Data Mining & Machine Learning

CS37300
Purdue University

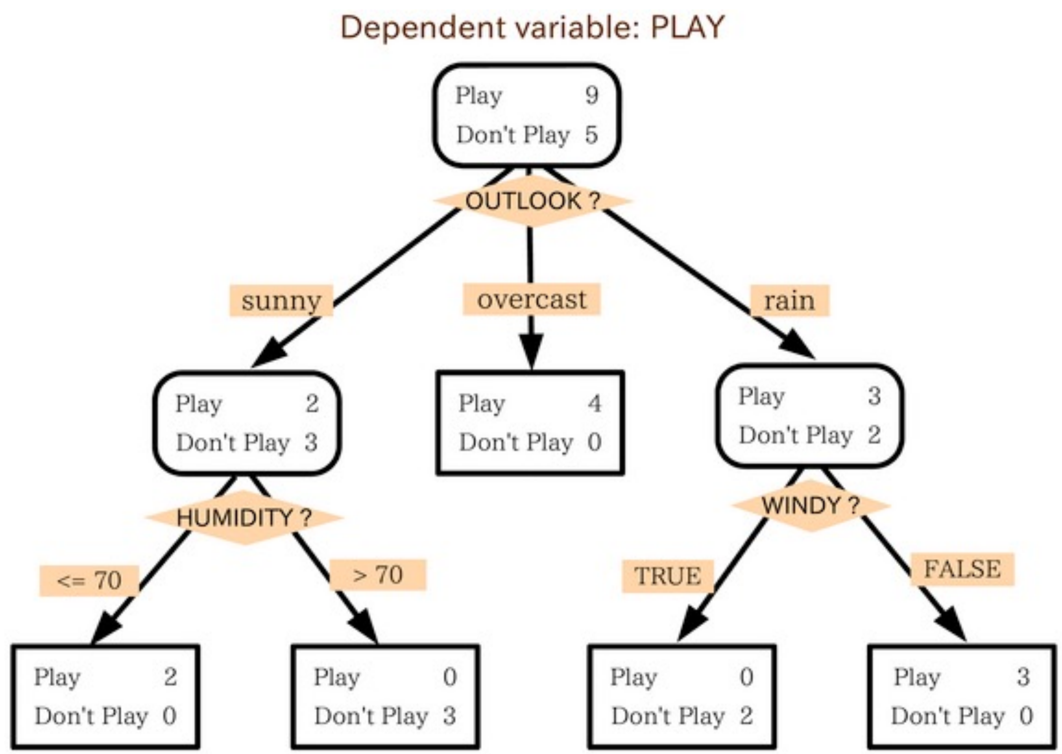
Sep 11, 2023

Today's topics

- Classification trees

Tree models

- Easy to understand
- Can handle continuous and discrete/categorical variables
- Recursive, divide and conquer learning method
- Efficient prediction for test samples



Tree learning

- Top-down recursive divide and conquer algorithm
  - Start with all samples at root
  - ***Select best feature***
  - Partition samples by selected feature
  - Recurse and repeat
- Other issues:
  - When to stop growing
  - Pruning irrelevant parts of the tree

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y
+	24	N	2006	N
-	29	N	2003	N

Y **N**

choose split on *Series7*

Fraud	Age	Degree	StartYr	FinHistory
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
+	24	N	2006	N
-	29	N	2003	N

Y **N**

choose split on *Age>28*

Fraud	Age	Degree	StartYr	FinHistory
-	29	N	2003	N

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
+	24	N	2006	N

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y
+	24	N	2006	N
-	29	N	2003	N

Score each feature split (*Age, Degree, StartYr,* FinHistory) for these samples

Y N

choose split on *Series7*

Fraud	Age	Degree	StartYr	FinHistory
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
+	24	N	2006	N
-	29	N	2003	N

Y N

choose split on *Age>28*

Fraud	Age	Degree	StartYr	FinHistory
-	29	N	2003	N

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
+	24	N	2006	N

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y
+	24	N	2006	N
-	29	N	2003	N

Y N

choose split on *Series7*

Y:

Fraud	Age	Degree	StartYr	FinHistory
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y

N:

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
+	24	N	2006	N
-	29	N	2003	N

Y N

choose split on *Age>28*

Y:

Fraud	Age	Degree	StartYr	FinHistory
-	29	N	2003	N

N:

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
+	24	N	2006	N

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y
+	24	N	2006	N
-	29	N	2003	N

Y

N

choose split on *Series7*

Fraud	Age	Degree	StartYr	FinHistory
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
+	24	N	2006	N
-	29	N	2003	N

Score each feature split (*Age, Degree, StartYr*) for these samples

Y

N

choose split on *Age>28*

Fraud	Age	Degree	StartYr	FinHistory
-	29	N	2003	N

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
+	24	N	2006	N

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y
+	24	N	2006	N
-	29	N	2003	N

Y N

choose split on *Series7*

Fraud	Age	Degree	StartYr	FinHistory
-	25	N	2003	Y
-	31	Y	1995	Y
-	27	Y	1999	Y

Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
+	24	N	2006	N
-	29	N	2003	N

Y N

choose split on *Age>28*

Fraud	Age	Degree	StartYr	FinHistory
-	29	N	2003	N

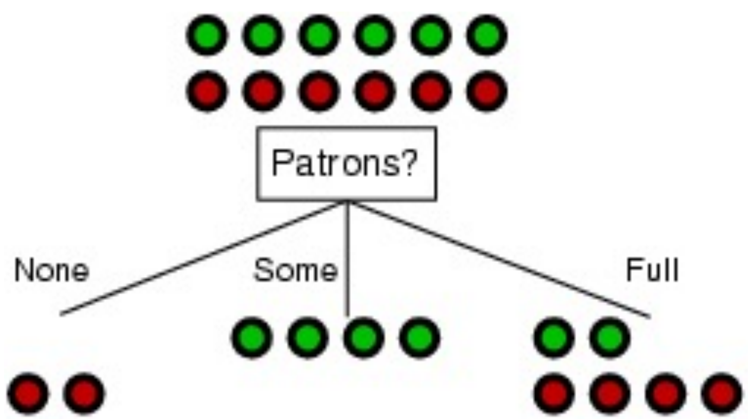
Fraud	Age	Degree	StartYr	FinHistory
+	22	Y	2005	N
+	24	N	2006	N

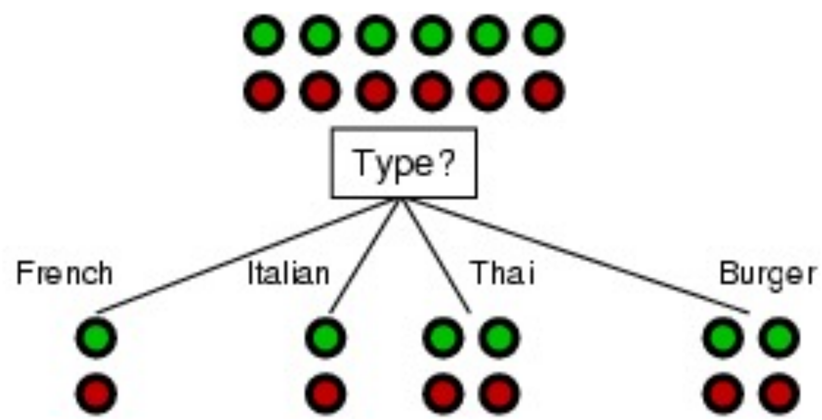
Tree models

- Most well-known systems
  - CART (Classification and Regression Trees)
  - C4.5
- How do they differ?
  - **Split scoring function**
  - Stopping criterion
  - Pruning mechanism

