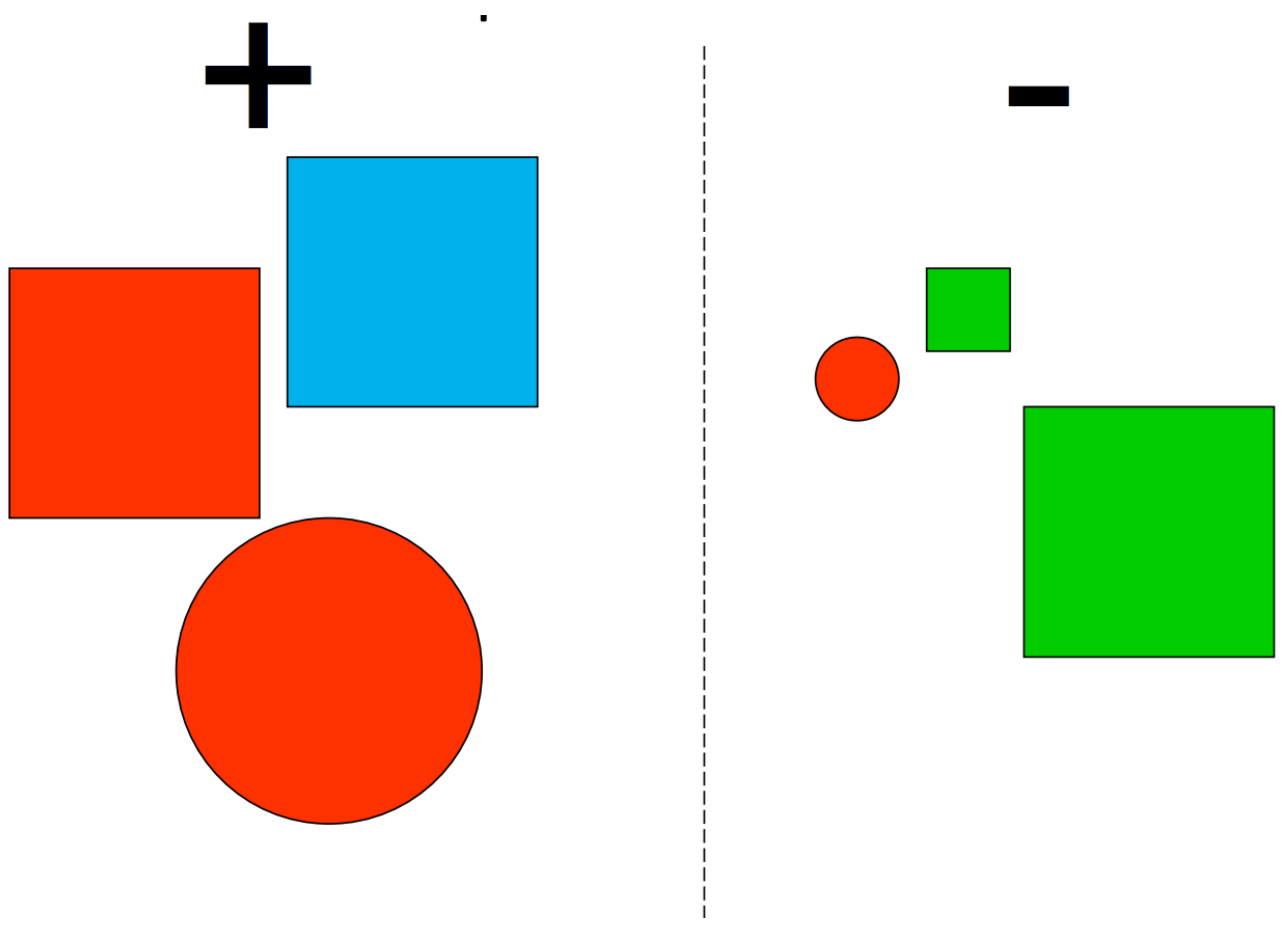
Entropy(Income=Med)
= -4/6 Log 4/6 -2/6 Log 2/6 = 0.9183

