10-701
Machine Learning

Decision trees

Optional additional reading: Mitchell Chapter 3

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

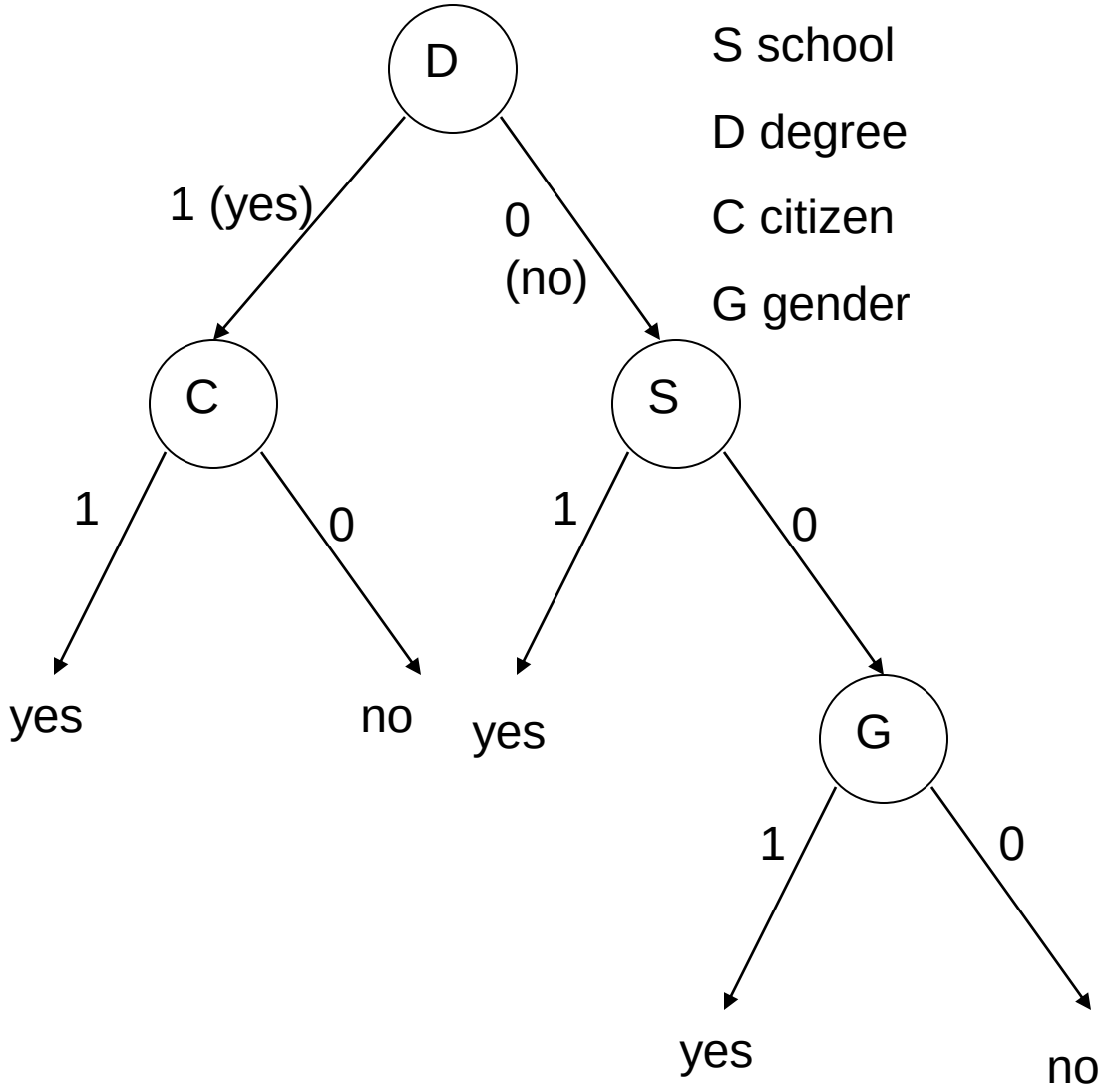
- One of the most intuitive classifiers
- Easy to understand and construct
- Surprisingly, also works very (very) well*

