11. Decision Trees

Decision Trees: Perspective

Decision Trees

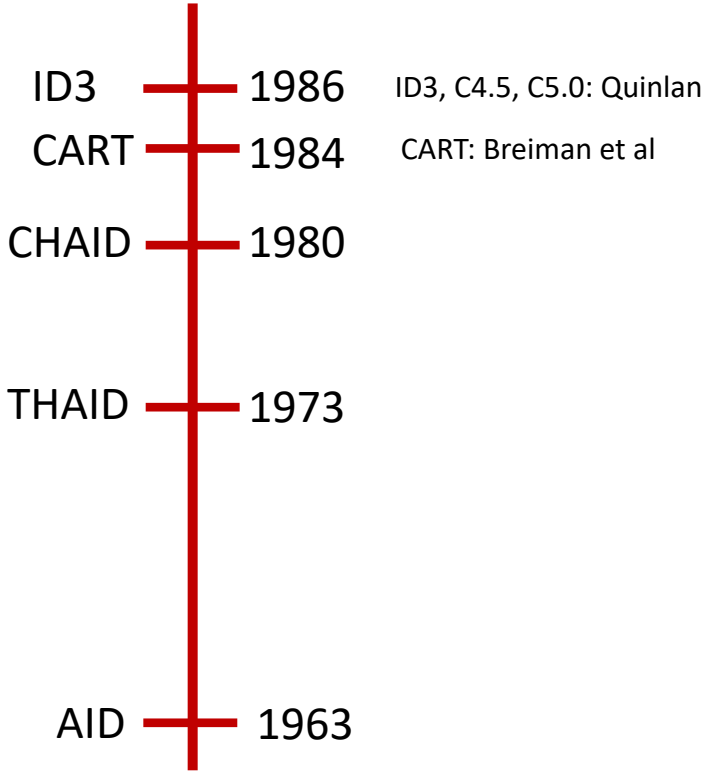
These are a very popular inductive inference algorithm used in various applications

These use attributes associated with training examples (instances & labels) in order to develop a decision tree that can be used to predict a new label for a new instance.

Decision trees predict a label for a new instance by filtering the instance attributes down the decision tree..

Decision trees are based on “divide and conquer” principles

Brief History



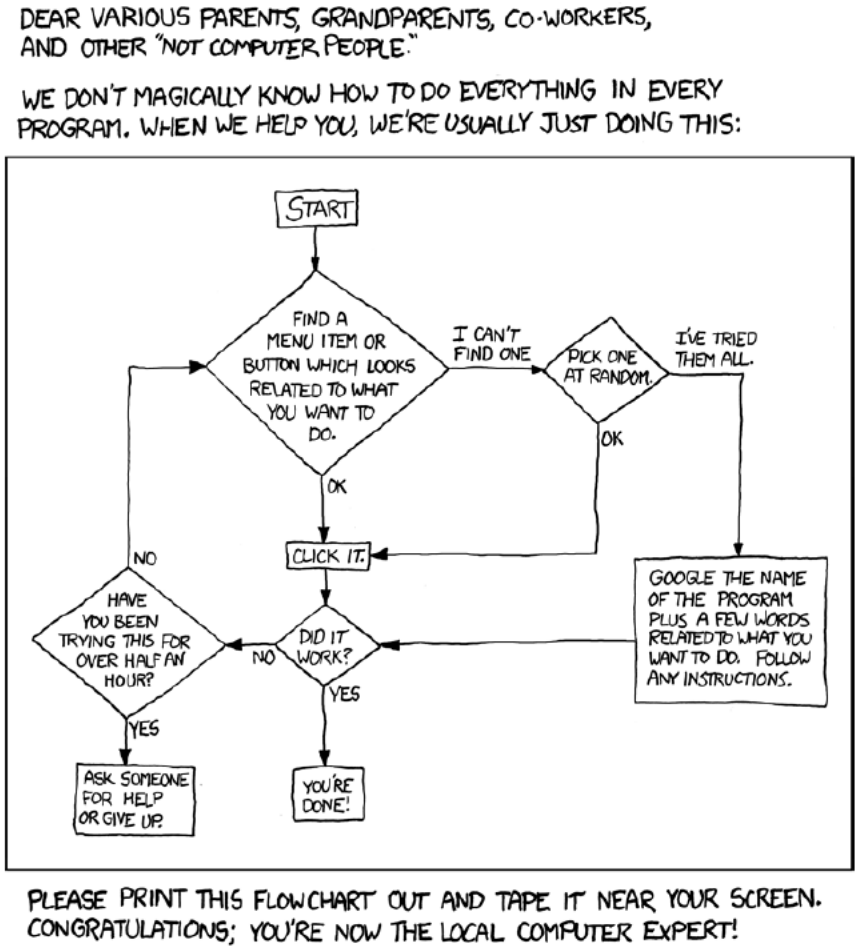
Decision Trees: Motivation

Motivation

Decision trees are modelled after how humans make decisions:

- Decision trees consider a variety of factors
- Decision trees also consider a logical path of queries

Decision trees are very popular in real world applications because they mimic how individuals think.



Source: https://xkcd.com/627/

Decision Trees: Example

Dataset

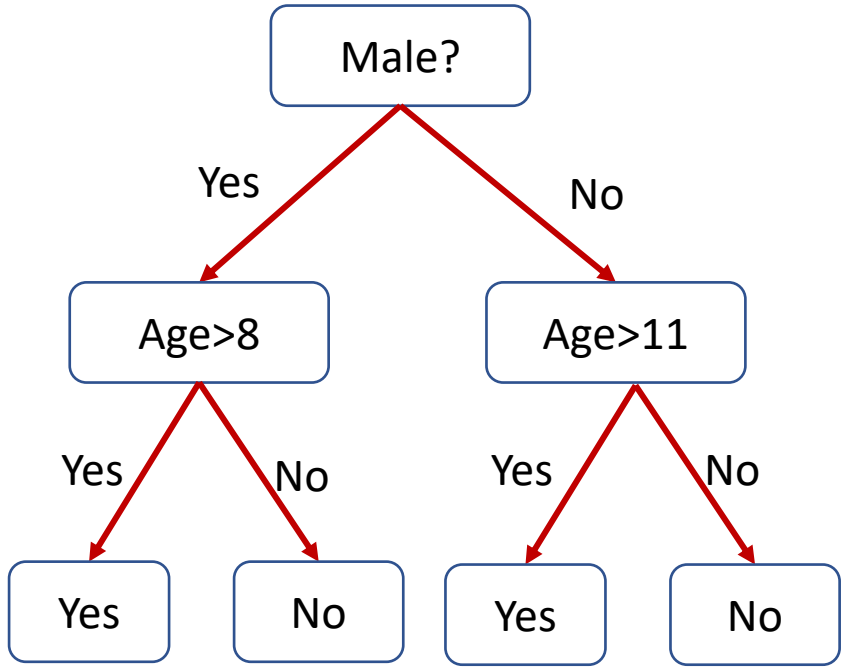
The dataset reports on gender, age and height (> or < than 55’’) of different individuals.

