10-701
Machine Learning

Decision trees

Types of classifiers

- We can divide the large variety of classification approaches into roughly two main types

1. Instance based classifiers
   - Use observation directly (no models)
   - e.g. K nearest neighbors

2. Generative:
   - build a generative statistical model
   - e.g., Bayesian networks

3. Discriminative
   - directly estimate a decision rule/boundary
   - e.g., decision tree

Decision trees

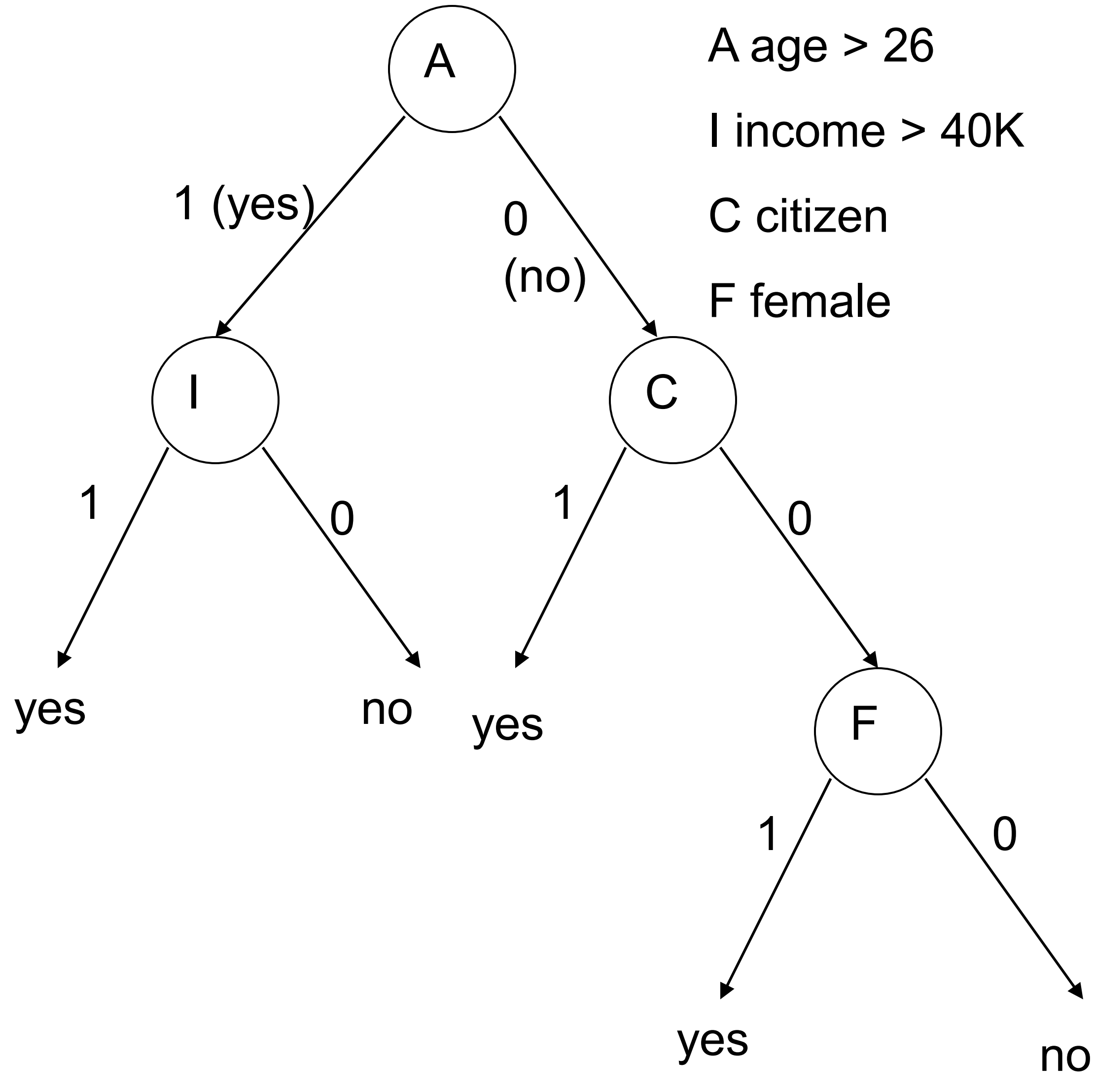
- One of the most intuitive classifiers
- Easy to understand and construct
- Surprisingly, also works very (very) well*

Lets build a decision tree!