Choosing a feature

- Idea: a good feature splits the samples into subsets that distinguish among the class labels as much as possible, ideally into pure sets of "all positive" or "all negative"



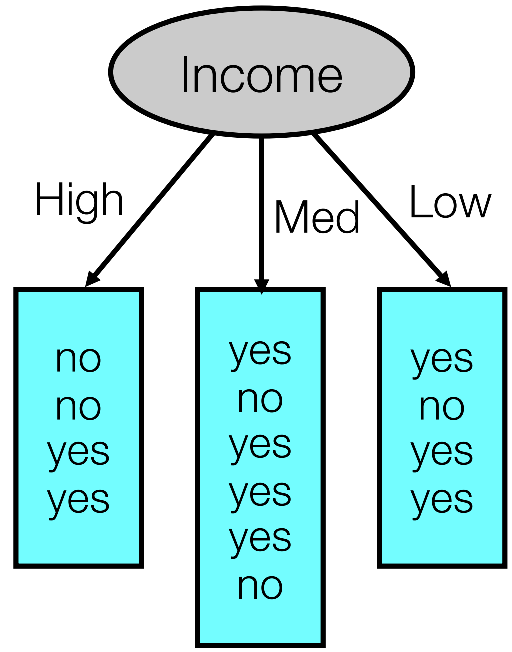


(Yelp data for restaurants)

Association between feature 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

Feature value

	BC=yes	BC=no
High	2	2
Med	4	2
Low	3	1

Entropy

- Quantifies the amount of randomness (unpredictability) of a probability distribution.

- Definition: The **entropy** H(X) of a discrete random variable X is defined by:

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

Entropy

- Quantifies the amount of randomness (unpredictability) of a probability distribution.

- Definition: The **entropy** H(X) of a discrete random variable X is defined by:

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

- Comes from Information Theory:
  Expresses optimal expected number of bits needed to communicate the value of X to another person

Entropy

- Quantifies the amount of randomness (unpredictability) of a probability distribution.

- Definition: The **entropy** H(X) of a discrete random variable X is defined by:

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

- Comes from Information Theory:
  Expresses optimal expected number of bits needed to communicate the value of X to another person

- Another interpretation:
  Expresses **amount of uncertainty** we have (a priori) about the value of X

Entropy of a random variable

An unbiased coin with 50% being 1, 50% being 0, has entropy:

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

Entropy of a random variable

An unbiased coin with 50% being 1, 50% being 0, has entropy:

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

A deterministic coin with 100% being 1, 0% being 0, has entropy:

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

Entropy of a random variable

An unbiased coin with 50% being 1, 50% being 0, has entropy:

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

A deterministic coin with 100% being 1, 0% being 0, has entropy:

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

A biased coin with 75% being 1, 25% being 0, has entropy:

$H(X) = 0.811$

Entropy of a random variable

An unbiased coin with 50% being 1, 50% being 0, has entropy:

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

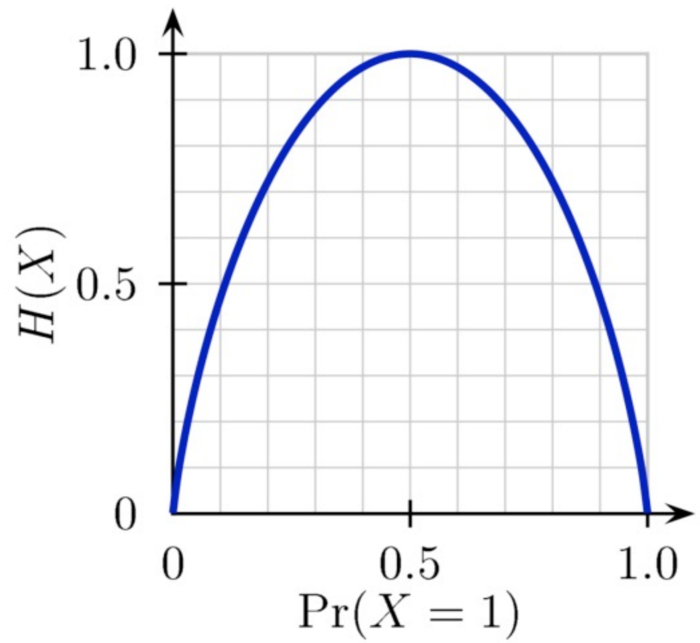
A deterministic coin with 100% being 1, 0% being 0, has entropy:

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

A biased coin with 75% being 1, 25% being 0, has entropy:

$H(X) = 0.811$

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



Information gain

- How much does a feature split decrease the entropy?
- Called **Information Gain**

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

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

Information gain

- How much does a feature split decrease the entropy?
- Called **Information Gain**

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

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

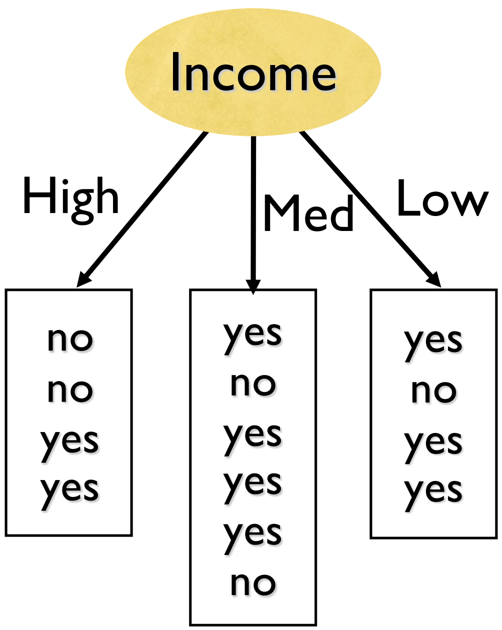
Entropy(BC)