.

Person	Age	Male	Height>55’’
Alice	14	0	1
Bob	10	1	1
Carol	13	0	1
Dave	8	1	0
Erin	11	0	0
Frank	9	1	1
Gena	10	0	0

Decision Tree

This decision tree that can be used to predict whether or not height is greater than 55’’ based on features like gender and age.



How to Construct Decision Trees?

Considerations

One ought to construct smaller rather than larger trees because… simpler is better (Occam's razor).

However, the construction of an (optimal) decision tree for some given dataset is computationally intractable.

The construction of decision trees therefore relies on the use of greedy heuristics

Source: Wikipedia

"We consider it a good principle to explain the phenomena by the simplest hypothesis possible." Ptolemy

How to Construct Decision Trees?

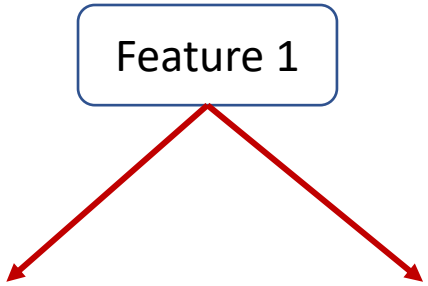
Greedy Algorithm

- Start from an empty decision tree
- Split on next best attribute
- Recurse

How to Construct Decision Trees?

Greedy Algorithm

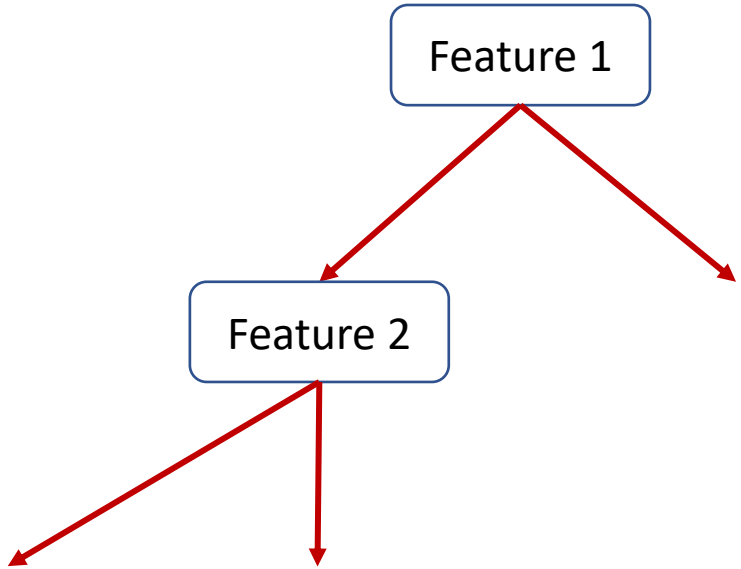
- Start from an empty decision tree
- Split on next best attribute
- Recurse



How to Construct Decision Trees?

Greedy Algorithm

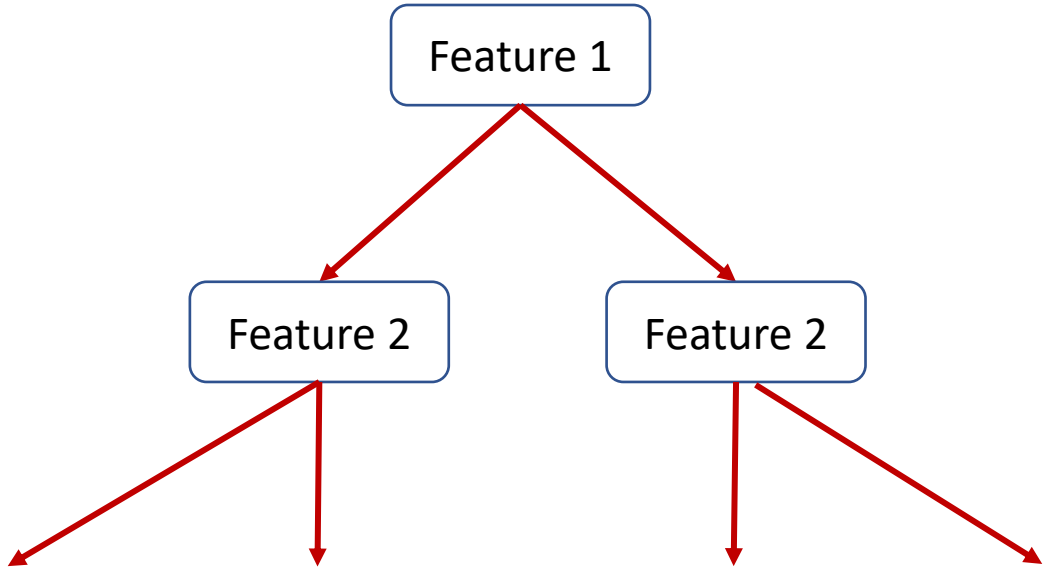
- Start from an empty decision tree
- Split on next best attribute
- Recurse



How to Construct Decision Trees?

