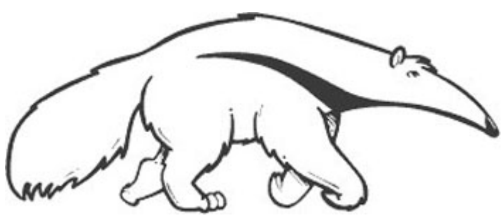
+

Machine Learning and Data Mining

Decision Trees

Prof. Alexander Ihler

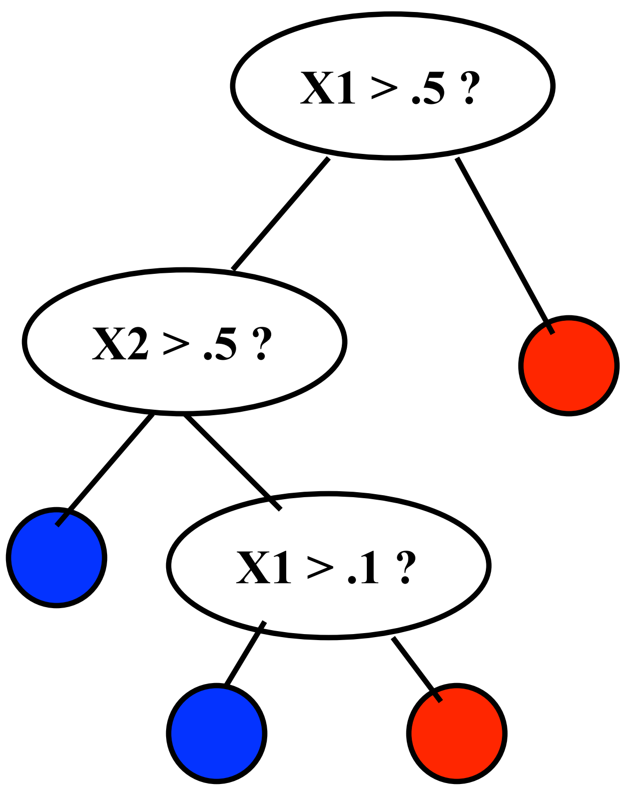


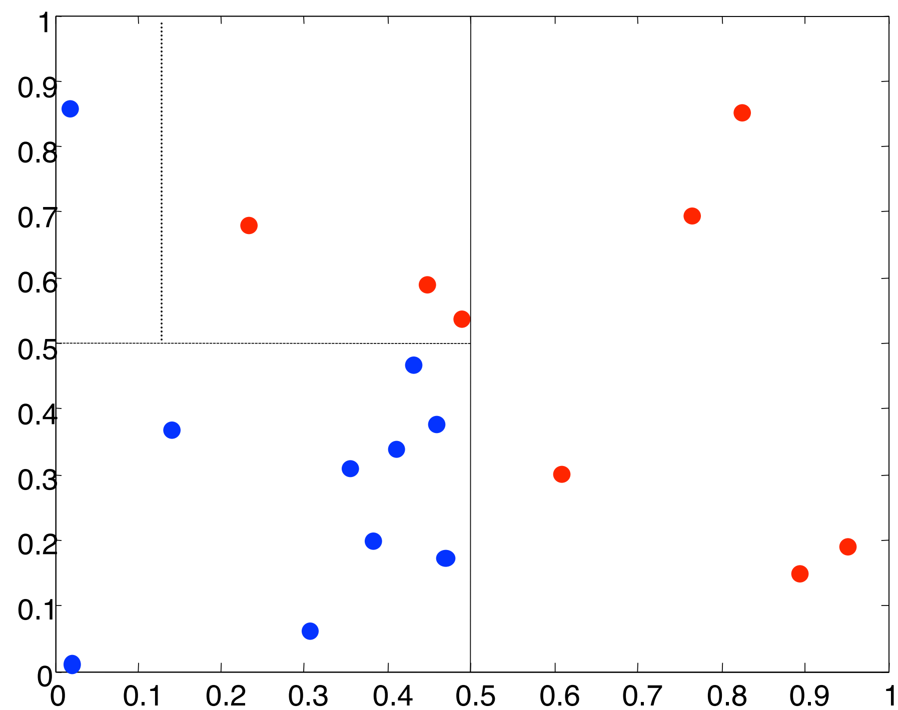
BREN:ICS
INFORMATION AND COMPUTER SCIENCES