$= -9/14 \log_2 9/14 - 5/14 \log_2 5/14$

$= 0.940$

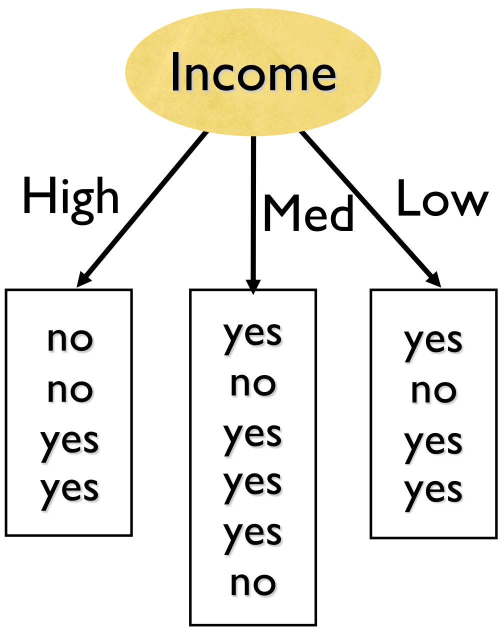
Information gain

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



Information gain

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

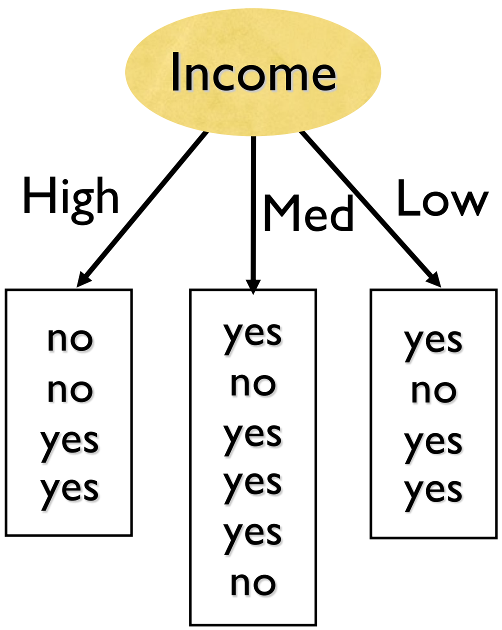


$Entropy(BC_{Income=high})$

$Gain(BC,Income)$
= 0.940 - (4/14 [1] + 6/14 [0.918] + 4/14 [0.811])
= 0.029

Information gain

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

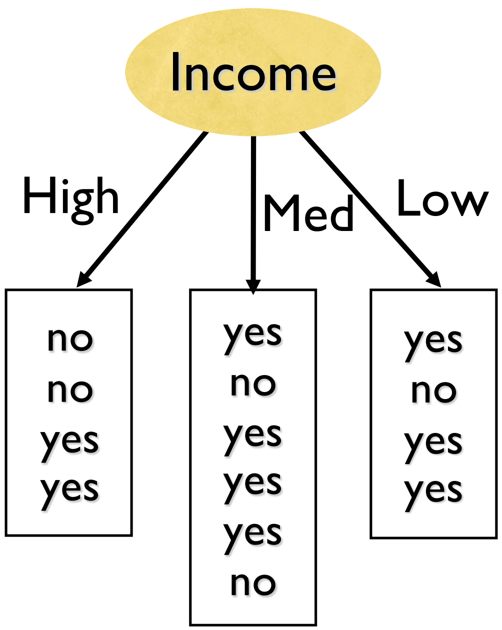


$Entropy(BC_{Income=high})$
$= -2/4 \log_2 2/4 - 2/4 \log_2 2/4 = 1$

$Gain(BC, Income)$
$= 0.940 - (4/14\ [1] + 6/14\ [0.918] + 4/14\ [0.811])$
$= 0.029$

Information gain

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



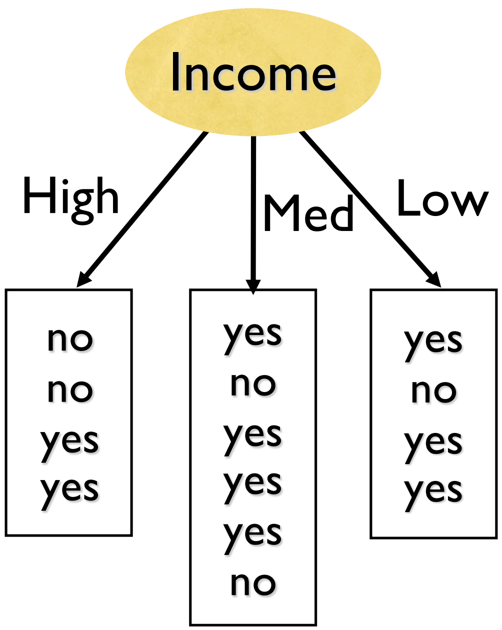
$Entropy(BC_{Income=high})$
$= -2/4 \log_2 2/4 - 2/4 \log_2 2/4 = 1$

$Entropy(BC_{Income=med})$

Gain(BC,Income)
$= 0.940 - (4/14\ [1] + 6/14\ [0.918] + 4/14\ [0.811])$
$= 0.029$

Information gain

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



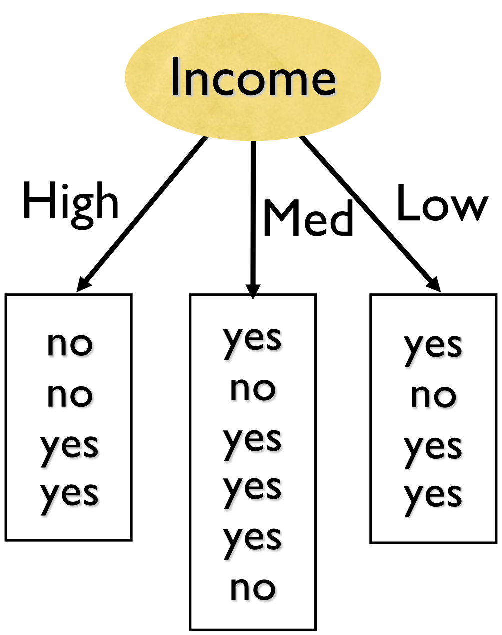
$Entropy(BC_{Income=high})$
$= -2/4 \log_2 2/4 - 2/4 \log_2 2/4 = 1$

$Entropy(BC_{Income=med})$
$= -4/6 \log_2 4/6 - 2/6 \log_2 2/6 = 0.918$

Gain(BC,Income)
$= 0.940 - (4/14\ [1] + 6/14\ [0.918] + 4/14\ [0.811])$
$= 0.029$