Greedy Algorithm

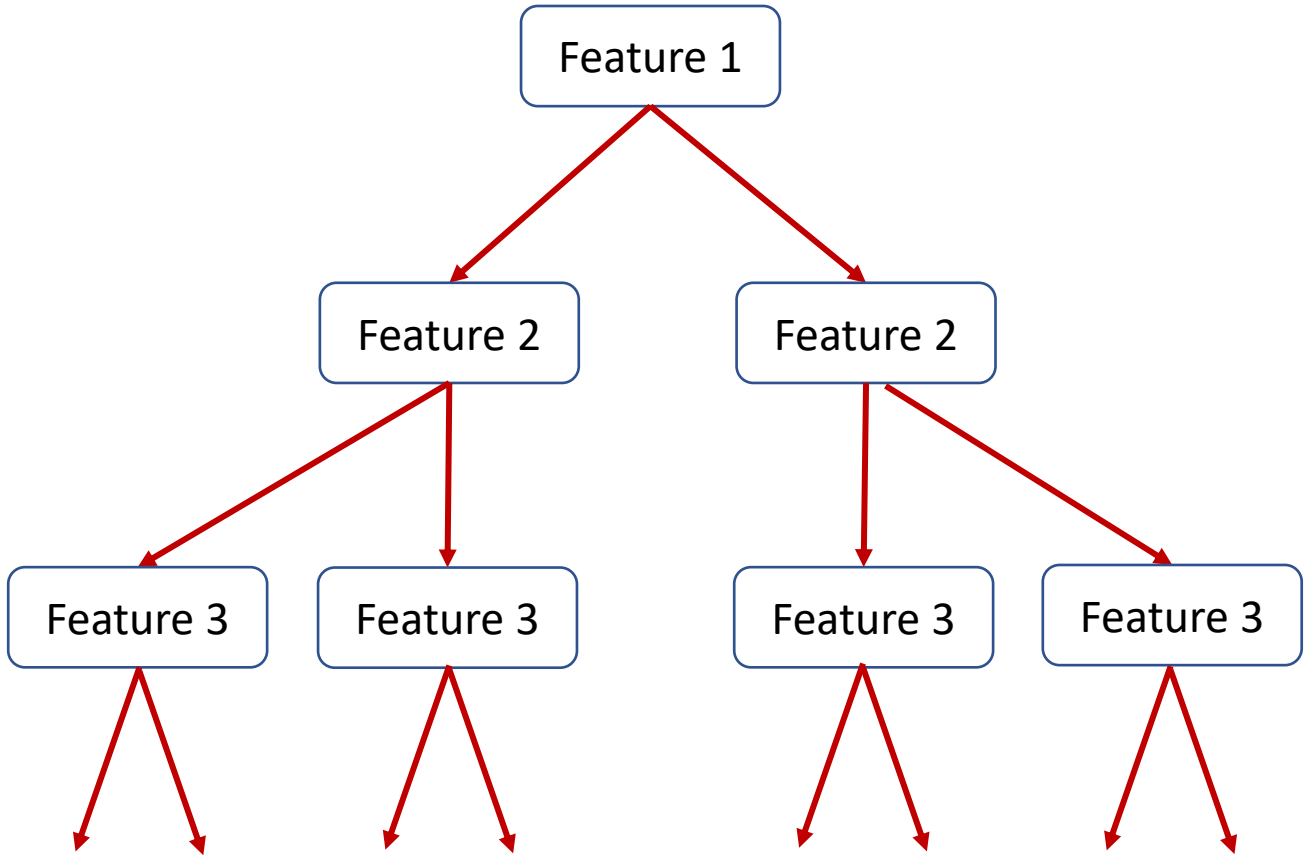
- Start from an empty decision tree
- Split on next best attribute
- Recurse



How to Construct Decision Trees?

Greedy Algorithm

- Start from an empty decision tree
- Split on next best attribute
- Recurse



How to Construct Decision Trees?

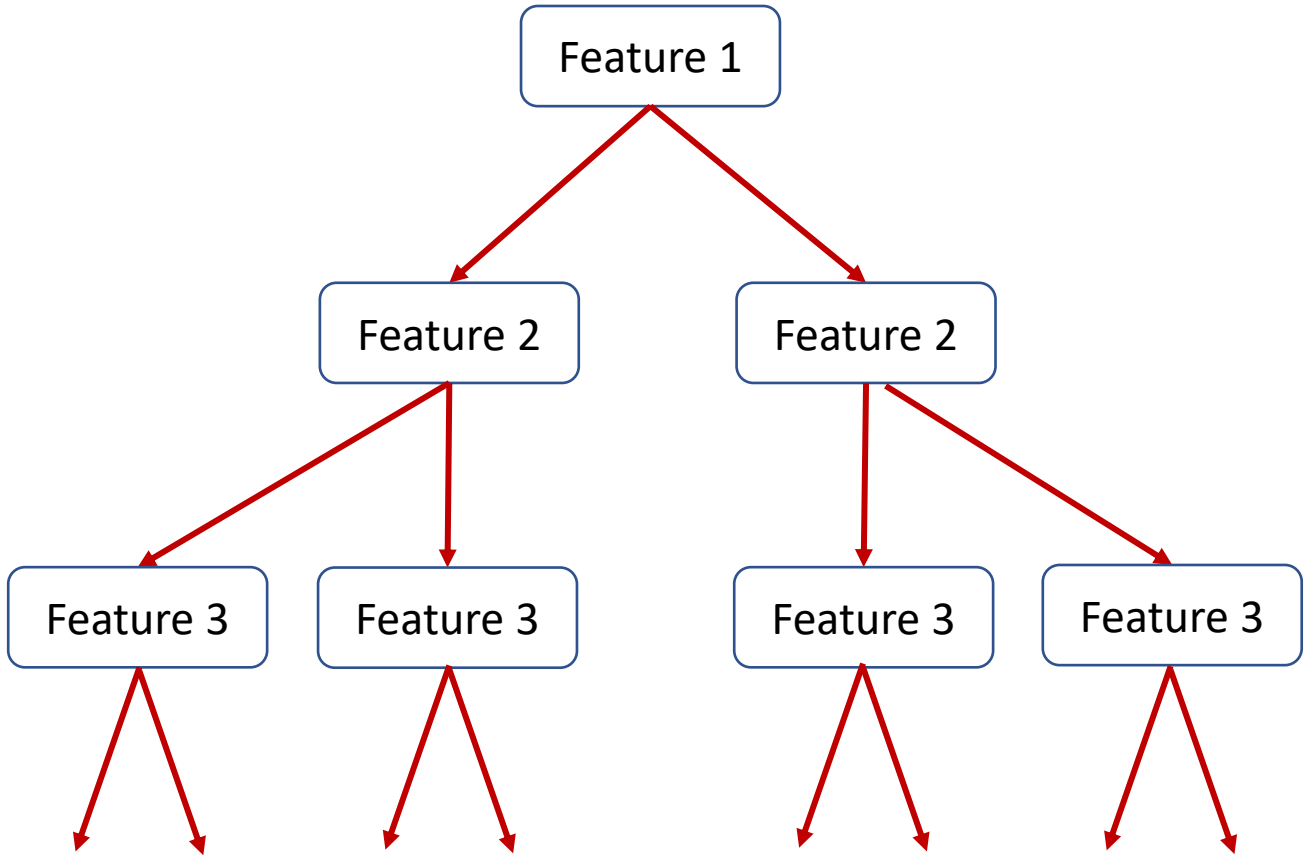
Greedy Algorithm

- Start from an empty decision tree
- Split on next best attribute
- Recurse

What is the Best Attribute for Split?