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

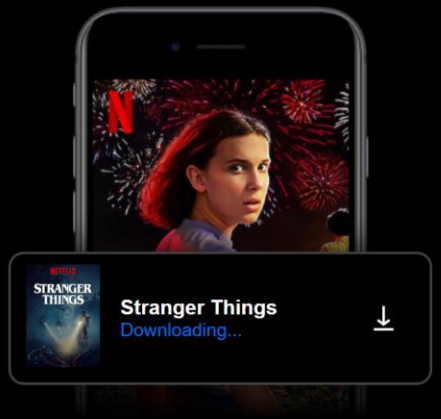


Netflix

Enjoy on your TV.

Watch on Smart TVs, Playstation, Xbox, Chromecast, Apple TV, Blu-ray players, and more.

TRY 30 DAYS FREE ›



Download your shows to watch on the go.

Save your data and watch all your favorites offline.

TRY 30 DAYS FREE ›

Dataset

Attributes (features)

Label

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 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
```

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

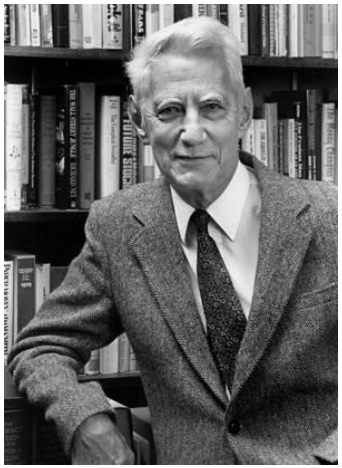
Identifying ‘bestAttribute’

- 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

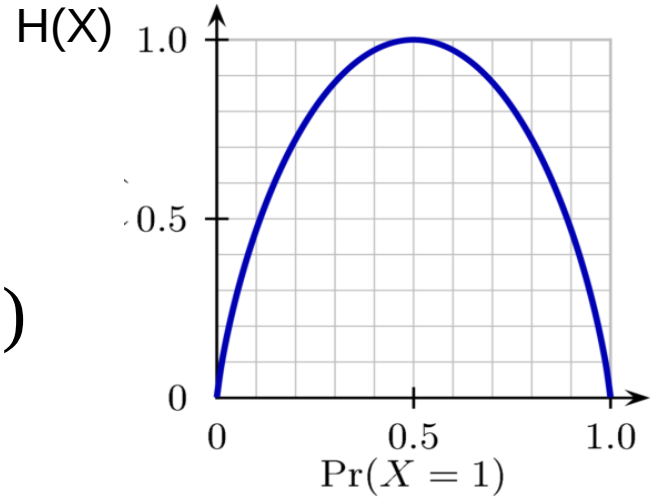
- Quantifies the amount of uncertainty associated with a specific probability distribution
- The higher the entropy, the less confident we are in the outcome
- Definition

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



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?

Expected bits per symbol

- Assume P(X=1) = 0.8
- Then P(11) = 0.64, P(10)=P(01)=.16 and P(00)=.04