Information gain

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



$Entropy(BC_{Income=high})$

$= -2/4 \log_2 2/4 - 2/4 \log_2 2/4 = 1$

$Entropy(BC_{Income=med})$

$= -4/6 \log_2 4/6 - 2/6 \log_2 2/6 = 0.918$

$Entropy(BC_{Income=low})$

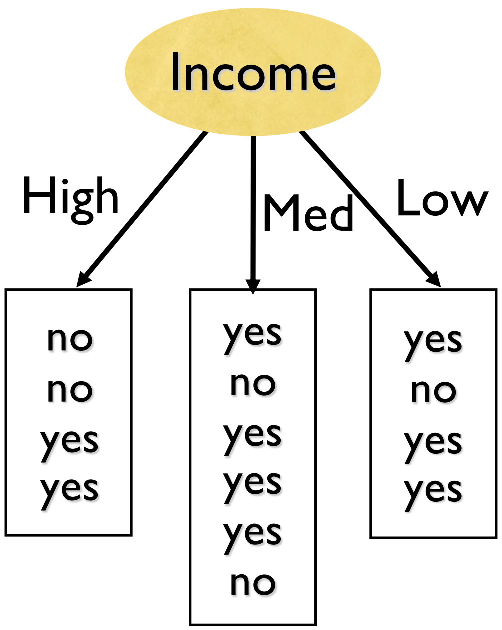
$Gain(BC, Income)$

$= 0.940 - (4/14\ [1] + 6/14\ [0.918] + 4/14\ [0.811])$

$= 0.029$

Information gain

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



$Entropy(BC_{Income=high})$
$= -2/4 \log_2 2/4 - 2/4 \log_2 2/4 = 1$

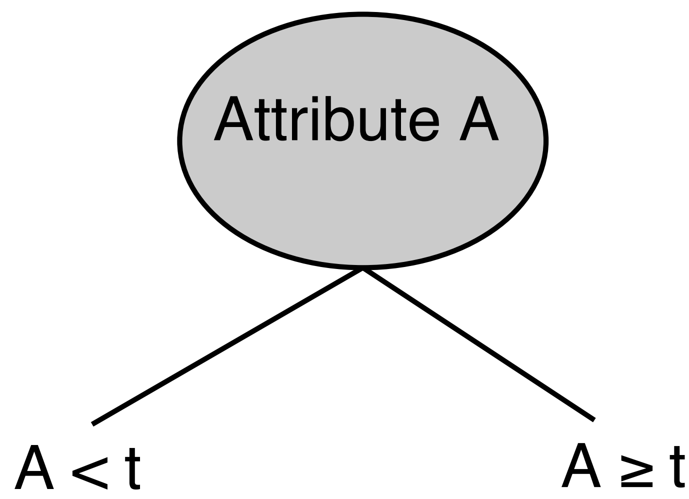
$Entropy(BC_{Income=med})$
$= -4/6 \log_2 4/6 - 2/6 \log_2 2/6 = 0.918$

$Entropy(BC_{Income=low})$
$= -3/4 \log_2 3/4 - 1/4 \log_2 1/4 = 0.811$

$Gain(BC, Income)$
$= 0.940 - (4/14\ [1] + 6/14\ [0.918] + 4/14\ [0.811])$
$= 0.029$

Continuous attributes

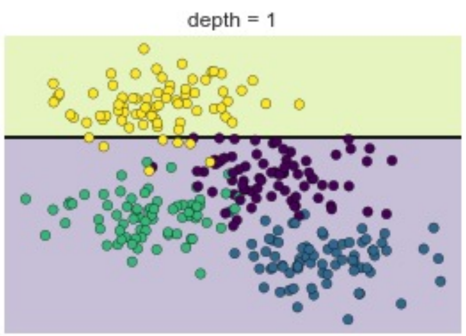
- Can’t split on the attribute values: infinite number of them.
- Instead, pick a **threshold** t



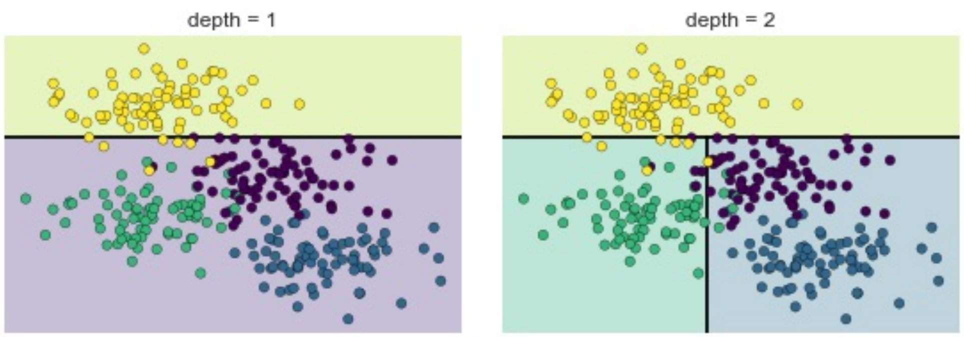
- Pick the A and t that give highest information gain

Continuous attributes

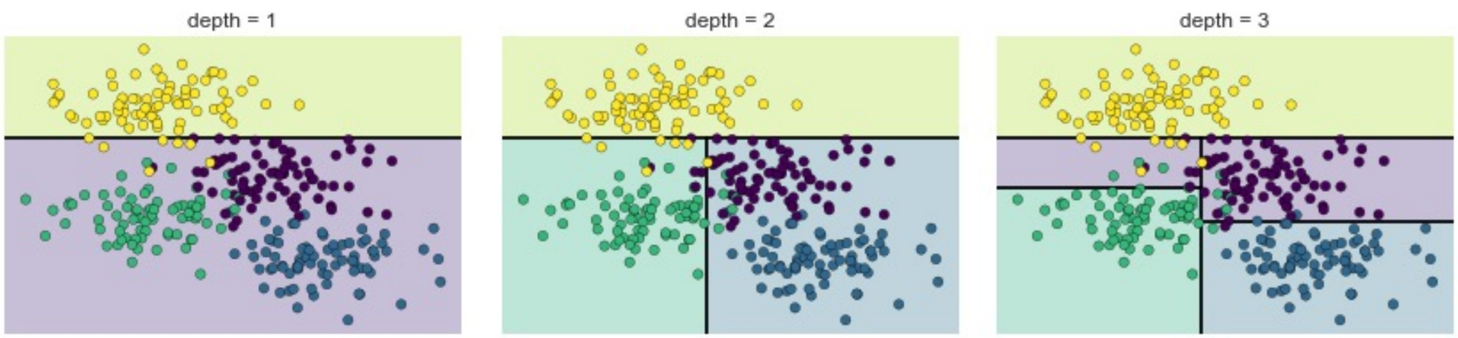
Continuous attributes



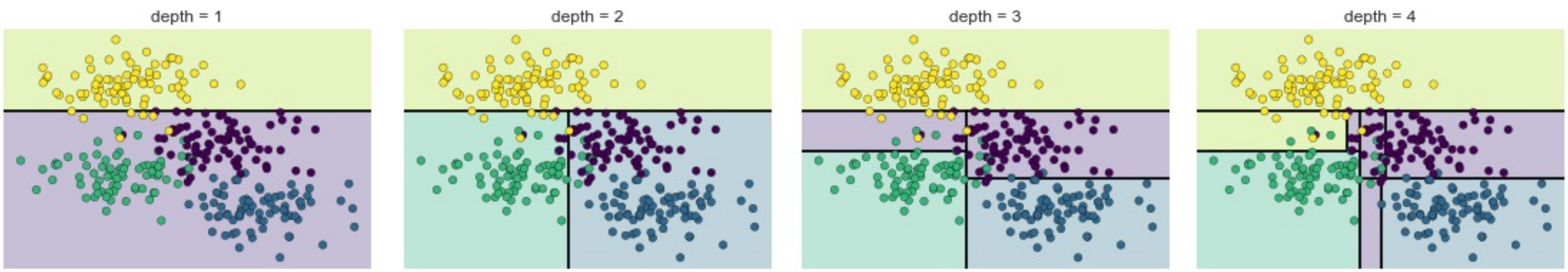
Continuous attributes



Continuous attributes



Continuous attributes



Gini gain

- Another option: split to maximize **Gini gain**
- Similar to information gain
- Uses **gini index** instead of entropy