Entropy(Income=Low)
= -3/4 Log 3/4 -1/4 Log 1/4 = 0.8113

Gain(S,Income)
= 0.9400 - (4/14 [1] + 6/14 [0.9183] + 4/14 [0.8113])
= 0.029

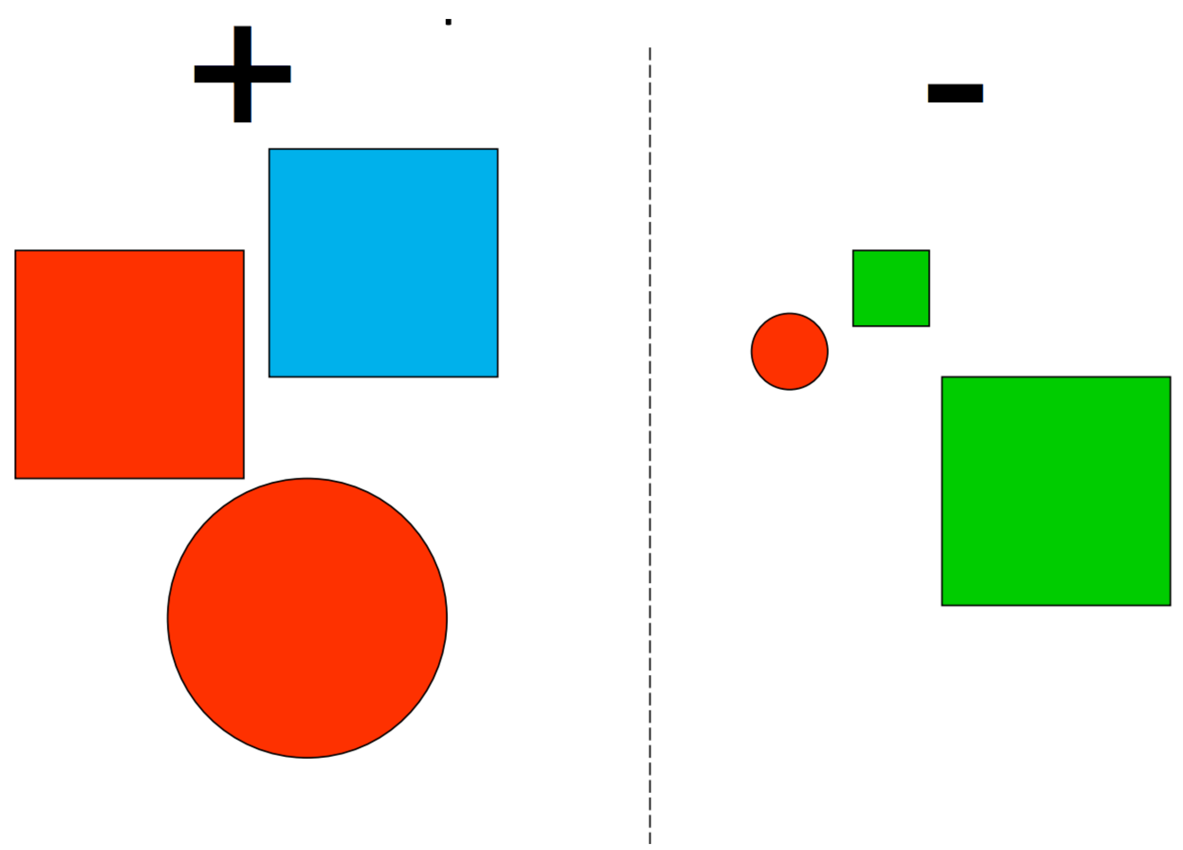
EXAMPLE: CHOOSE THE ATTRIBUTE WITH LARGEST INFORMATION GAIN

- Features: color, shape, size
- What's the best feature to use at root?