* More on this in future lectures

Structure of a decision tree

- Internal nodes correspond to attributes (features)
- Leafs correspond to classification outcome
- edges denote assignment



Building a decision tree

```
Function BuildTree(n,A)    // n: samples (rows), A: attributes
  If empty(A) or all n(L) are the same
     status = leaf
     class = most common class in n(L)
  else
     status = internal
     a ⇐ bestAttribute(n,A)
     LeftNode = BuildTree(n(a=1), A \ {a})
     RightNode = BuildTree(n(a=0), A \ {a})
  end
end
```

n(L): Labels for samples in this set

We will discuss this function next

Recursive calls to create left and right subtrees, n(a=1) is the set of samples in n for which the attribute a is 1

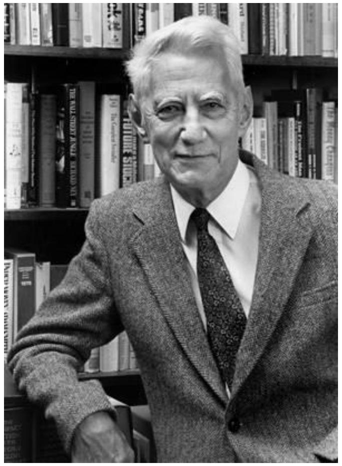
Identifying ‘bestAttribute’

- There are many possible ways to select the best attribute for a given set.
- We will discuss one possible way which is based on information theory and generalizes well to non binary variables

Entropy

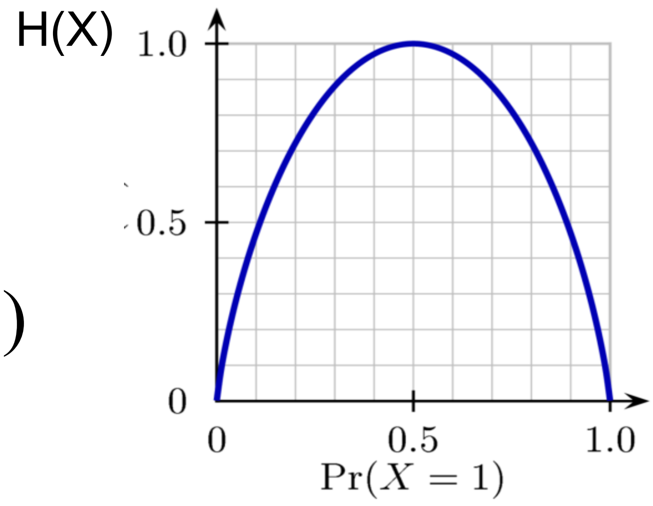
- Quantifies the amount of uncertainty associated with a specific probability distribution
- The higher the entropy, the less confident we are in the outcome
- Definition

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



Claude Shannon (1916 – 2001), most of the work was done in Bell labs

Entropy



- Definition

$$H(X) = \sum_i - p(X = i) \log_2 p(X = i)$$

- So, if P(X=1) = 1 then

$$H(X) = -p(x=1)\log_2 p(X=1) - p(x=0)\log_2 p(X=0)$$
$$= -1\log 1 - 0\log 0 = 0$$

- If P(X=1) = .5 then

$$H(X) = -p(x=1)\log_2 p(X=1) - p(x=0)\log_2 p(X=0)$$
$$= -.5\log_2 .5 - .5\log_2 .5 = -\log_2 .5 = 1$$

Interpreting entropy

- Entropy can be interpreted from an information standpoint
- Assume both sender and receiver know the distribution. How many bits, on average, would it take to transmit one value?
- If $P(X=1) = 1$ then the answer is 0 (we don’t need to transmit anything)
- If $P(X=1) = .5$ then the answer is 1 (either values is equally likely)
- If $0<P(X=1)<.5$ or $0.5<P(X=1)<1$ then the answer is between 0 and 1

  - Why?