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

Gini gain

- Another option: split to maximize **Gini gain**
- Similar to information gain
- Uses **gini index** instead of entropy

$$Gini(X) = 1 - \sum_x p(x)^2$$

- Intuition:
  - It's the probability that two independent samples X_1,X_2 have different values

Gini gain

- Another option: split to maximize **Gini gain**
- Similar to information gain
- Uses **gini index** instead of entropy

$$Gini(X) = 1 - \sum_x p(x)^2$$

- Intuition:
  - It's the probability that two independent samples X_1,X_2 have different values

$$\begin{aligned} P(X_1 = X_2) &= \sum_x P(X_1 = x \wedge X_2 = x) \\ &= \sum_x P(X_1 = x)P(X_2 = x) = \sum_x p(x)p(x) = \sum_x p(x)^2 \end{aligned}$$

Gini gain

- Another option: split to maximize **Gini gain**
- Similar to information gain
- Uses **gini index** instead of entropy

$$Gini(X) = 1 - \sum_x p(x)^2$$

- Intuition:
  - It's the probability that two independent samples X_1,X_2 have different values

$$P(X_1 = X_2) = \sum_x P(X_1 = x \wedge X_2 = x)$$
$$= \sum_x P(X_1 = x)P(X_2 = x) = \sum_x p(x)p(x) = \sum_x p(x)^2$$

$$P(X_1 \neq X_2) = 1 - P(X_1 = X_2) = 1 - \sum_x p(x)^2$$

Gini gain

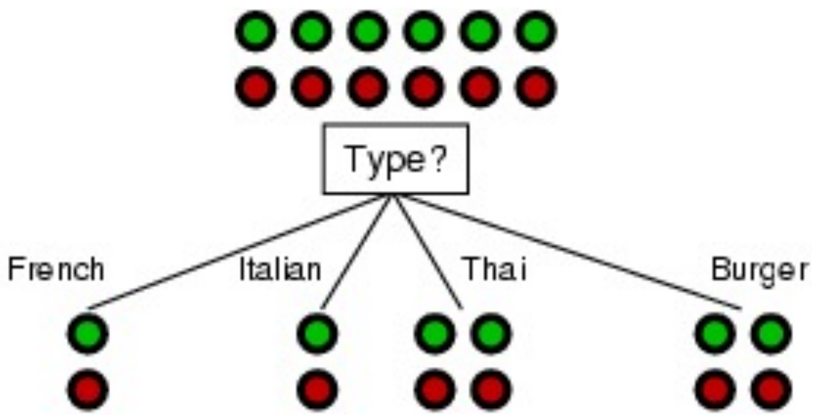
- Another option: split to maximize **Gini gain**
- Similar to information gain
- Uses **gini index** instead of entropy

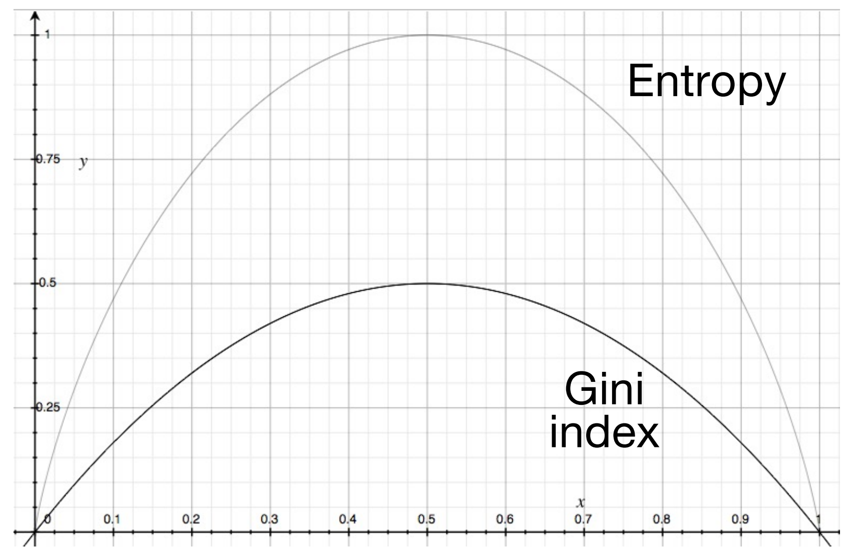
$$Gini(X) = 1 - \sum_x p(x)^2$$

- Intuition:
  - It's the probability that two independent samples X_1,X_2 have different values
- **Gini Gain** Measures decrease in gini index after split:

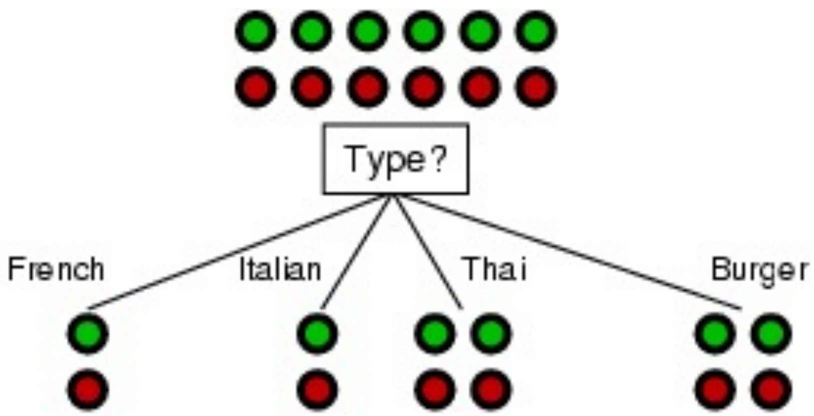
$$Gain(S,A) = Gini(S) - \sum_{a \in values(A)} \frac{|S_{A=a}|}{|S|} Gini(S_{A=a})$$