- Lets define the following code
  - For 11 we send 0
  - For 10 we send 10
  - For 01 we send 110
  - For 00 we send 1110

Expected bits per symbol

- Assume P(X=1) = 0.8
- Then P(11) = 0.64, P(10)=P(01)=.16 and P(00)=.04
- Lets define the following code
  - For 11 we send 0
  - For 10 we send 10
  - For 01 we send 110
  - For 00 we send 1110

so: 01001101110001101110

can be broken to: 01 00 11 01 11 00 01 10 11 10

which is: 110 1110 0 110 0 1110 110 10 0 10

- What is the expected bits / symbol?
  (.64*1+.16*2+.16*3+.04*4)/2 = 0.8
- Entropy (lower bound) H(X)=0.7219

Conditional entropy

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

- Entropy measures the uncertainty in a specific distribution
- What if both sender and receiver know something about the transmission?
- For example, say I want to send the label (liked) when the length is known
- This becomes a conditional entropy problem: H(Li | Le=v)

Is the entropy of Liked among movies with length v

Conditional entropy: Examples for specific values

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=v)

1. H(Li | Le = S) = .92

Conditional entropy: Examples for specific values

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=v)

1. H(Li | Le = S) = .92
2. H(Li | Le = M) = 0
3. H(Li | Le = L) = .92

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

Where we are

- We were looking for a good criteria for selecting the best attribute for a node split
- We defined the entropy, conditional entropy and information gain
- We will now use information gain as our criteria for a good split
- That is, BestAttribute will return the attribute that maximizes the information gain at each node

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 | T) = 0.61

H(Li | Le) = 0.61

H(Li | D) = 0.36

H(Li | 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

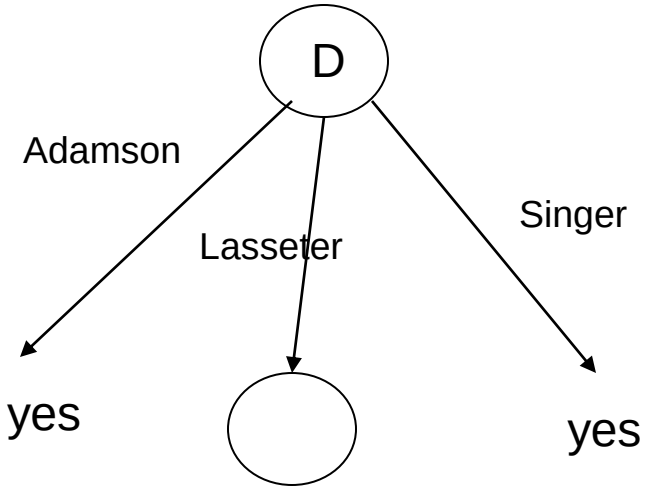
IG(Li | Le) = .91-.61 = 0.3

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

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

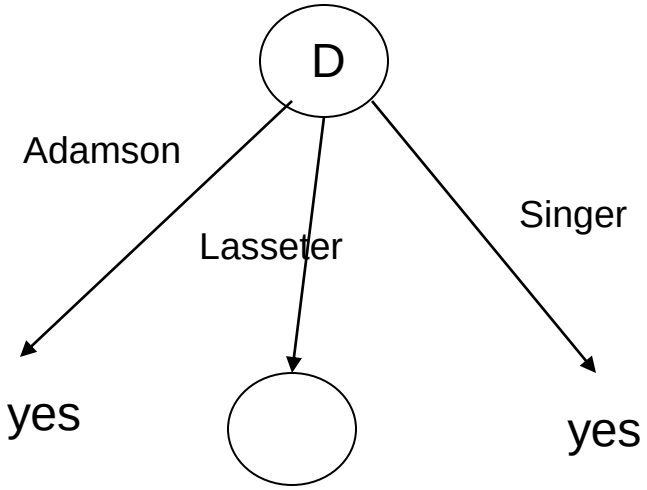
Movie	Type	Length	Director	Famous actors	Liked ?
m1	Comedy	Short	Adamson	No	Yes
m2	Animated	Short	Lasseter	No	No