Conditional entropy

Movie length	Liked?
Short	Yes
Short	No
Medium	Yes
long	No
Long	No
Medium	Yes
Short	Yes
Long	Yes
Medium	Yes

- We can generalize the conditional entropy idea to determine H(Li | Le)
- That is, what is the expected number of bits we need to transmit if both sides know the value of Le for each of the records (samples)
- Definition: $$H(Y \mid X) = \sum_i P(X = i) H(Y \mid X = i)$$

We explained how to compute this in the previous slides

Conditional entropy: Example

$$H(Y \mid X) = \sum_i P(X = i) H(Y \mid X = i)$$

Movie length	Liked?
Short	Yes
Short	No
Medium	Yes
long	No
Long	No
Medium	Yes
Short	Yes
Long	Yes
Medium	Yes

- Lets compute H(Li | Le)

H(Li | Le) = P(Le = S) H(Li | Le=S)+

P(Le = M) H(Li | Le=M)+

P(Le = L) H(Li | Le=L) =

1/3*.92+1/3*0+1/3*.92 =

0.61

we already computed:

H(Li | Le = S) = .92

H(Li | Le = M) = 0

H(Li | Le = L) = .92

Information gain

- How much do we gain (in terms of reduction in entropy) from knowing one of the attributes
- In other words, what is the reduction in entropy from this knowledge
- Definition: $IG(Y|X)^* = H(Y)-H(Y|X)$

*$IG(X|Y)$ is always ≥ 0

Proof: Jensen inequality

Building a decision tree

```
Function BuildTree(n,A)    // n: samples (rows), A: attributes
  If empty(A) or all n(L) are the same
     status = leaf
     class = most common class in n(L)
  else
     status = internal
     a ⇐ bestAttribute(n,A)
     LeftNode = BuildTree(n(a=1), A \ {a})
     RightNode = BuildTree(n(a=0), A \ {a})
   end
end
```

Based on information gain

Example: Root attribute

P(Li=yes) = 2/3

H(Li) = .91

H(Li | T) =

H(Li | Le) =

H(Li | D) =

H(Li | F) =

Movie	Type	Length	Director	Famous actors	Liked ?
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	animated	long	Lasseter	Yes	No
m5	Comedy	Long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
M7	animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Example: Root attribute

$P(Li=yes) = 2/3$

$H(Li) = .91$

$H(Li \mid T) = 0.61$

$H(Li \mid Le) = 0.61$

$H(Li \mid D) = 0.36$

$H(Li \mid F) = 0.85$

Movie	Type	Length	Director	Famous actors	Liked ?
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	animated	long	Lasseter	Yes	No
m5	Comedy	Long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
M7	animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Example: Root attribute

P(Li=yes) = 2/3

H(Li) = .91

H(Li | T) = 0.61

H(Li | Le) = 0.61

H(Li | D) = 0.36

H(Li | F) = 0.85

IG(Li | T) = .91-.61 = 0.3

IG(Li | Le) = .91-.61 = 0.3

IG(Li | D) = .91-.36 = 0.55

IG(Li | Le) = .91-.85 = 0.06

Movie	Type	Length	Director	Famous actors	Liked ?
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	animated	long	Lasseter	Yes	No
m5	Comedy	Long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
M7	animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Example: Root attribute