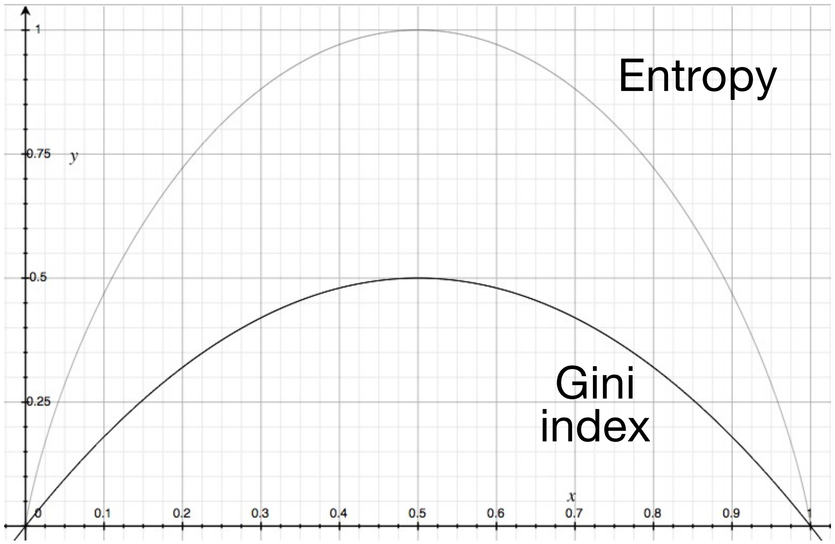
Comparing information gain to Gini gain





Comparing information gain to Gini gain

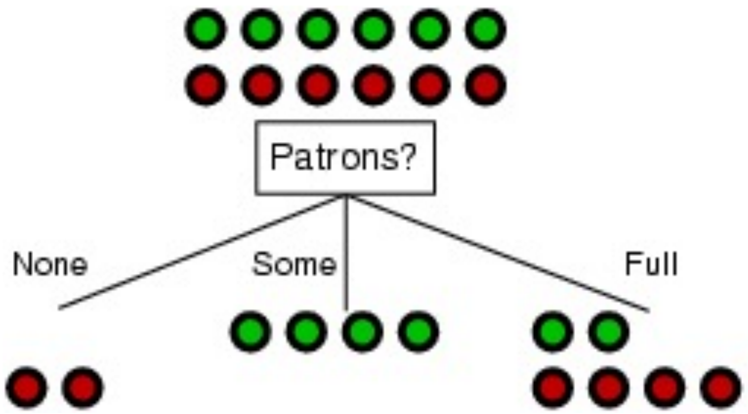


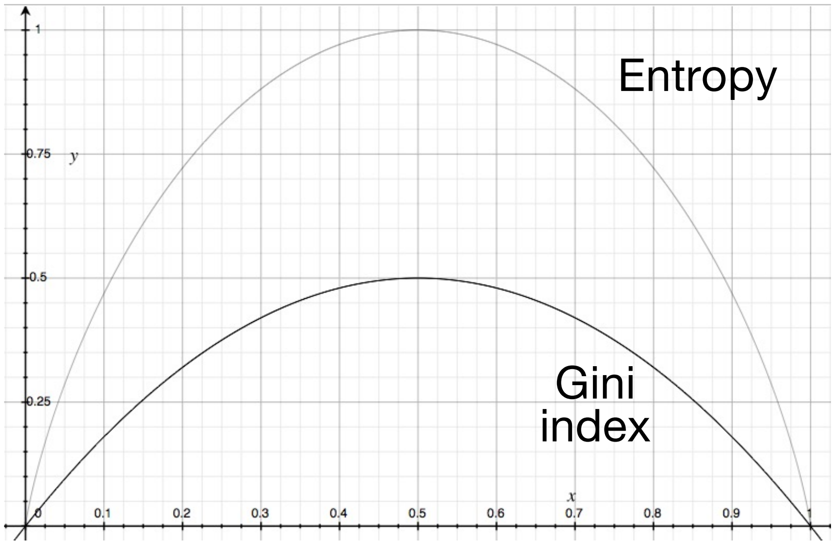


$$IG = 0$$

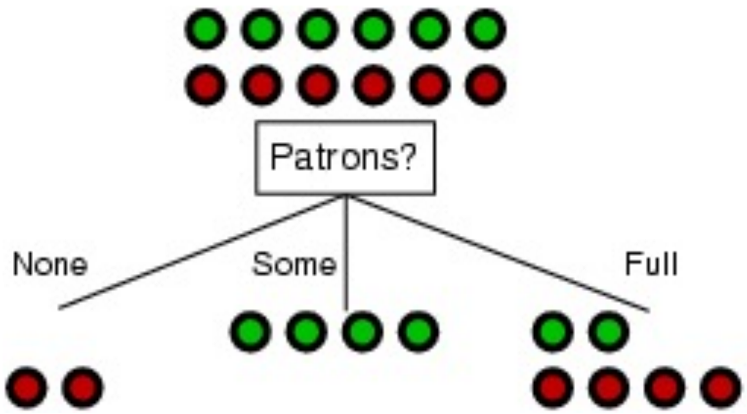
$$GG = 0$$

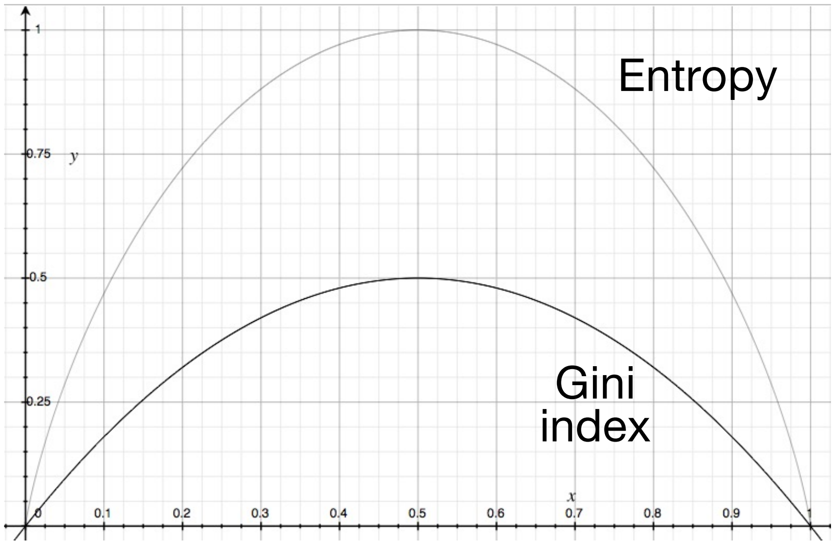
Comparing information gain to Gini gain





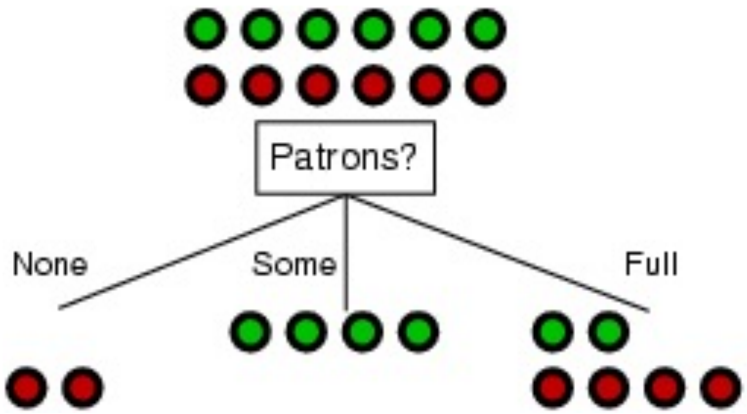
Comparing information gain to Gini gain

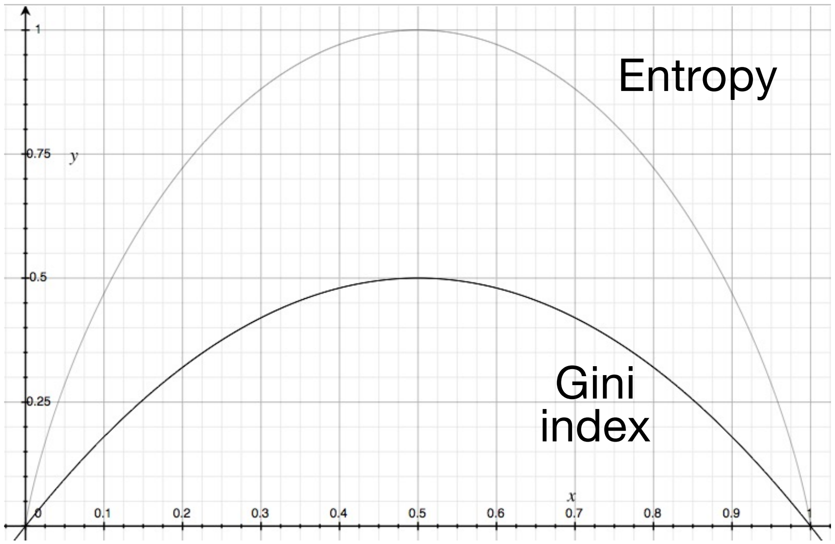