UNIVERSITY of CALIFORNIA IRVINE

Decision trees

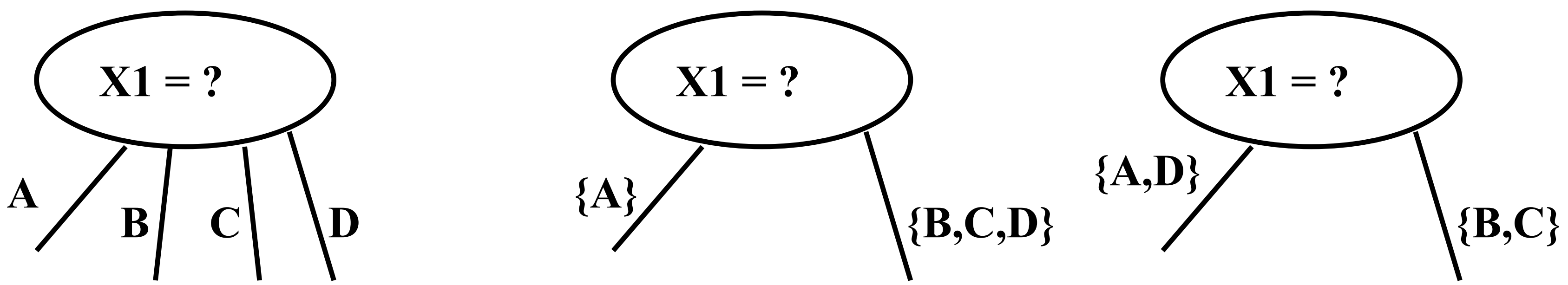
- "Split" input into cases
  - Usually based on a single variable
  - Recurse down until we reach a decision
  - Continuous vars: choose split point





Decision trees

- Categorical variables
  - Could have a child per value
  - Binary tree: split values into two sets

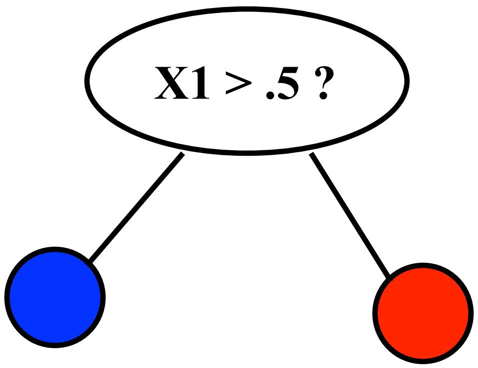


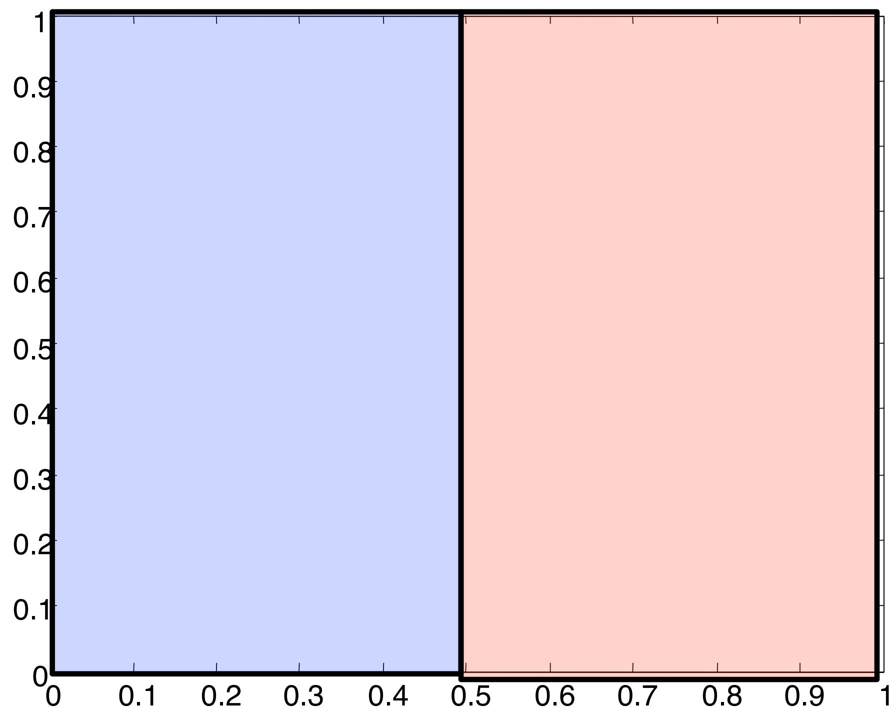
The discrete variable will not appear again below here…