P(Li=yes) = 2/3

H(Li) = .91

H(Li | T) = 0.61

H(Li | Le) = 0.61

H(Li | D) = 0.36

H(Li | F) = 0.85

IG(Li | T) = .91-.61 = 0.3

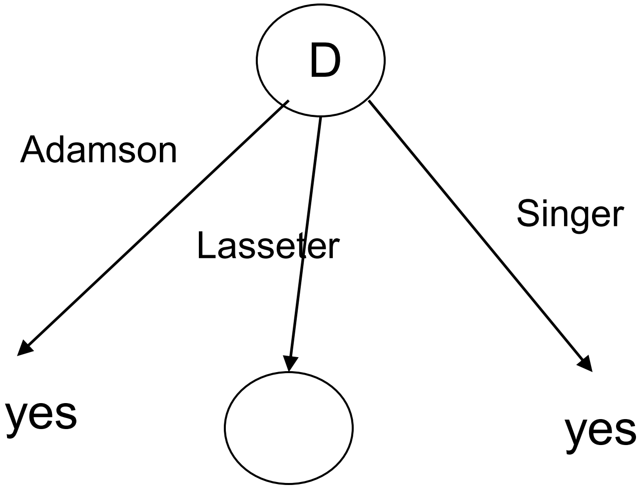
IG(Li | Le) = .91-.61 = 0.3

$\boxed{\text{IG(Li | D) = .91-.36 = 0.55}}$

IG(Li | Le) = .91-.85 = 0.06

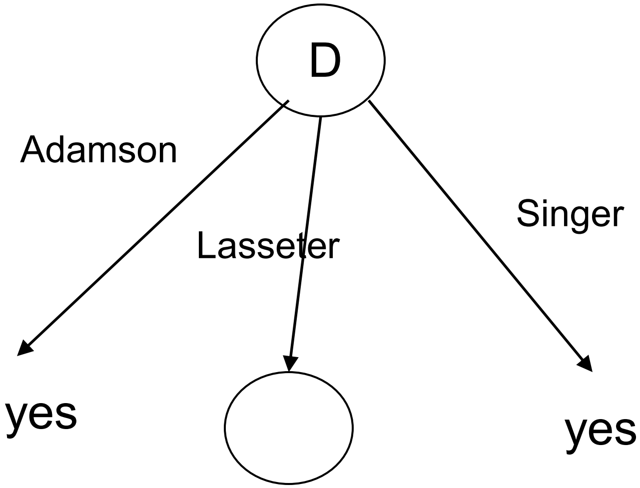
Movie	Type	Length	Director	Famous actors	Liked ?
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	animated	long	Lasseter	Yes	No
m5	Comedy	Long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
M7	animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Building a tree



Movie	Type	Length	Director	Famous actors	Liked ?
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	animated	long	Lasseter	Yes	No
m5	Comedy	Long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
M7	animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Building a tree

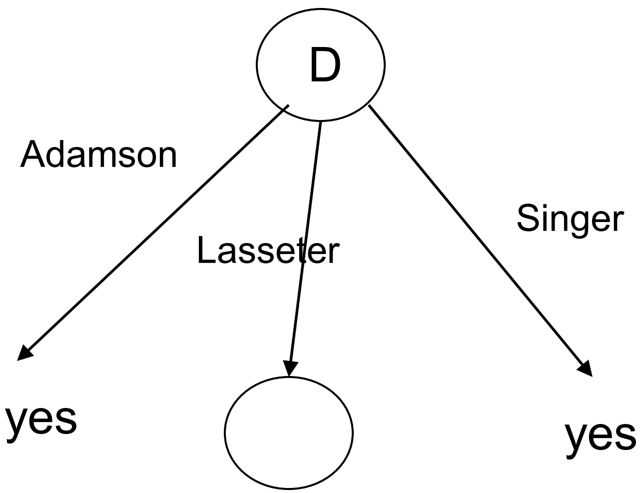


Movie	Type	Length	Director	Famous actors	Liked ?
m2	Animated	Short	Lasseter	No	No
m4	animated	Long	Lasseter	Yes	No
m5	Comedy	Long	Lasseter	Yes	No
m9	Drama	Medium	Lasseter	No	Yes

We only need to focus on the records (samples) associated with this node

Building a tree

We eliminated the 'director' attribute. All samples have the same director



Movie	Type	Length	Famous actors	Liked ?
m2	Animated	Short	No	No
m4	animated	Long	Yes	No
m5	Comedy	Long	Yes	No
m9	Drama	Medium	No	Yes

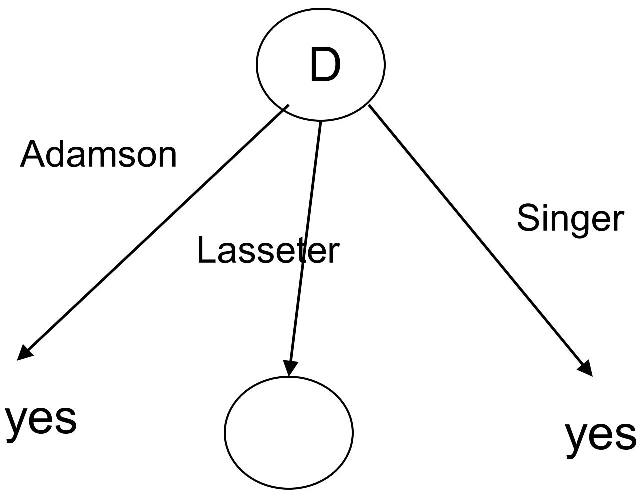
P(Li=yes) = 1/4   H(Li) = .81

H(Li | T) = 0

H(Li | Le) = 0

H(Li | F) = 0.5

Building a tree



Movie	Type	Length	Famous actors	Liked ?
m2	Animated	Short	No	No
m4	animated	long	Yes	No
m5	Comedy	Long	Yes	No
m9	Drama	Medium	No	Yes