$$IG = 1.0 - \left[\frac{2}{12}0\right] - \left[\frac{4}{12}0\right] - \left[\frac{6}{12}0.919\right] = 0.541$$

Comparing information gain to Gini gain

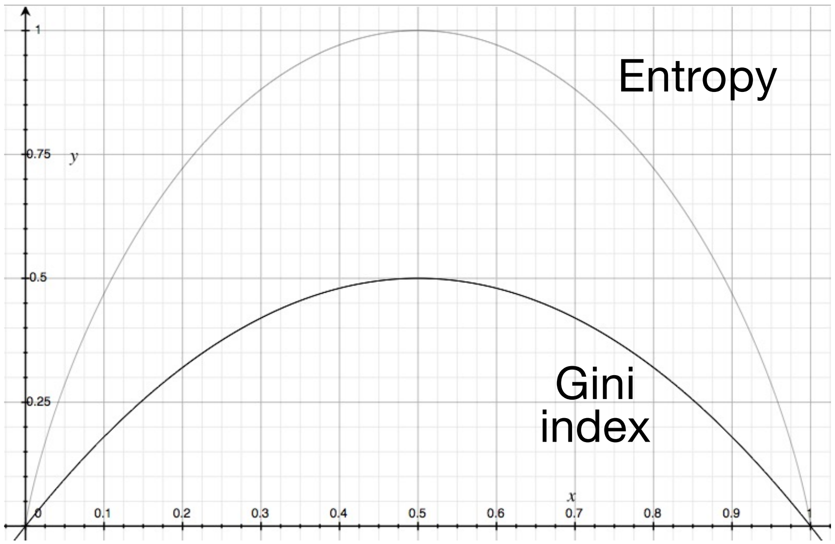




$$IG = 1.0 - \left[\frac{2}{12}\,0\right] - \left[\frac{4}{12}\,0\right] - \left[\frac{6}{12}\,0.919\right] = 0.541$$

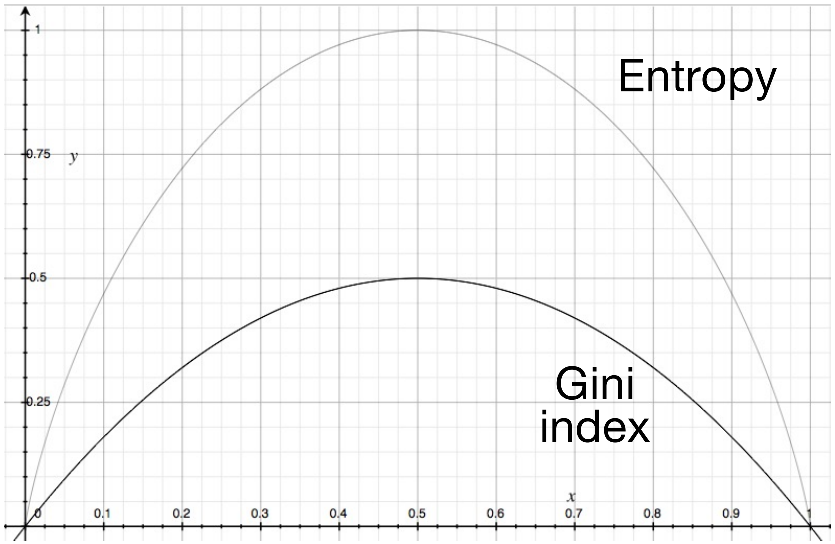
$$GG = 0.5 - \left[\frac{2}{12}\,0\right] - \left[\frac{4}{12}\,0\right] - \left[\frac{6}{12}\,0.444\right] = 0.278$$