Could appear again multiple times…

Decision trees

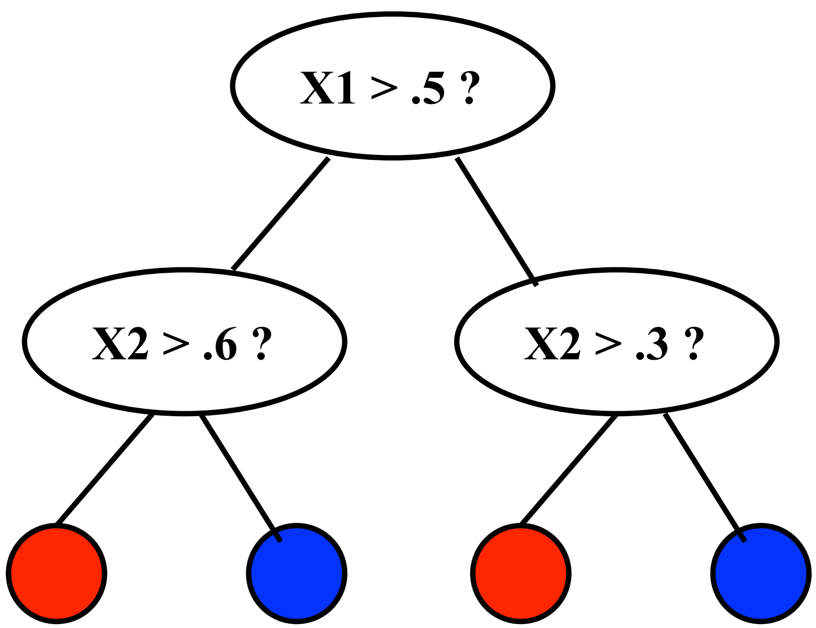
- "Complexity" of function depends on the depth
- A depth-1 decision tree is called a decision "stump"
  - Simpler than a linear classifier!