EXAMPLE: CHOOSE THE ATTRIBUTE WITH LARGEST INFORMATION GAIN

Example	Color	Shape	Size	Class
1	Red	Square	Big	+
2	Blue	Square	Big	+
3	Red	Circle	Big	+
4	Red	Circle	Small	-
5	Green	Square	Small	-
6	Green	Square	Big	-



- H(S) = -(0.5*log0.5+0.5*log0.5)=1
- Gain(S, Color) =1-0.5*(-0.67*log0.67-0.33*log0.33)-0.17*(-1*log1-0*log0)-0.33*(-1*log1-0*log0) = 0.54
- Gain(S, Shape) = 1- 0.67*(-0.5*log0.5-0.5*log0.5)-0.33*(-0.5*log0.5-0.5*log0.5)=0
- Gain(S, Size) = 1-0.67*(-0.75*log0.75-0.25*log0.25)-0.33*(-1*log1-0*log0)=0.46

BUILDING TREE RECURSIVELY

Buildtree(examples, attributes)

/*examples: a list of training examples at the current node
attributes: a set of candidate attributes to place question on*/

If examples={} **then** return

If examples have the same label *y* **then** return a leaf node with label *y*

If attributes={} **then** return a leaf node with the majority label in examples

A = Best_attribute(examples, attributes) /*Suppose attribute *A* has *n* possible values*/

Create an internal node, *node(A)*, with *n* children

For attribute *A*'s *i*-th possible value *A(i)*:

The i-th child of node(A) = **Buildtree**({examples with its value on *A* being *A(i)*}, attributes-{*A*})

DEALING WITH CONTINUOUS ATTRIBUTES

- Discretize the value of a continuous attribute into several intervals
  - Two bins for the continuous variable x: $x<=t$, $x>t$
  - $Gain(S, x, t)=Entropy(S)-|S_{x<=t}|*Entropy(S_{x<=t})/|S|-|S_{x>t}|*Entropy(S_{x>t})/|S|$
  - $Gain(S, x)=max_t \, Gain(S, x, t)$

ADDITIONAL ATTRIBUTE SELECTION CRITERIA: GINI GAIN

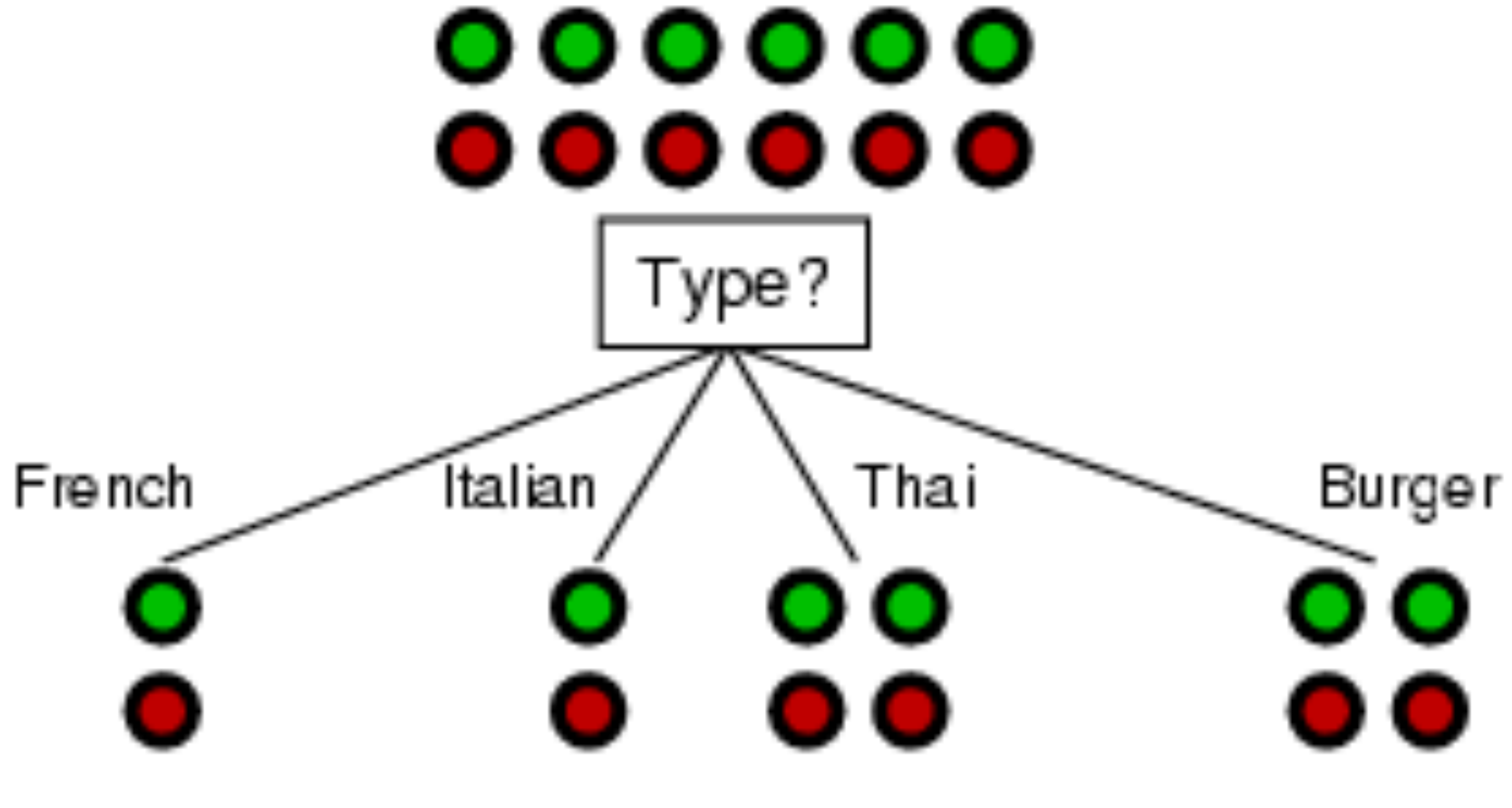
- Similar to information gain
- Uses gini index instead of entropy