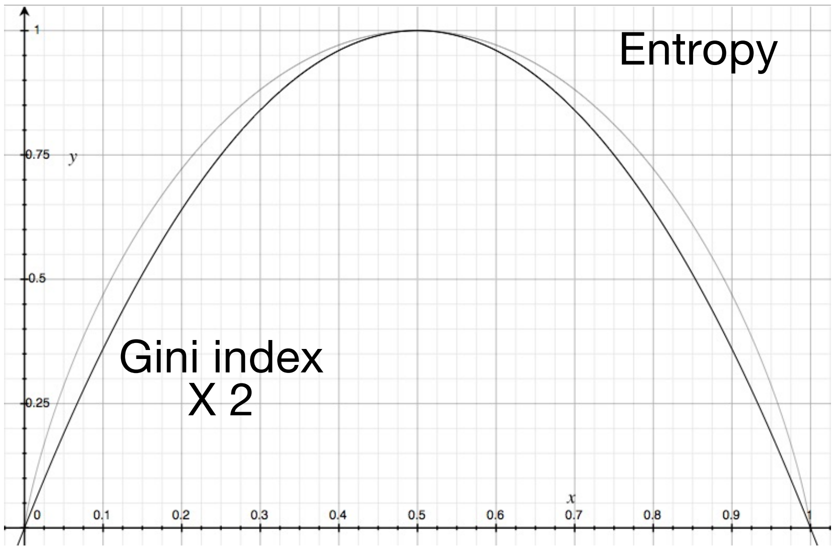
Comparing information gain to Gini gain



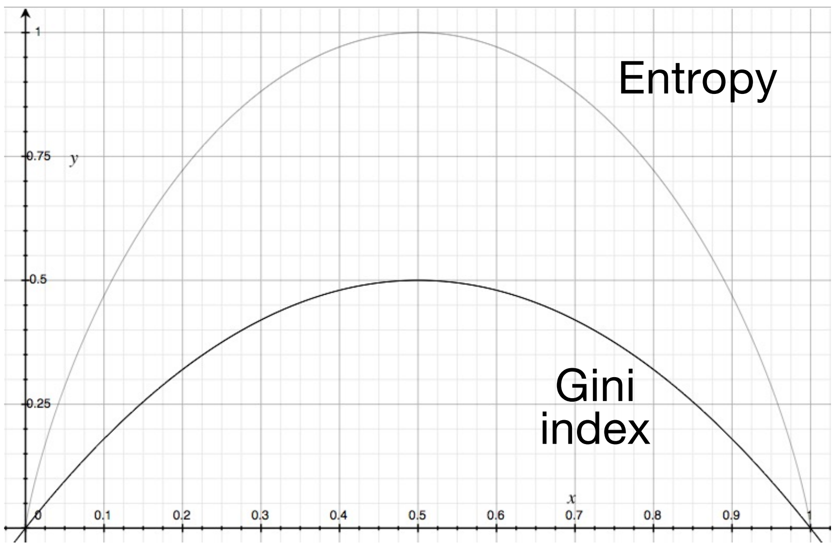
Entropy

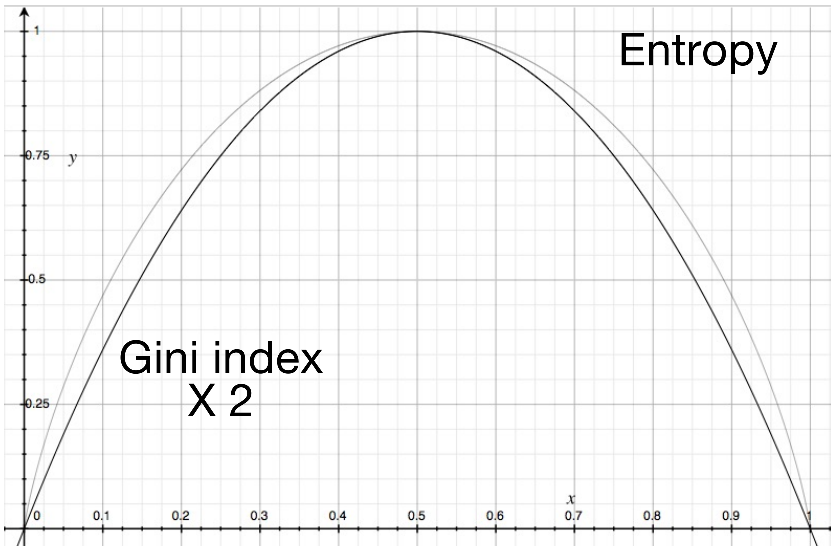
Comparing information gain to Gini gain





Comparing information gain to Gini gain





- The methods might often behave similarly
- When to use which one?
- Not clear. Often personal preference, software package convenience
- Can always just try both and decide based on the results.

Tree learning

- Top-down recursive divide and conquer algorithm
  - Start with all samples at root
  - Select best feature
  - Partition samples by selected feature
  - Recurse and repeat
- Other issues:
  - ***When to stop growing***
  - ***Pruning irrelevant parts of the tree***

When to stop growing

When to stop growing

- Full growth methods
  - All samples at a leaf node belong to the same class
  - There are no features left for further splits
  - There are no samples left

When to stop growing

- Full growth methods
    - All samples at a leaf node belong to the same class
    - There are no features left for further splits
    - There are no samples left
- What impact does this have on the quality of the learned trees?

When to stop growing

- Full growth methods
  - All samples at a leaf node belong to the same class
  - There are no features left for further splits
  - There are no samples left
- What impact does this have on the quality of the learned trees?
  - Trees **overfit** the data and accuracy decreases
  - Pruning is used to avoid overfitting

Pruning

- Postpruning
  - Use a separate set of samples to evaluate the utility of pruning nodes from the tree (after tree is fully grown)
- Prepruning
  - We decide stopping criteria before building the tree

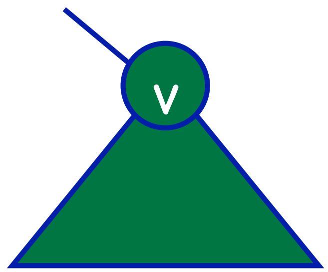
Source: www.ailab.si/blaz/predavanja/uisp/slides/uisp05-PostPruning.ppt

Post-pruning: Reduced error pruning

Preview of ideas from lecture next week

- Use **validation set** to estimate accuracy in sub-trees and for individual nodes

- Let T be a sub-tree rooted at node v



- Define:

$$\text{Gain from prunning at } v = \#\text{misclassification in } T - \#\text{misclassification at } v$$

- where #misclassifications is measured on the validation set

- Make a "bottom-up" pass: start with the deepest nodes, then 2nd-deepest, etc.

- As we go through them, if a node has Gain ≥ 0, prune it, else don't prune it

Pre-pruning

- Stop growing tree at some point during top-down construction when there is no longer sufficient data to make reliable decisions
- Approach:
  - Choose threshold on feature score (information gain, Gini gain)
  - Stop splitting if the best feature score is below threshold

Algorithm comparison

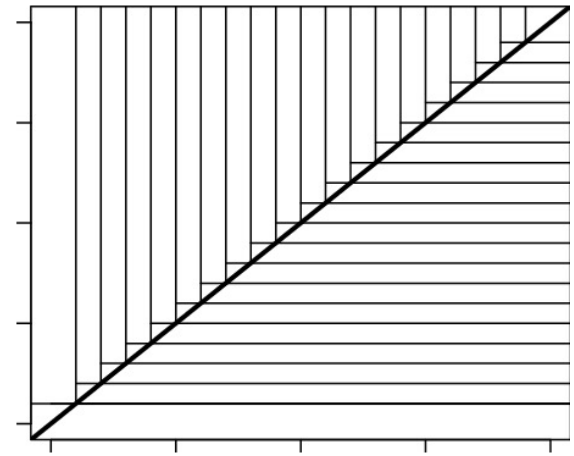
- CART
  - Evaluation criterion: **Gini gain**
  - Pruning mechanism: **Cross-validation to select gini threshold**
- C4.5
  - Evaluation criterion: **Information gain**
  - Pruning mechanism: **Reduced error pruning**

Decision Trees vs kNN

- Decision Trees
  - Need meaningful attributes
  - Need that classifications typically don't depend on knowing all of the attributes (to avoid a really deep tree)
  - It's good for discrete attributes
- kNN
  - Good when nearby points likely to have same label (meaningful distances)
  - Might not be as good for discrete attributes
    (distances are less meaningful, and not very small)

Decision Trees vs kNN

- Decision Trees
    - Need meaningful attributes
    - Need that classifications typically don't depend on knowing all of the attributes (to avoid a really deep tree)
    - It's good for discrete attributes



- kNN
    - Good when nearby points likely to have same label (meaningful distances)
    - Might not be as good for discrete attributes
      (distances are less meaningful, and not very small)

Decision Trees vs kNN

- Decision Trees
  - Need meaningful attributes
  - Need that classifications typically don’t depend on knowing all of the attributes (to avoid a really deep tree)
  - It’s good for discrete attributes
- kNN
  - Good when nearby points likely to have same label (meaningful distances)
  - Might not be as good for discrete attributes
    (distances are less meaningful, and not very small)