m3	Drama	Medium	Adamson	No	Yes
m4	animated	long	Lasseter	Yes	No
m5	Comedy	Long	Lasseter	Yes	No
m6	Drama	Medium	Singer	Yes	Yes
M7	animated	Short	Singer	No	Yes
m8	Comedy	Long	Adamson	Yes	Yes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

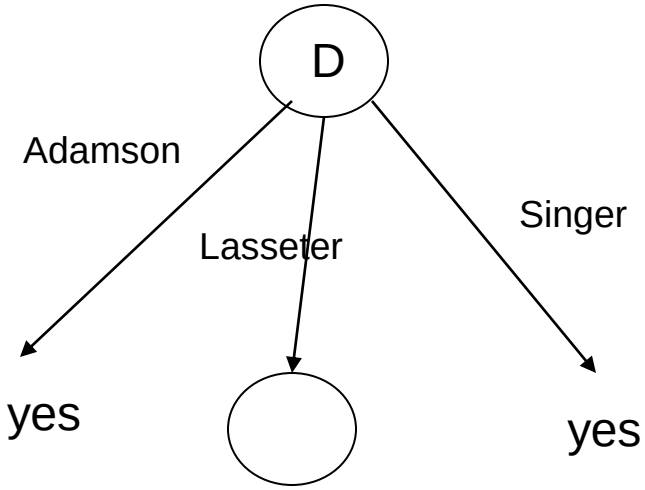


Movie	Type	Length	Director	Famous actors	Liked ?
m2	Animated	Short	Lasseter	No	No
m4	animated	Long	Lasseter	Yes	No
m5	Comedy	Long	Lasseter	Yes	No
m9	Drama	Medium	Lasseter	No	Yes

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

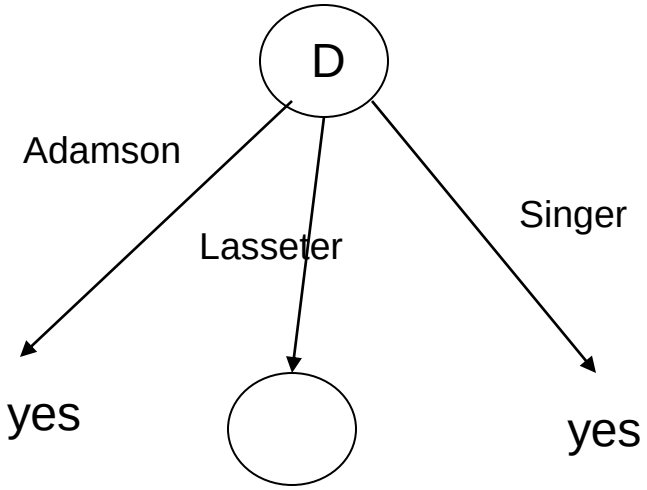
P(Li=yes) = 1/4 H(Li) = .81

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

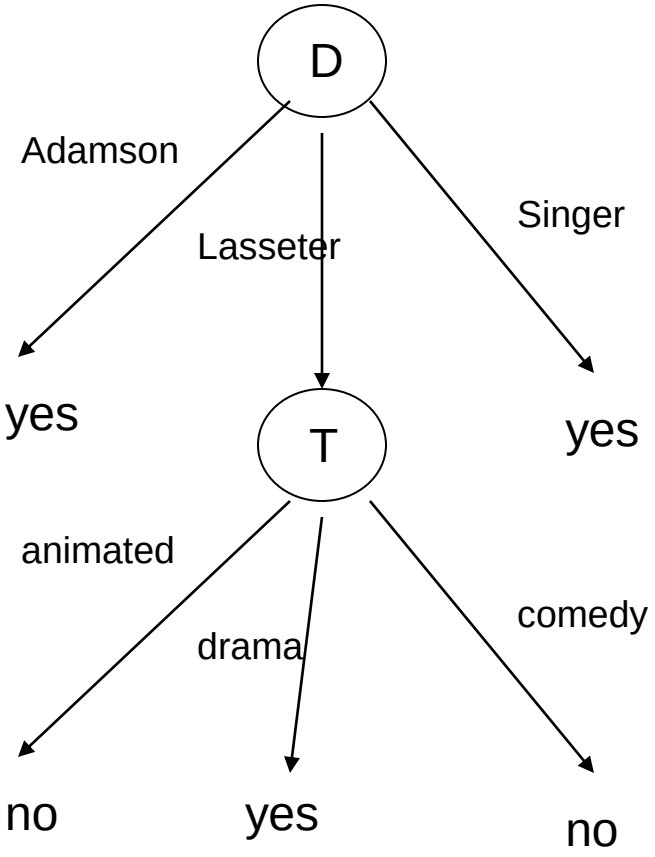
P(Li=yes) = 1/4 H(Li) = .81

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

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

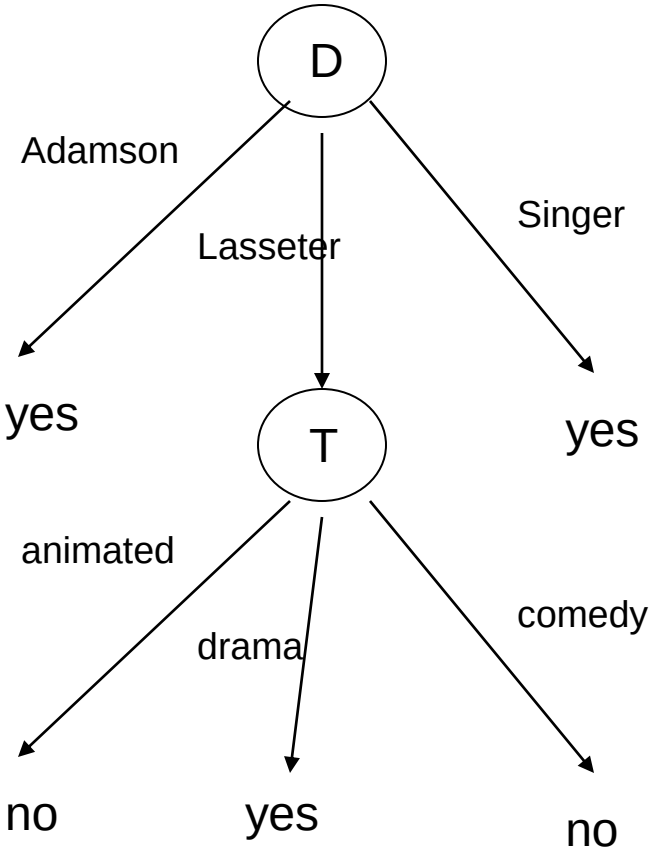
H(Li | F) = 0.5 IG(Li | F) = .31

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

Ranking classifiers

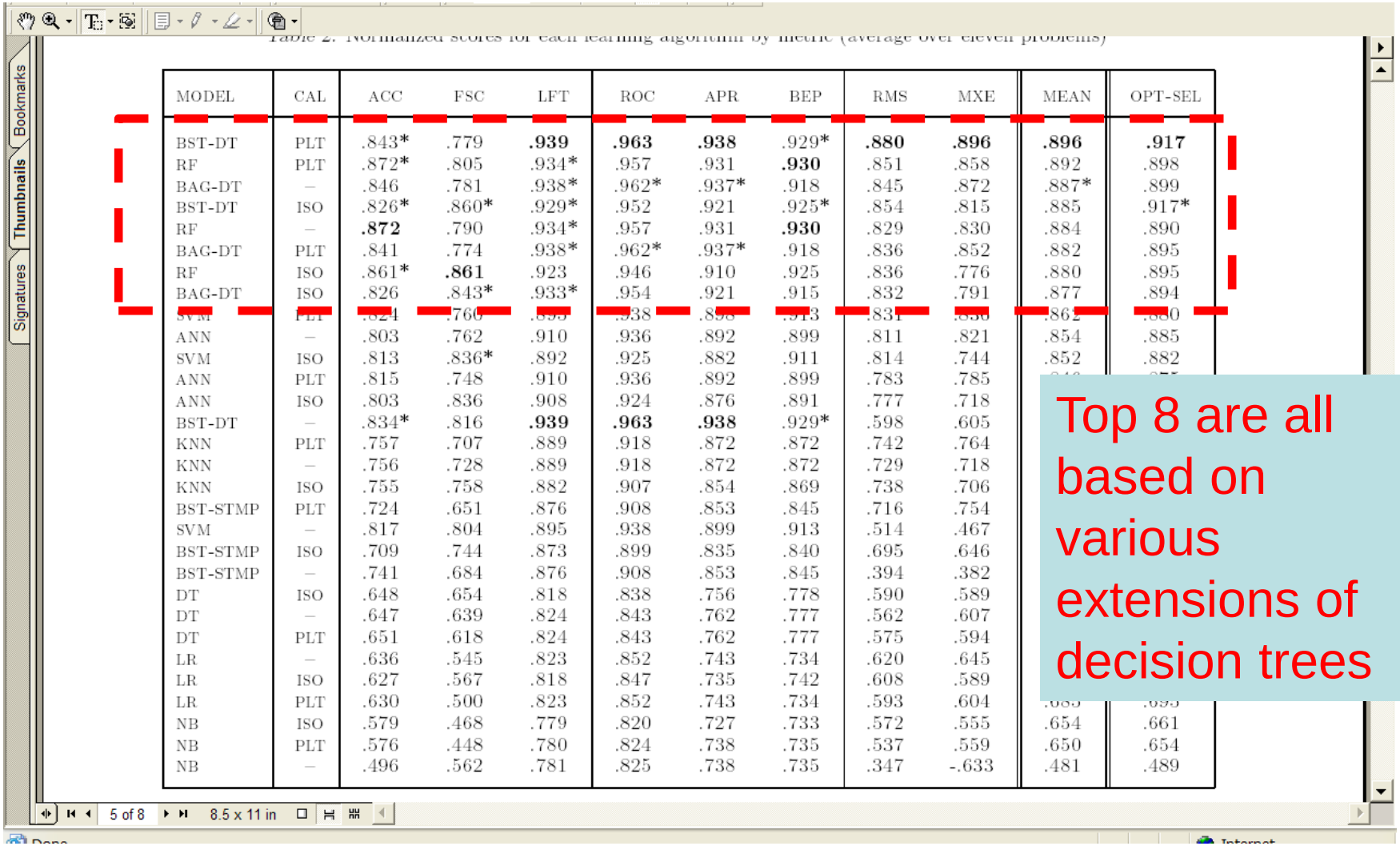


Table 2. Normalized scores for each learning algorithm by metric (average over eleven problems)