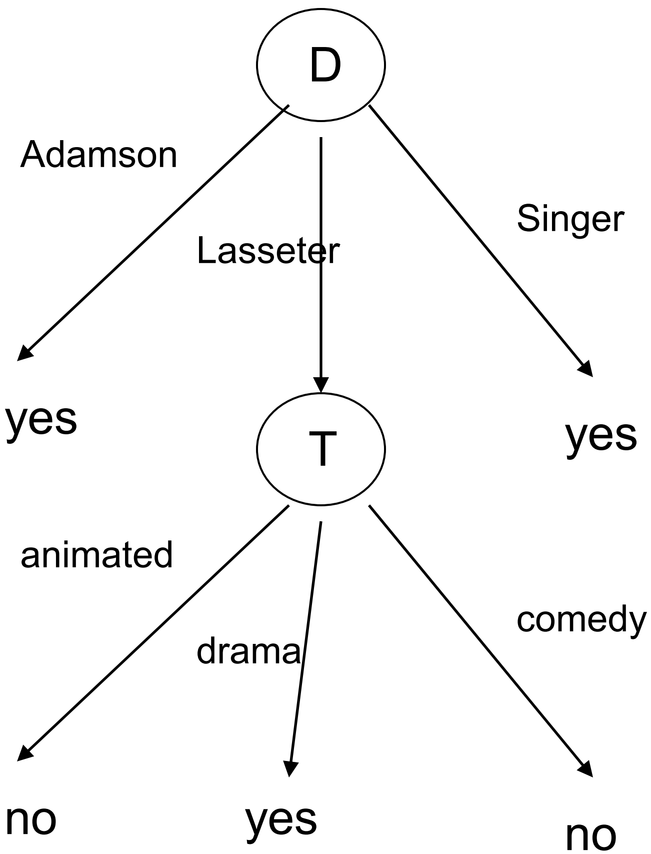
P(Li=yes) = 1/4   H(Li) = .81

H(Li | T) = 0 $\boxed{\text{IG(Li | T) = 0.81}}$

H(Li | Le) = 0   IG(Li | Le) = 0.81

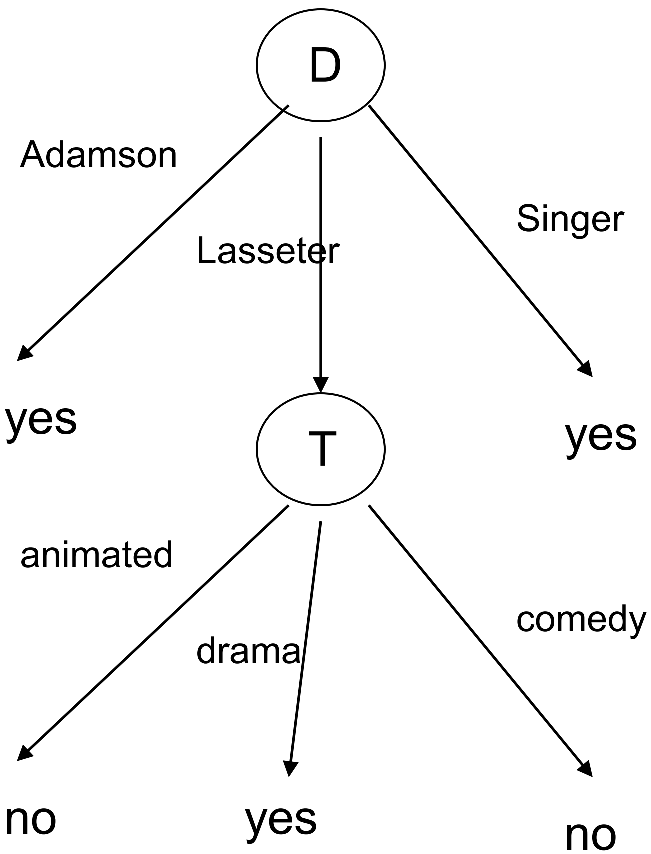
H(Li | F) = 0.5   IG(Li | F) = .31

Building a tree



Movie	Type	Length	Famous actors	Liked ?
m2	Animated	Short	No	No
m4	animated	long	Yes	No
m5	Comedy	Long	Yes	No
m9	Drama	Medium	No	Yes

Final tree



Movie	Type	Length	Director	Famous actors	Liked ?
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	animated	long	Lasseter	Yes	No
m5	Comedy	Long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
M7	animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes
m9	Drama	Medium	Lasseter	No	Yes

Additional points

- The algorithm we gave reaches homogonous nodes (or runs out of attributes)
- This is dangerous: For datasets with many (non relevant) attributes the algorithm will continue to split nodes
- This will lead to overfitting!