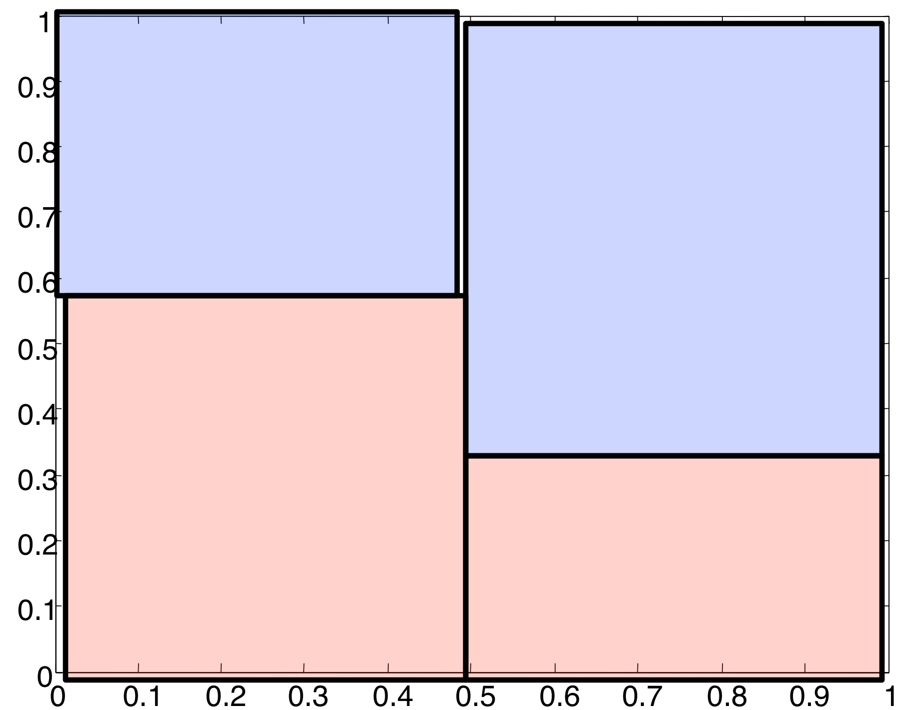




Decision trees

- "Complexity" of function depends on the depth





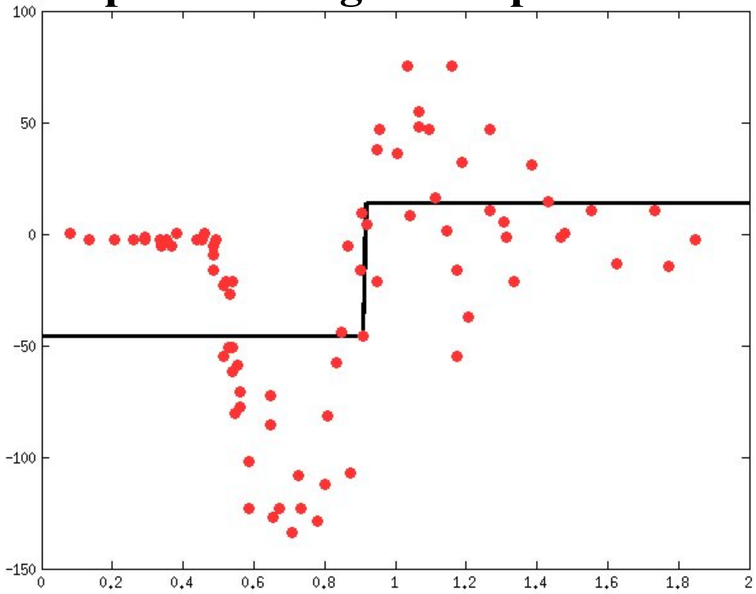
Depth d = up to 2^d regions & predictions

Decision trees for regression

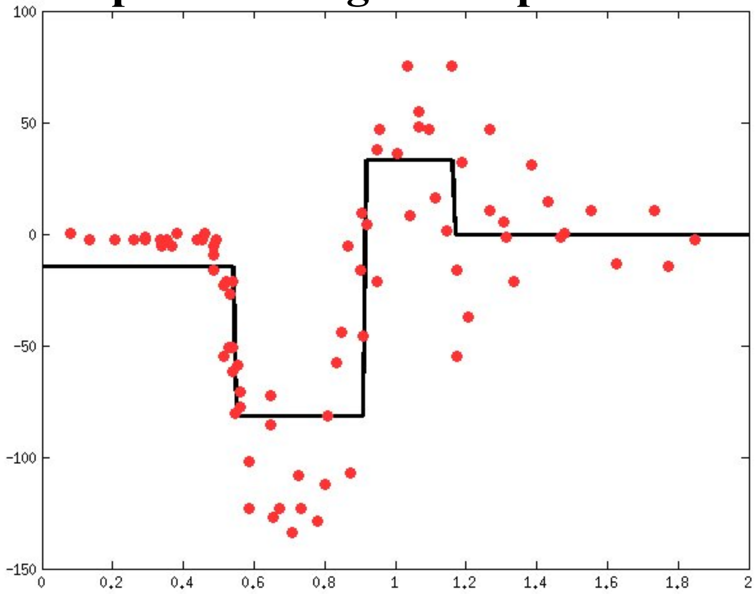
- Exactly the same
- Predict real valued numbers at leaf nodes

- Examples on a single scalar feature:

Depth 1 = 2 regions & predictions



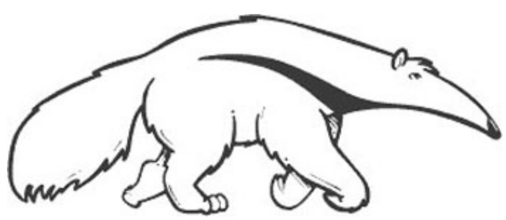
Depth 2 = 4 regions & predictions …



Machine Learning and Data Mining

Learning Decision Trees

Prof. Alexander Ihler



Learning decision trees

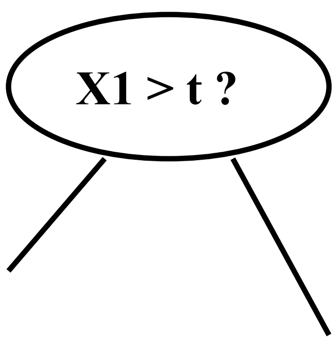
- Break into two parts
  - Should this be a leaf node?
  - If so: what should we predict?
  - If not: how should we further split the data?