MODEL	CAL	ACC	FSC	LFT	ROC	APR	BEP	RMS	MXE	MEAN	OPT-SEL
BST-DT	PLT	.843*	.779	**.939**	**.963**	**.938**	.929*	**.880**	**.896**	**.896**	**.917**
RF	PLT	.872*	.805	.934*	.957	.931	**.930**	.851	.858	.892	.898
BAG-DT	–	.846	.781	.938*	.962*	.937*	.918	.845	.872	.887*	.899
BST-DT	ISO	.826*	.860*	.929*	.952	.921	.925*	.854	.815	.885	.917*
RF	–	**.872**	.790	.934*	.957	.931	**.930**	.829	.830	.884	.890
BAG-DT	PLT	.841	.774	.938*	.962*	.937*	.918	.836	.852	.882	.895
RF	ISO	.861*	**.861**	.923	.946	.910	.925	.836	.776	.880	.895
BAG-DT	ISO	.826	.843*	.933*	.954	.921	.915	.832	.791	.877	.894
SVM	PLT	.824	.760	.895	.938	.898	.913	.831	.836	.862	.880
ANN	–	.803	.762	.910	.936	.892	.899	.811	.821	.854	.885
SVM	ISO	.813	.836*	.892	.925	.882	.911	.814	.744	.852	.882
ANN	PLT	.815	.748	.910	.936	.892	.899	.783	.785		
ANN	ISO	.803	.836	.908	.924	.876	.891	.777	.718		
BST-DT	–	.834*	.816	**.939**	**.963**	**.938**	.929*	.598	.605		
KNN	PLT	.757	.707	.889	.918	.872	.872	.742	.764		
KNN	–	.756	.728	.889	.918	.872	.872	.729	.718		
KNN	ISO	.755	.758	.882	.907	.854	.869	.738	.706		