There are a number of metrics to inform how to split

A typical metric -- the information gain -- is inspired from information theory



Interlude:
Elements of Information Theory

Outline

- Entropy
- Conditional Entropy
- Joint Entropy
- Relative Entropy
- Mutual Information
- Jensen’s Inequality
- Data Processing Inequality
- Fano Inequality

Entropy

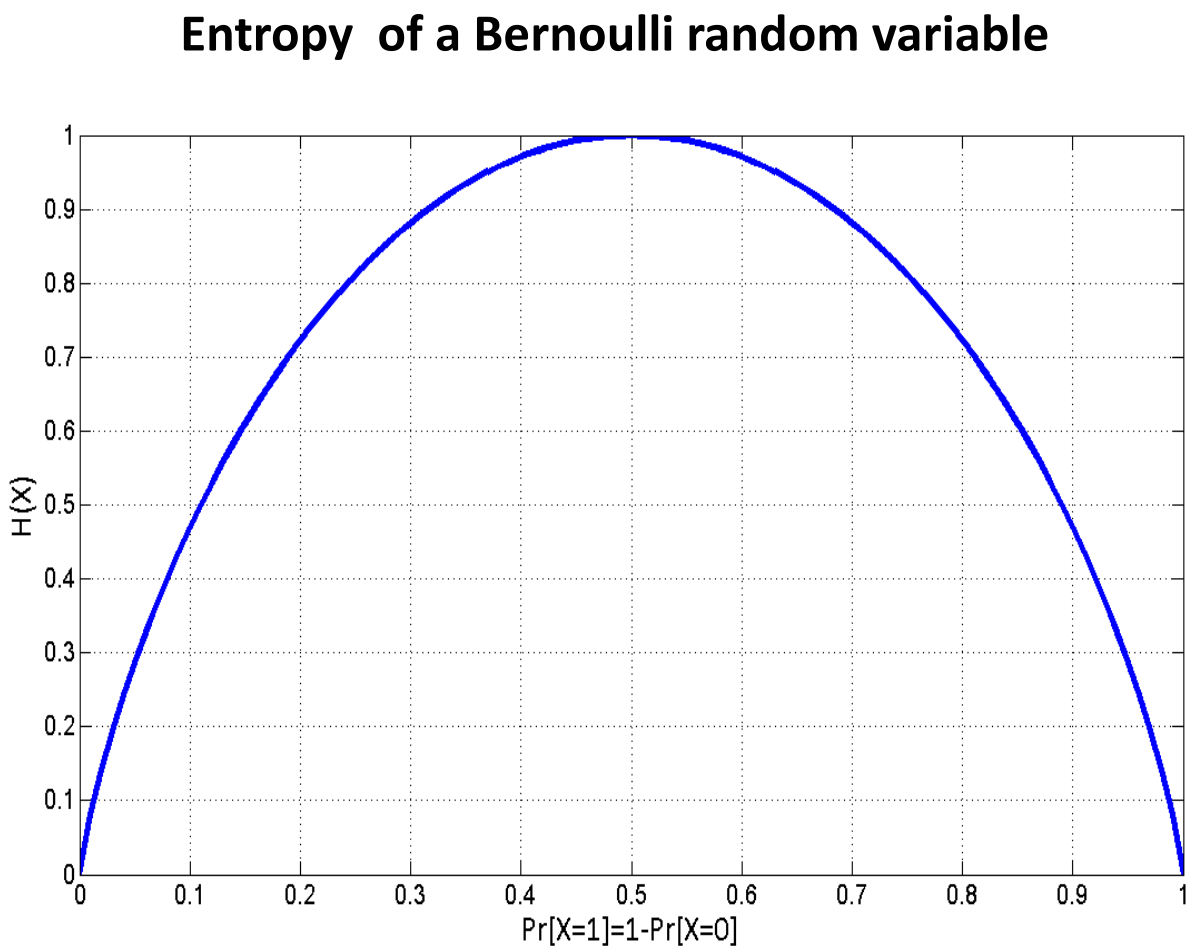
- The **entropy** H(X) of a (discrete) random variable X with probability mass function $p_X(x)$ is defined as:

$$H(X) = -\sum_{x \in S_X} p_X(x) \log p_X(x)$$

- The entropy typical unit is the bit, with the base of the logarithm equal to 2.

- The entropy arises as a natural answer to important questions in communications, namely, the ultimate data compression limit.

Entropy


Entropy of a Bernoulli random variable
H(X)
Pr[X=1]=1-Pr[X=0]
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Joint and Conditional Entropy

- Consider a pair of (discrete) random variables (X,Y) with joint probability mass function $p_{XY}(x,y)$.

- The **joint entropy** H(X,Y) of the pair of random variables (X,Y) is given by

$$H(X,Y) = -\sum_{x \in S_X} \sum_{y \in S_Y} p_{XY}(x,y) \log p_{XY}(x,y)$$

- The **conditional entropy** H(Y|X) of the random variable Y given the random variable X is given by

$$H(Y \mid X) = -\sum_{x \in S_X} \sum_{y \in S_Y} p_{XY}(x,y) \log p_{Y|X}(y \mid x)$$

Relative Entropy

- **Relative entropy**, or **Kullback-Leibler distance**, is a measure of the distance between two distributions.

- The relative entropy between two probability mass functions $p_X(x)$ and $q_X(x)$ is defined as:

$$D(p(x)||q(x)) = \sum_{x \in S_X} p(x) \log \frac{p(x)}{q(x)}$$

- The relative entropy also arises as the answer to relevant questions in communications.

Mutual Information

- Consider two (discrete) random variables X and Y with joint p.m.f. $p_{XY}(x,y)$ and marginal p.m.f. $p_X(x)$ and $p_Y(y)$.