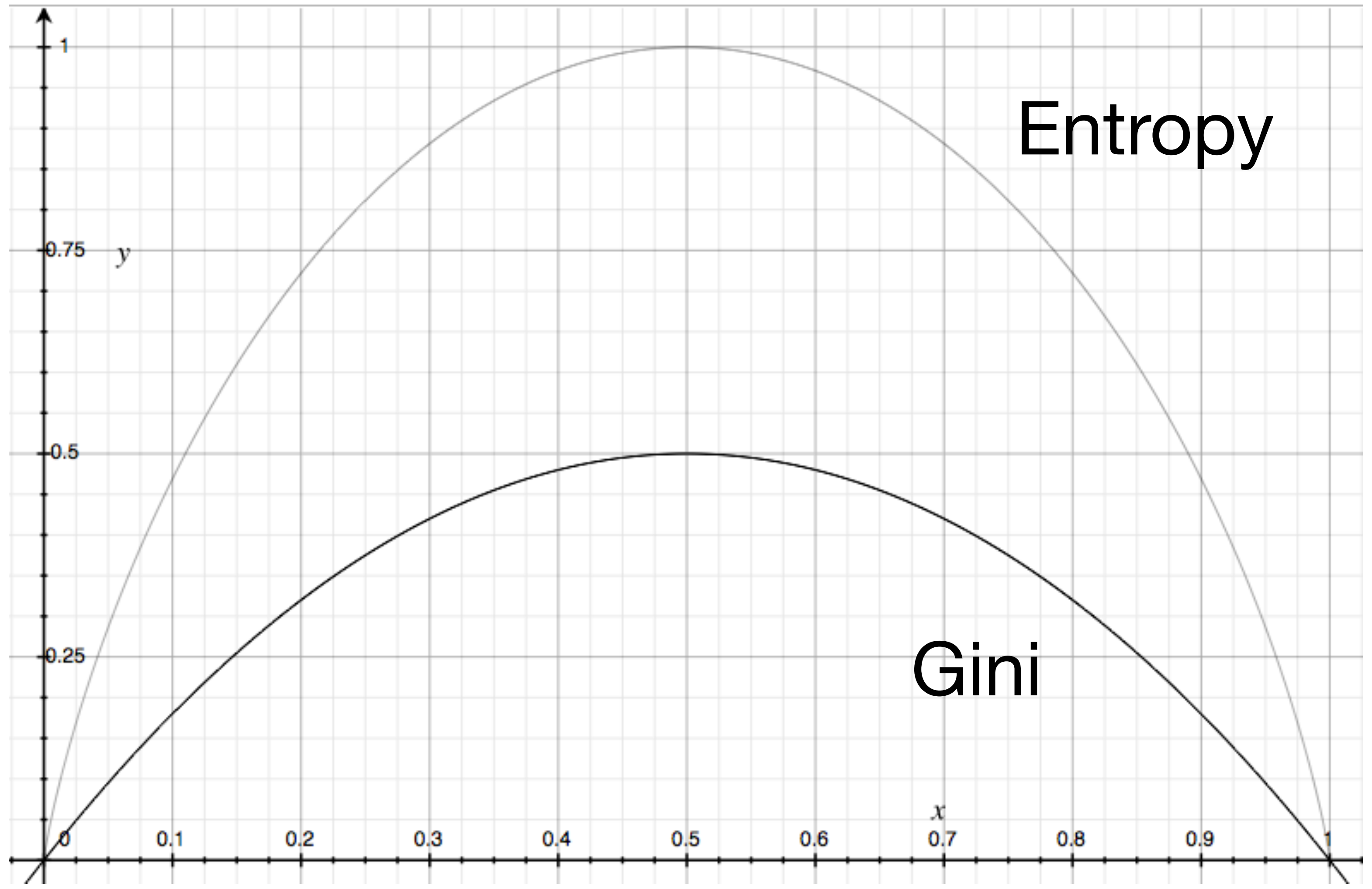
$$Gini(X) = 1 - \sum_{x} p(x)^2$$

- Measures decrease in gini index after split:

$$Gain(S, A) = Gini(S) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} Gini(S_v)$$