Avoiding overfitting: Tree pruning

- Split data into train and test set
- Build tree using training set

- For all internal nodes (starting at the root)
  - remove sub tree rooted at node
  - assign class to be the most common among training set
  - check test data error
    - if error is lower, keep change
    - otherwise restore subtree, repeat for all nodes in subtree

Continuous values

- Either use threshold to turn into binary or discretize
- Its possible to compute information gain for all possible tresholds (there are a finite number of training samples)
- Harder if we wish to assign more than two values (can be done recursively)

The ‘best’ classifier

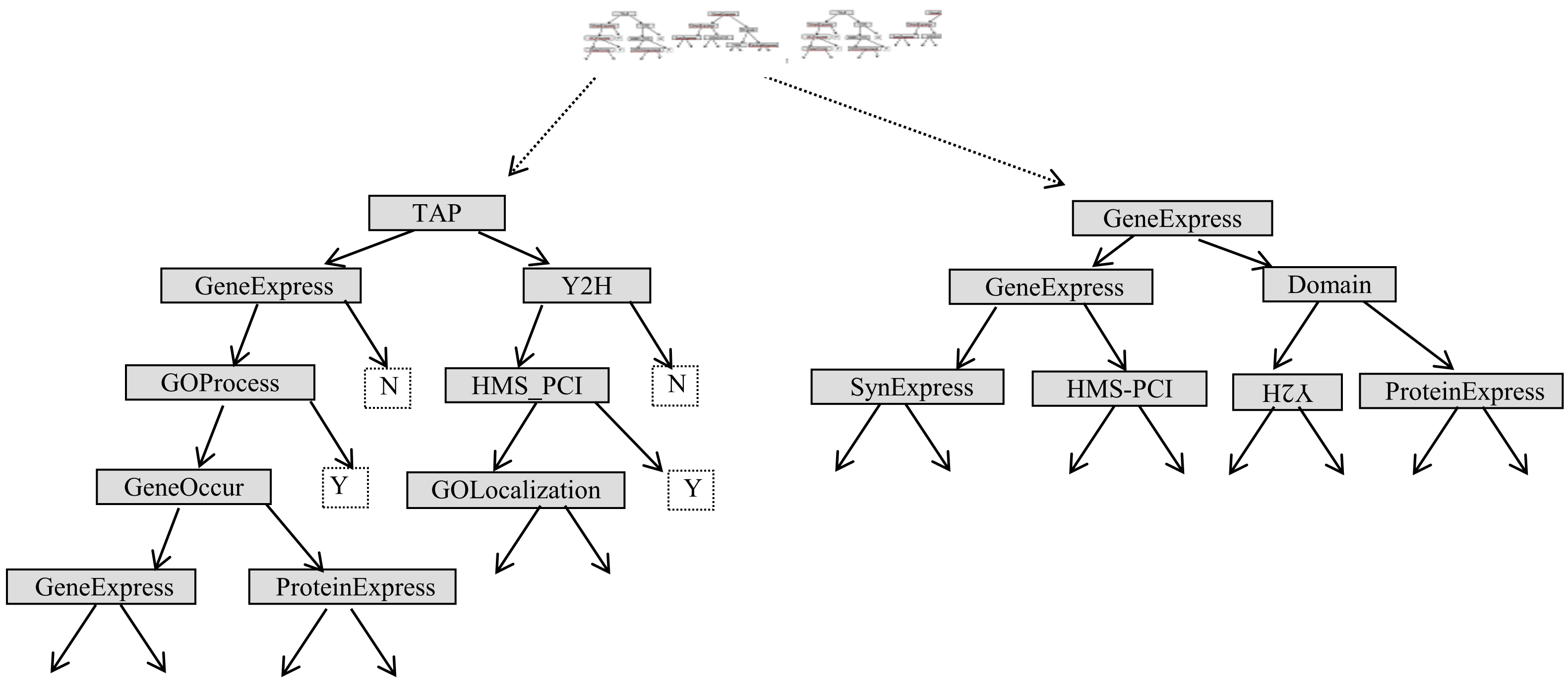
- There has been a lot of interest lately in decision trees.
- They are quite robust, intuitive and, surprisingly, very accurate

Important points

- Discriminative classifiers
- Entropy
- Information gain
- Building decision trees

Random forest

- A collection of decision trees
- For each tree we select a subset of the attributes (recommended square root of |A|) and build tree using just these attributes
- An input sample is classified using majority voting



Decision trees and Naïve Bayes

- What are the relationships between the assumptions the two classifiers make?
- How does this affect their ability to model different input datasets?
  - Number of feature?
  - Number of samples?
- How does this affect the way they handle the different features?