- The **mutual information** I(X;Y) is the relative entropy between the joint distribution $p_{XY}(x,y)$ and the product distribution $p_X(x)\cdot p_Y(y)$:

$$I(X;Y) = D(p_{XY}(x,y)\,||\,p_X(x)p_Y(y)) = \sum_{x\in S_X}\sum_{y\in S_Y} p_{XY}(x,y)\log\frac{p_{XY}(x,y)}{p_X(x)p_Y(y)}$$

- Mutual information also arises as the natural answer to important questions in communications, namely, the ultimate date transmission limit.

Relations between Entropy and Mutual Information

Mutual Information and Entropy

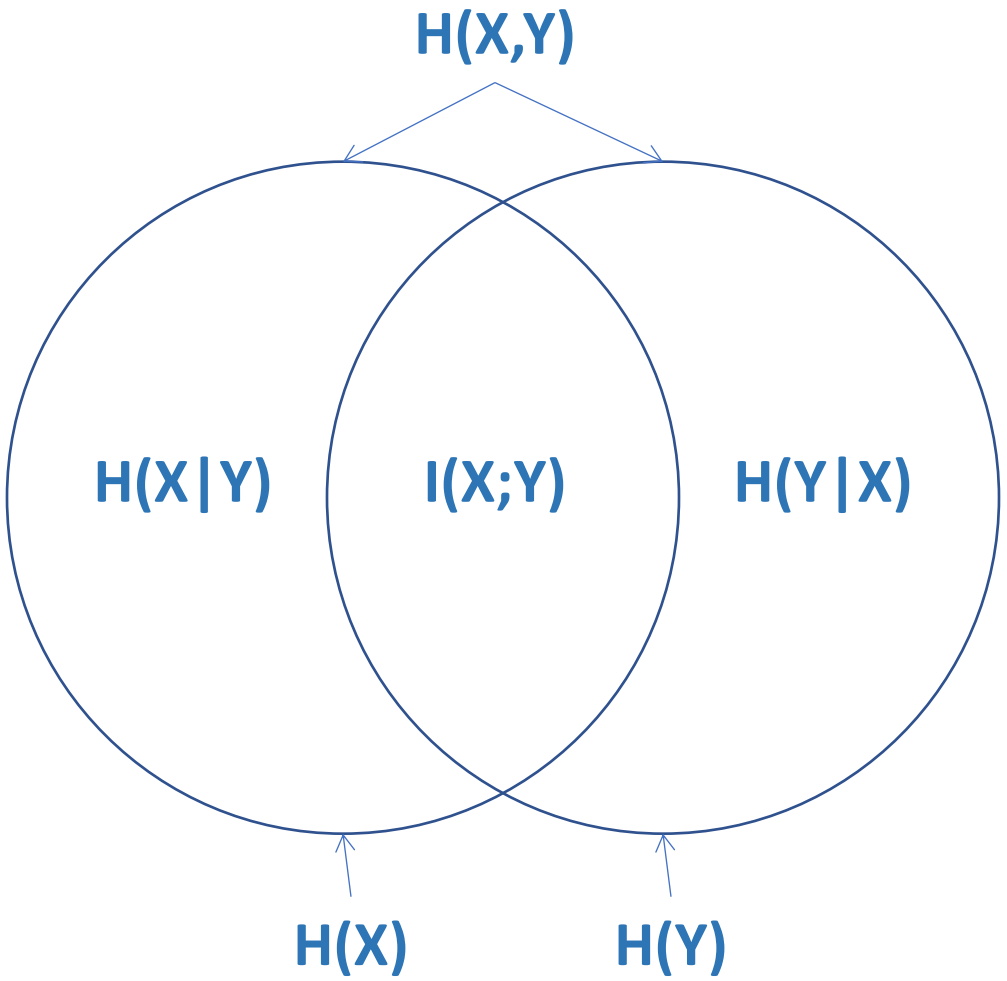
$$I(X;Y) = H(X) - H(X \mid Y)$$

$$I(X;Y) = H(Y) - H(Y \mid X)$$

$$I(X;Y) = H(X) + H(Y) - H(X,Y)$$

$$I(X;Y) = I(Y;X)$$

$$I(X;X) = H(X)$$



Properties

- Let X be a (discrete) random variable. The **entropy** satisfies $0 \leq H(X) \leq \log|S_X|$.

- Let X and Y be two (discrete) random variables. The **mutual information** satisfies $I(X;Y) \geq 0$, with equality if and only if X and Y are independent.

- Let $p_X(x)$ and $q_X(x)$, $x \in S_X$, be two probability mass functions. The **relative entropy** satisfies $D(p_X(x)||q_X(x)) \geq 0$, with equality if and only if $p_X(x) = q_X(x)$, for all $x \in S_X$.

Chain Rules

- Chain rule for entropy

$$H(X_1, X_2, \ldots, X_n) = \sum_{i=1}^{n} H(X_i \mid X_{i-1}, \ldots, X_1)$$

- Chain rule for mutual information

$$I(X_1, X_2, \ldots, X_n; Y) = \sum_{i=1}^{n} I(X_i; Y \mid X_{i-1}, \cdots, X_1)$$

- Chain rule for relative entropy

$$D(p(x,y)||q(x,y)) = D(p(x)||q(x)) + D(p(y \mid x)||q(y \mid x))$$

Jensen's Inequality

- A function f(x) is **convex** over an interval (a,b) if for every $x_1, x_2 \in (a,b)$ and $0 \leq \lambda \leq 1$, if

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$

- Or, a function f(x) is **convex** over an interval (a,b) if its second derivative is nonnegative everywhere.

- The **Jensen's inequality** states that if f(·) is a convex function then **$E[f(X)] \geq f(E[X])$**, where X is a random variable.

Data Processing Inequality

- Let the random variables X, Y, Z form a **Markov chain** in that order (i.e., $X \rightarrow Y \rightarrow Z$), so that X and Z are conditionally independent given Y.

- The **data processing inequality** states that if $X \rightarrow Y \rightarrow Z$ then $I(X;Y) \geq I(X;Z)$.

- In particular, let Z be a function of Y, $Z=g(y)$: So, $X \rightarrow Y \rightarrow Z=g(Y)$ and $I(X;Y) \geq I(X;Z=g(Y))$.