BST-STMP	PLT	.724	.651	.876	.908	.853	.845	.716	.754		
SVM	–	.817	.804	.895	.938	.899	.913	.514	.467		
BST-STMP	ISO	.709	.744	.873	.899	.835	.840	.695	.646		
BST-STMP	–	.741	.684	.876	.908	.853	.845	.394	.382		
DT	ISO	.648	.654	.818	.838	.756	.778	.590	.589		
DT	–	.647	.639	.824	.843	.762	.777	.562	.607		
DT	PLT	.651	.618	.824	.843	.762	.777	.575	.594		
LR	–	.636	.545	.823	.852	.743	.734	.620	.645		
LR	ISO	.627	.567	.818	.847	.735	.742	.608	.589		
LR	PLT	.630	.500	.823	.852	.743	.734	.593	.604		
NB	ISO	.579	.468	.779	.820	.727	.733	.572	.555	.654	.661
NB	PLT	.576	.448	.780	.824	.738	.735	.537	.559	.650	.654
NB	–	.496	.562	.781	.825	.738	.735	.347	-.633	.481	.489

Top 8 are all based on various extensions of decision trees

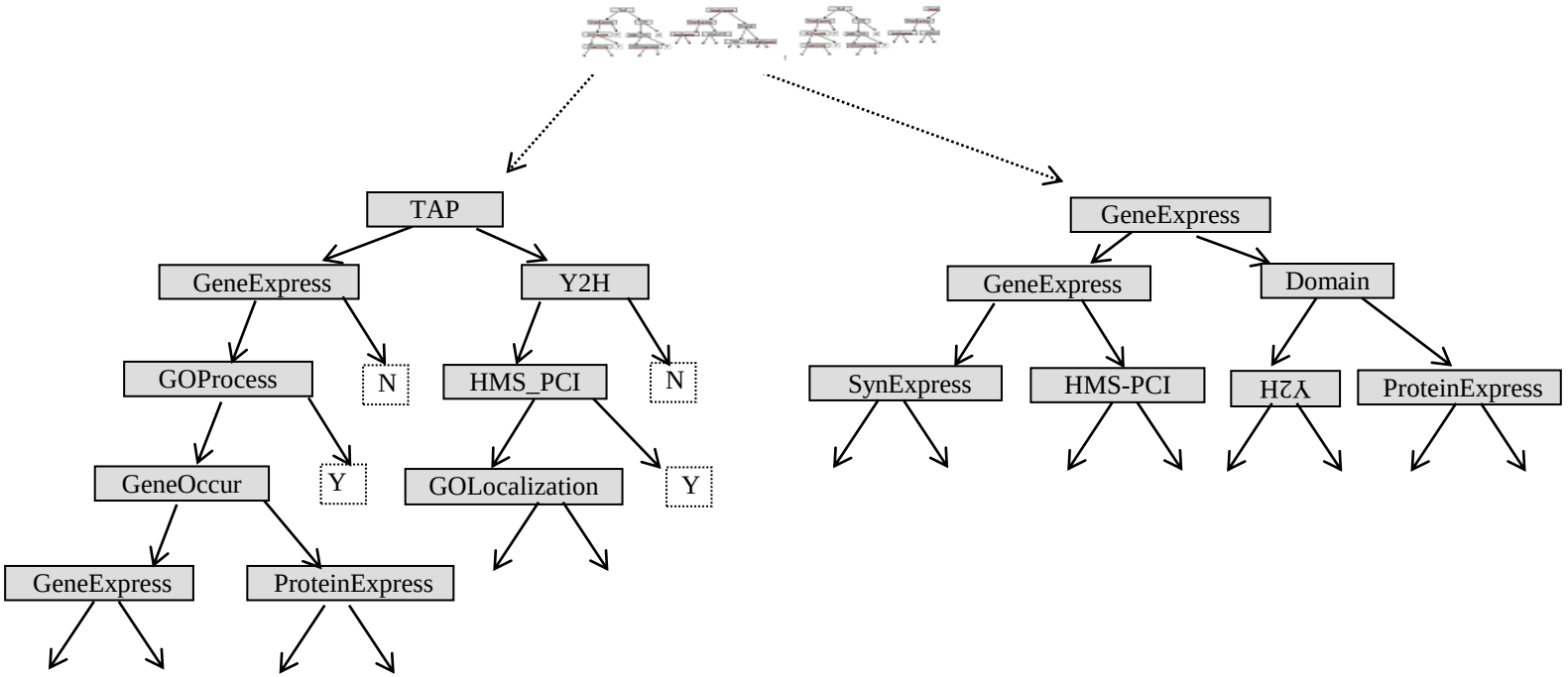
Rich Caruana & Alexandru Niculescu-Mizil, An Empirical Comparison of Supervised Learning Algorithms, ICML 2006

Important points

- Discriminative classifiers
- Entropy
- Information gain
- Building decision trees
- Optional reading: Murphy 16.1-16.2

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?