COMPARING INFORMATION GAIN TO GINI GAIN

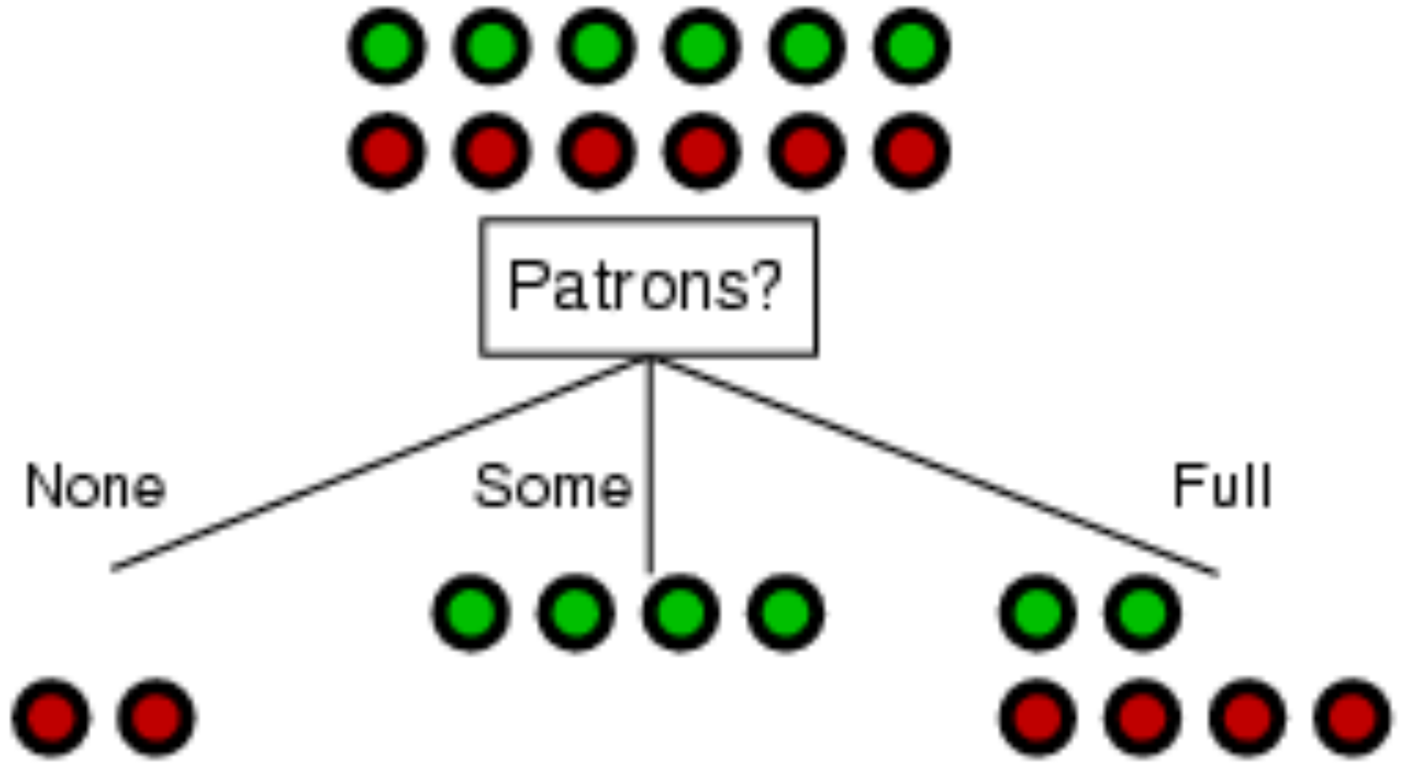


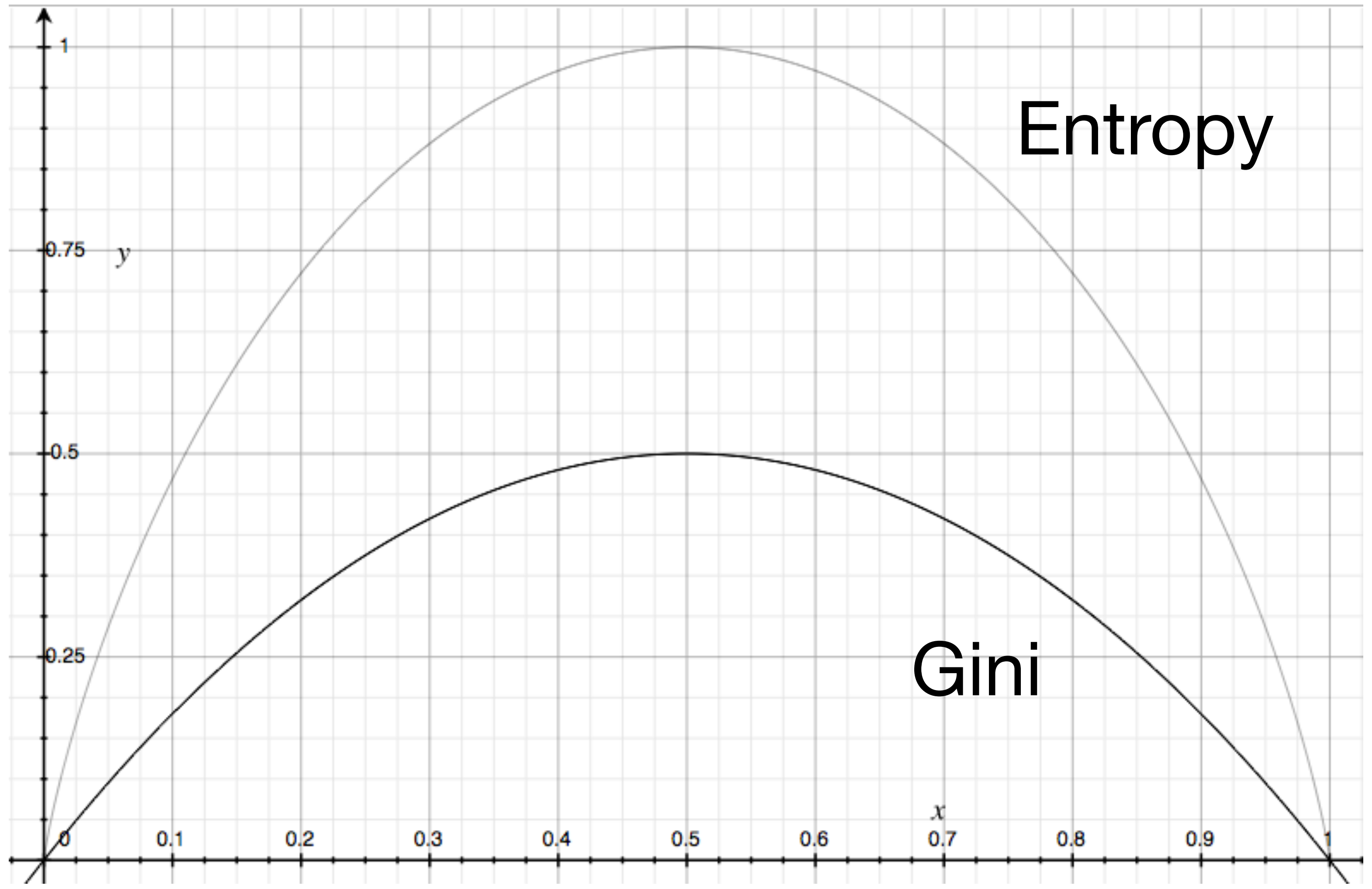


$$IG = 0$$

$$GG = 0$$

COMPARING INFORMATION GAIN TO GINI GAIN





$$IG = 1.0 - \left[\frac{2}{12}\, 0\right] - \left[\frac{4}{12}\, 0\right] - \left[\frac{6}{12}\, 0.919\right] = 0.541$$

$$GG = 0.5 - \left[\frac{2}{12}\, 0\right] - \left[\frac{4}{12}\, 0\right] - \left[\frac{6}{12}\, 0.444\right] = 0.278$$

ADDITIONAL ATTRIBUTE SELECTION CRITERIA: CHI-SQUARE SCORE

- Widely used to test independence between two categorical attributes (e.g., feature and class label)
- Considers counts in a contingency table and calculates the normalized squared deviation of observed (actual) values from expected (predicted) values

$$\chi^2 = \sum_{i=1}^{k} \frac{(o_i - e_i)^2}{e_i}$$

- Sampling distribution is known to be chi-square distributed, given that cell counts are above minimum thresholds

CALCULATING EXPECTED VALUES FOR A CELL

$$\chi^2 = \sum_{i=1}^{k} \frac{(o_i - e_i)^2}{e_i}$$