Fano's Inequality

- Let X and Y be correlated random variables with joint probability distribution $p_{XY}(x,y)$.

- The objective is to determine an estimate of the random variable X from the observation of the random variable Y, i.e., $X_{est}=g(Y)$

- **Fano's inequality** relates the probability of error $P_e=Pr\{X\neq X_{est}\}$ to the conditional entropy $H(X|Y)$, i.e., $H(P_e) + P_e\log(|S_X|-1)\geq H(X|Y)$.

End of Interlude

How to Construct Decision Trees?

How to choose the best attribute split?

The best attribute for partition is the one that is most helpful in discriminating among the labels.

The Information Gain (IG) provides a measure of the importance of different elements (attributes) of a feature vector.

The information gain is given by $IG(Y|X) = H(Y) - H(Y|X)$ where $H(\cdot)$ and $H(\cdot \mid \cdot)$ represent the entropy of a random quantity and entropy of a random quantity conditioned on another, respectively.

$H(Y) = -\sum p_i \log p_i$ Entropy

Example

X_1	X_2	X_3	Y
1	1	1	A
1	1	0	A
0	0	1	B
1	0	0	B

3 attributes, 1 label

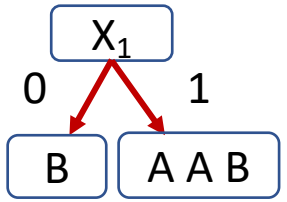
How to Construct Decision Trees?

Example

X_1	X_2	X_3	Y
1	1	1	A
1	1	0	A
0	0	1	B
1	0	0	B

3 attributes, 1 label

Split at X_1



$$H(Y) = {}^{1}/_{2}\log 2 + {}^{1}/_{2}\log 2 = 1$$

$$H(Y|X_1 = 0) = 1\log 1 + 0\log 0 = 0$$

$$H(Y|X_1 = 1) = {}^{1}/_{3}\log 3 + {}^{2}/_{3}\log {}^{3}/_{2} = 0.9814$$

$$IG = 1 - {}^{1}/_{4}.0 - {}^{3}/_{4}.0.9184 = \textit{0.312}$$

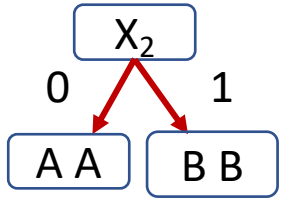
How to Construct Decision Trees?

Example

	X_1	X_2	X_3	Y
1	1	1	1	A
2	1	1	0	A
3	0	0	1	B
4	1	0	0	B

3 attributes, 1 label

Split at X_2



$$H(Y) = {}^{1}/_{2}\log 2 + {}^{1}/_{2}\log 2 = 1$$

$\{3, 4\}$ $H(Y|X_2 = 0) = 1\log 1 + 0\log 0 = 0$

$\{1, 2\}$ $H(Y|X_2 = 1) = 0\log 0 + 1\log 1 = 0$

$$IG = 1 - {}^{1}/_{2}.0 - {}^{1}/_{2}.0 = 1$$

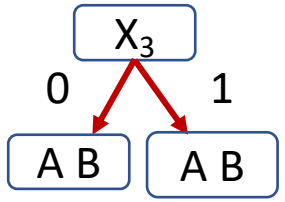
How to Construct Decision Trees?

Example

	X_1	X_2	X_3	Y
1	1	1	1	A
2	1	1	0	A
3	0	0	1	B
4	1	0	0	B

3 attributes, 1 label

Split at X_3



$$H(Y) = {}^1/_2 \log 2 + {}^1/_2 \log 2 = 1$$

{2,4} $$H(Y|X_3 = 0) = {}^1/_2 \log 2 + {}^1/_2 \log 2 = 1$$

{1,3} $$H(Y|X_3 = 1) = {}^1/_2 \log 2 + {}^1/_2 \log 2 = 1$$

$$IG = 1 - {}^1/_2 . 1 - {}^1/_2 . 1 = 0$$

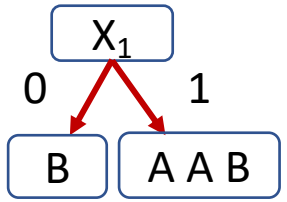
How to Construct Decision Trees?

Example

X_1	X_2	X_3	Y
1	1	1	A
1	1	0	A
0	0	1	B
1	0	0	B

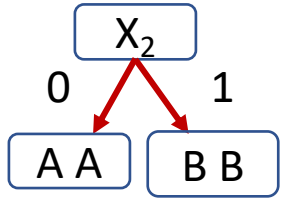
3 attributes, 1 label

Split at X_1



$IG = 0.312$

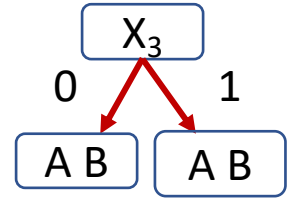
Split at X_2



$IG = 1$

Best Split!!

Split at X_3



$IG = 0$

Worst Split!!

Decision Trees: Other Generalizations

From discrete to continuous attributes

Use standard DT algorithm where the attributes are discretized (inefficient).

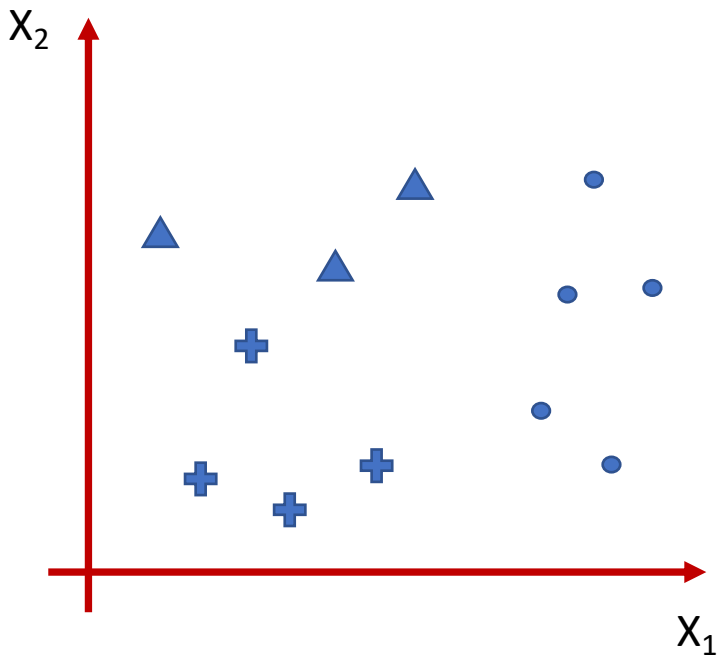
Or use a modified DT algorithm where each attribute is split with respect to thresholds.