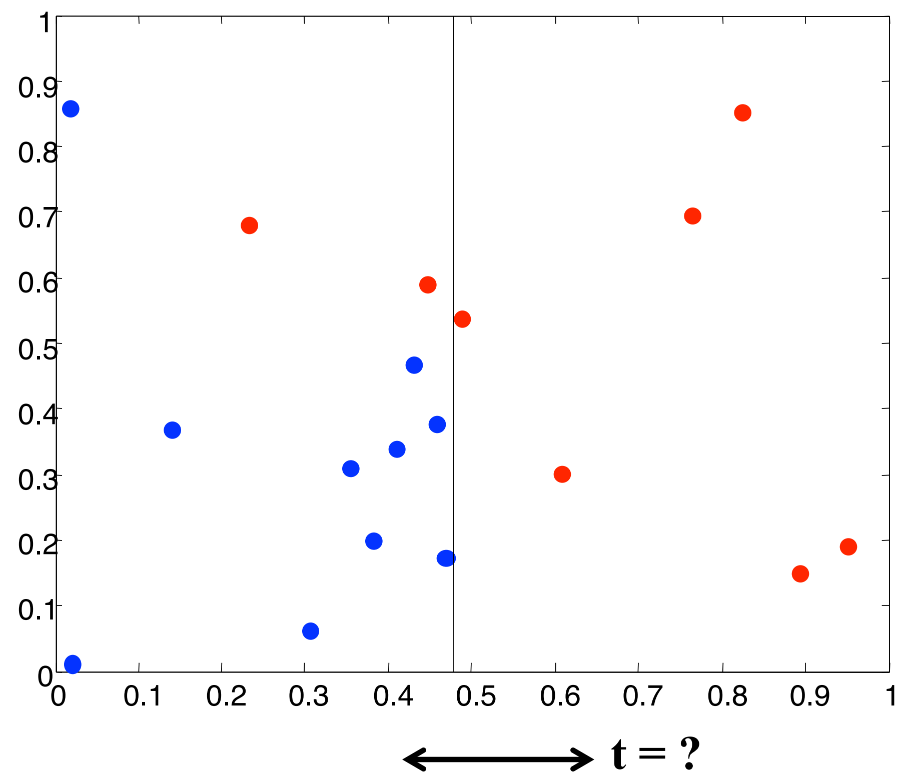
Example algorithms:
ID3, C4.5
See e.g. wikipedia, "Classification and regression tree"

- Leaf nodes: best prediction given this data subset
  - Classify: pick majority class; Regress: predict average value
- Non-leaf nodes: pick a feature and a split
  - Greedy: "score" all possible features and splits
  - Score function measures "purity" of data after split
    - How much easier is our prediction task after we divide the data?
- When to make a leaf node?
  - All training examples the same class (correct), or indistinguishable
  - Fixed depth (fixed complexity decision boundary)
  - Others …

Scoring decision tree splits

- Suppose we are considering splitting feature 1
  - How can we score any particular split?
- "Greedy" – could choose split with the best accuracy
  - Assume we have to predict a value next
  - MSE (regression)
  - 0/1 loss (classification)
- But: "soft" score can work better





Tree-structured splitting

- "CART" = classification and regression trees
  - A particular algorithm, but many similar variants
  - See e.g. http://en.wikipedia.org/wiki/Classification_and_regression_tree
  - Also ID3 and C4.5 algorithms
- Classification
  - Union of rectangular decision regions
  - Split criterion, e.g., information gain (or "cross-entropy")
  - Alternative: "Gini index" (similar properties)
- Regression
  - Divide input space ("x") into regions
  - Each region has its own regression function
  - Split criterion, e.g., predictive improvement

Entropy and Information

- "Entropy" is a measure of randomness
  - How hard is it to communicate a result to you?
  - Depends on the probability of the outcomes

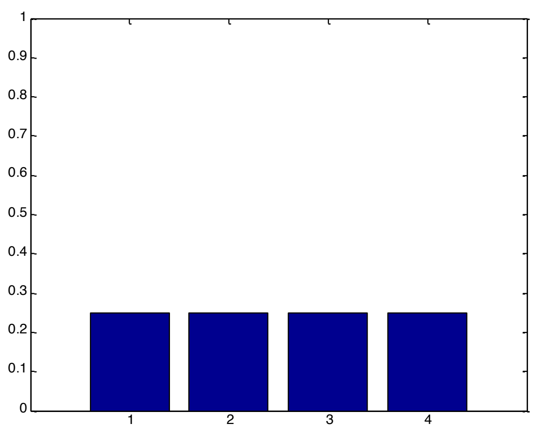
- Communicating fair coin tosses
  - Output: H H T H T T T H H H H T …
  - Sequence takes n bits – each outcome totally unpredictable