		Class +	Class -
Attribute	0	a	b
Attribute	1	c	d
			N

$$o_{(0,+)} = a$$

$$\begin{aligned} e_{(0,+)} &= p(A=0, C=+) \cdot N \\ &= p(A=0)p(C=+|A=0) \cdot N \\ &= p(A=0)p(C=+) \cdot N \qquad \text{(assuming independence)} \\ &= \left[\frac{a+b}{N}\right] \cdot \left[\frac{a+c}{N}\right] \cdot N \end{aligned}$$

EXAMPLE CALCULATION

Observed

	Buy	No buy
High	2	2
Med	4	2
Low	3	1

Expected

	Buy	No buy
High	2.57	1.43
Med	3.86	2.14
Low	2.57	1.43

$$\chi^2 = \sum_{i=1}^{k} \frac{(o_i - e_i)^2}{e_i} = \left(\frac{(2-2.57)^2}{2.57}\right) + \left(\frac{(4-3.86)^2}{3.86}\right) + \left(\frac{(3-2.57)^2}{2.57}\right)$$

$$+ \left(\frac{(2-1.43)^2}{1.43}\right) + \left(\frac{(2-2.14)^2}{2.14}\right) + \left(\frac{(1-1.43)^2}{1.43}\right)$$

$$= 0.57$$

WHEN TO STOP GROWING

- Full growth methods
  - There are no examples left
  - All examples at a node belong to the same class
  - There are no attributes left for further splits
- What impact does this have on the quality of the learned trees?
  - Trees **overfit** the training data and accuracy on testing data suffers

OVERFITTING

- Consider a distribution D of data representing a population and a sample D_S drawn from D, which is used as training data
- Given a model space M, a score function S, and a learning algorithm that returns a model $m \in M$, the algorithm **overfits** the training data D_S if:
  $\exists m' \in M$ such that $S(m, D_S) > S(m', D_S)$ but $S(m, D) < S(m', D)$
  - In other words, there is another model (m′) that is better on the entire distribution and if we had learned from the full data we would have selected it instead

HOW TO AVOID OVERFITTING IN DECISION TREES

- Post-pruning
  - Separate the training data into a training set and a validation set (i.e., a pruning set).
  - Fully grow a tree
  - Use the pruning set to evaluate the utility of pruning (i.e. deleting) nodes from the tree
- Pre-pruning
  - Apply a statistical test to decide whether to expand a node
  - Add penalty terms in scoring functions to prefer trees with smaller sizes