Computation of IG to pick the best split makes the algorithm computationally more expensive for continuous variables than it is for discrete ones.

Many splits possible but usually only one split is made to circumvent complexity.

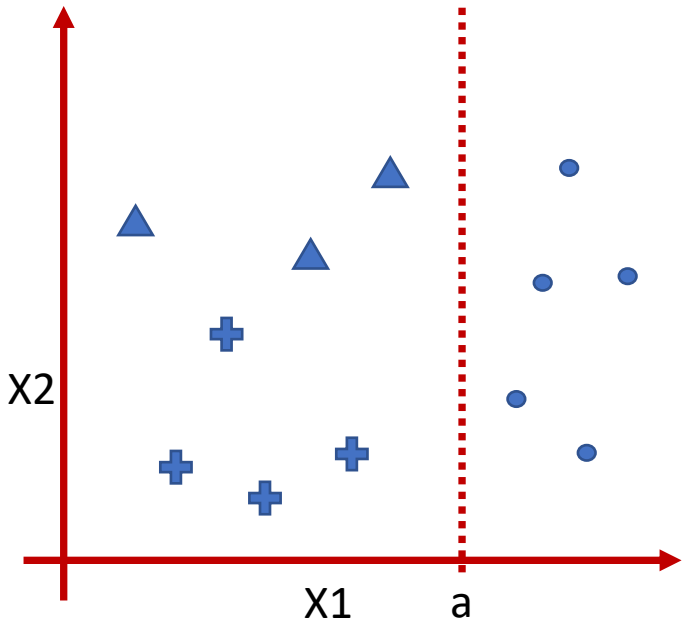
Example

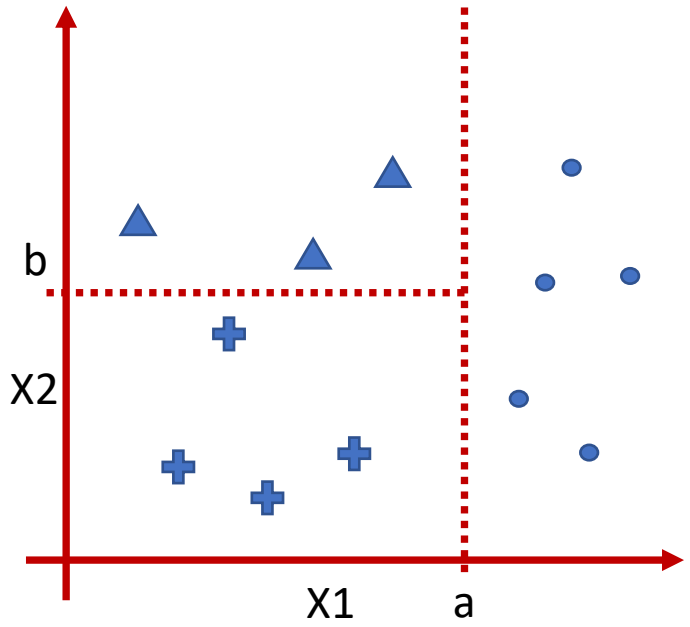
A training set with continuous attributes X_1 and X_2



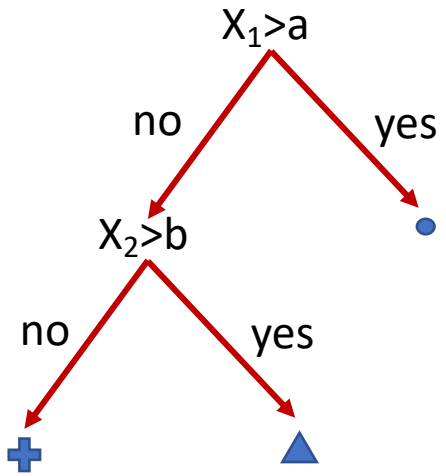
Decision Trees: Other Generalizations

Axis aligned thresholds





Decision Tree



Adapted from : http://www.uta.fi/sis/tie/ml/index.html

Decision Trees: Considerations

Considerations

Decision trees are a powerful learning algorithm that can represent any function of the attributes but may also require exponentially many nodes

More than one decision tree can be learnt consistent with a given training set

Smaller trees are associated with higher bias; larger trees are associated with higher variance. Hence, decision trees also tend to overfit.

There are different strategies that can be used to control the bias-variance tradeoff including (1) fixing depth/leaves (2) pruning

Decision Trees: Considerations

How to control overfitting?... Pruning

Prune a large tree from the leaves to the root where each node might be replaced with one of its subtrees or with a leaf, based on some bound or estimate

input:
original decision tree T .
function
output: pruned decision tree T
foreach node j in a bottom-up walk on T (from leaves to root):
  find T which minimizes $f(T, m)$, where T is any of the following:
    the current tree after replacing node j with a leaf 1.
    the current tree after replacing node j with a leaf 0.
    the current tree after replacing node j with its left subtree.
    the current tree after replacing node j with its right subtree.
    the current tree.
  let $T := T'$.

$f(T, m)$ is a bound/estimate for the generalization error of a decision tree T based on a sample of size m

Decision Trees: Strengths and Weaknesses

Strengths

- It is a very popular technique
- It is expressive
- It is fast
- It is useful when
  - Target Function is discrete
  - Concepts are likely to be disjunctions
  - Attributes may be noisy or missing

Weaknesses

- It is less useful for continuous outputs
- It is also more difficult to apply for continuous inputs
- It is also unstable in the sense that a small change in the data can lead to a large change in the structure of the decision tree.

Random Forests

Overview

Random forests is a fancier/refined version of bagging. In particular, in general performance is such that Random Forests>Decision Trees

It involves building a number of decision trees based on the bootstrapped training samples.

Each time a split in a tree is considered, a random sample of q attributes is chosen as split candidates from the full set of p attributes (feature bagging).

The optimum value of q is data dependent.

input: Decision Tree T_S ,training set S with p attributes , test data point to be labelled x.

For $b = 1{:}\,B$

Draw bootstrap samples $S_b^*$of size m with replacement from S

Select q attributes at random from the p attributes.

Pick the best variable/split---point among the q.

Split the node into two daughter nodes.

Repeat for each node until the minimum node size is reached

Output the ensemble of trees $\{T_{S_b^*}(x)\}_{b=1}^{B}$

For regression:

return $T_{bag}(x) = {}^{1}/_{B} \sum_{b=1}^{B} T_{S_b^*}(x)$

For classification:

return $T_{bag}(x) = Majority\ Vote\{T_{S_b^*}(x)\}_{b=1}^{B}$

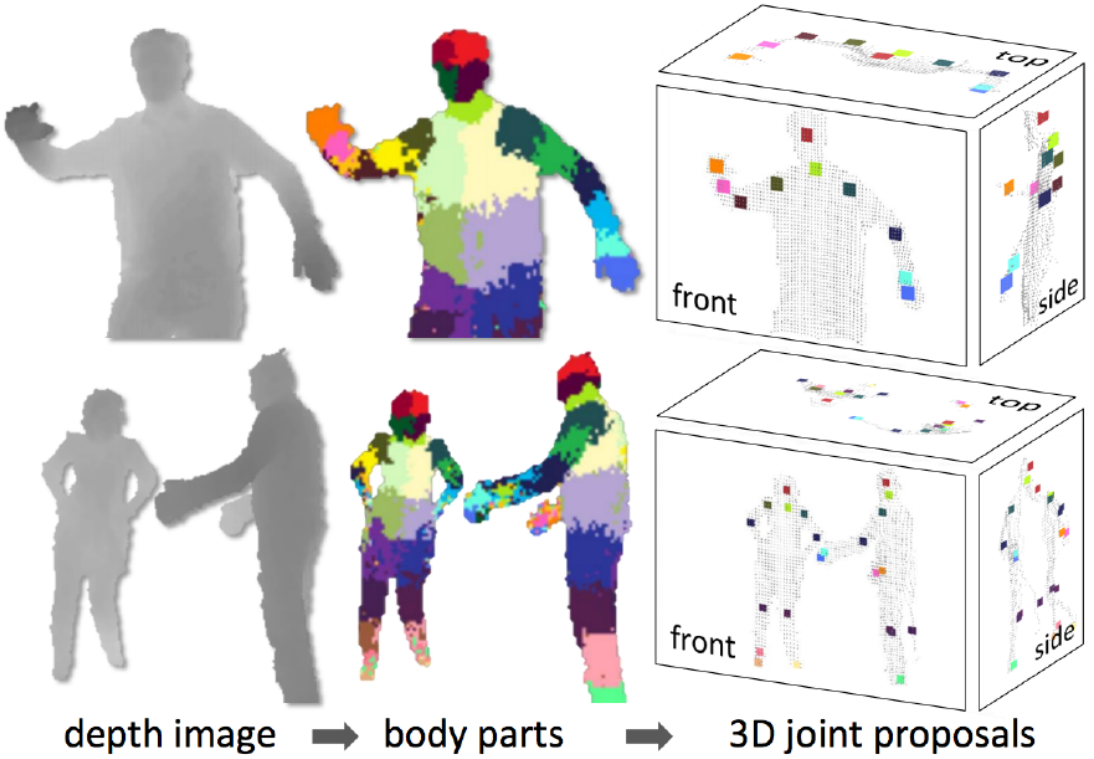
Decision Trees and Random Forests: Applications

Microsoft Kinect Pose Estimation

The Microsoft research team trained a decision forest, i.e. a collection of decision trees.

Millions of depth images were collected and prelabelled with target body parts. Each tree was then trained on a set of features from these images.

Training just three trees using 1 million test images took about a day using a 1000 core cluster.



Source: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/BodyPartRecognition.pdf

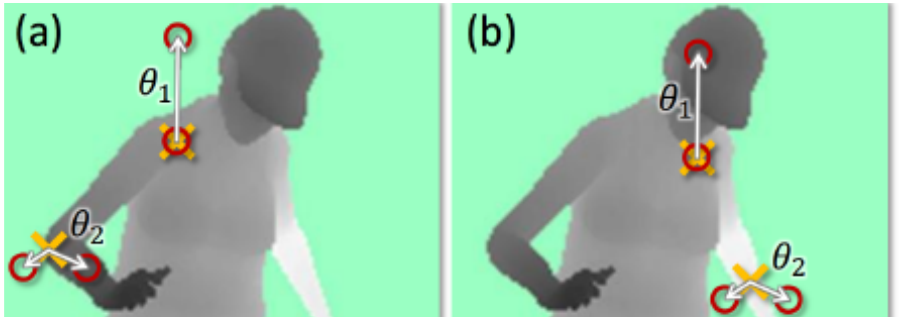
Decision Trees and Random Forests: Applications

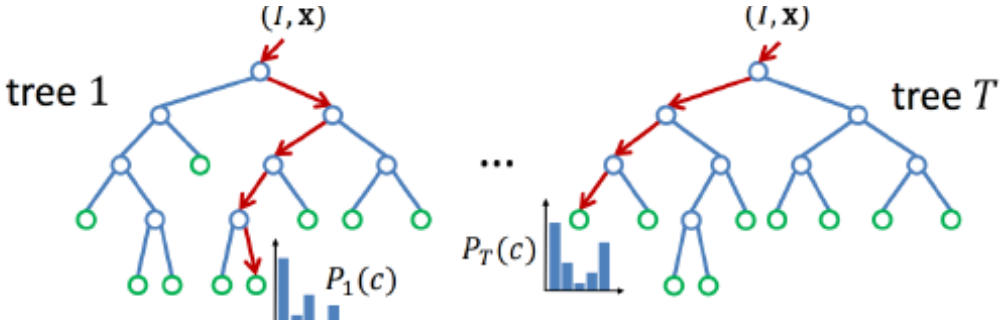
Microsoft Kinect Pose Estimation

The algorithm first assigns the probability of a pixel being in each body part.

The algorithm then picks out areas of maximum probability for each body part type.

That is, a certain area will be assigned to the category "leg" as long as the leg classifier has a maximum probability associated with that area.





Decision Trees and Random Forests: Applications

Microsoft Kinect Pose Estimation

The different body parts probability maxima are depicted using coloured areas.

The final stage is to compute suggested joint positions relative to the identified body parts.