- Communicating my daily lottery results
  - Output: 0 0 0 0 0 0 …
  - Most likely to take one bit – I lost every day.
  - Small chance I'll have to send more bits (won & when)

- Takes less work to communicate because it's less random
  - Use a few bits for the most likely outcome, more for less likely ones`

Entropy and Information

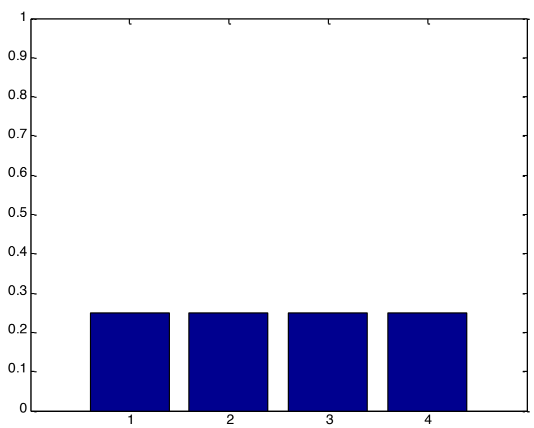
- Entropy $H(x) \equiv \mathbb{E}[\log 1/p(x)] = \sum p(x) \log 1/p(x)$
  - Log base two, units of entropy are “bits”
- Examples:

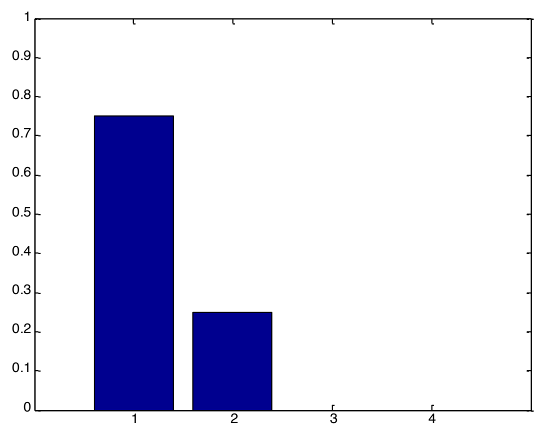


$H(x) = .25 \log 4 + .25 \log 4 +$
$.25 \log 4 + .25 \log 4$
$= \log 4 = 2$ bits

Entropy and Information

- Entropy $H(x) \equiv \mathbb{E}[\ \log 1/p(x)\ ] = \sum p(x) \log 1/p(x)$
  - Log base two, units of entropy are "bits"
- Examples:



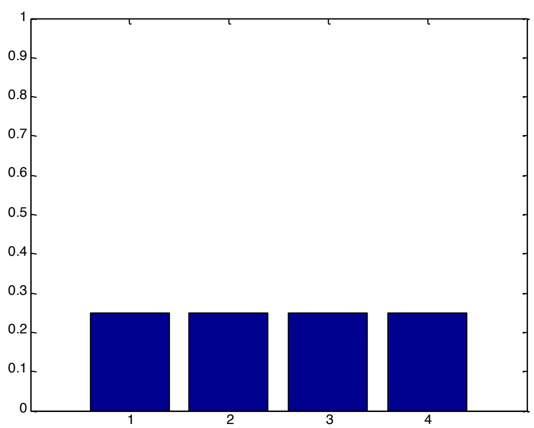


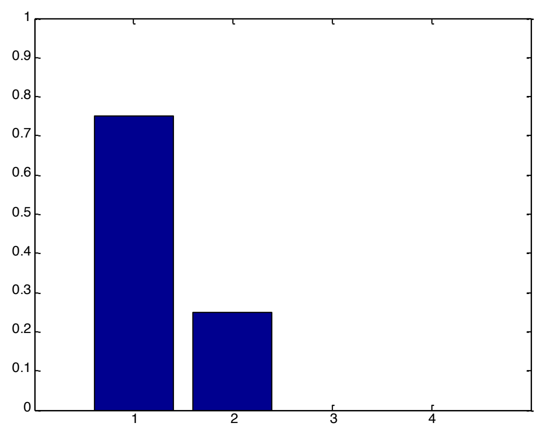
$H(x) = .25 \log 4 + .25 \log 4 +$
$.25 \log 4 + .25 \log 4$
$= \log 4 = 2$ bits

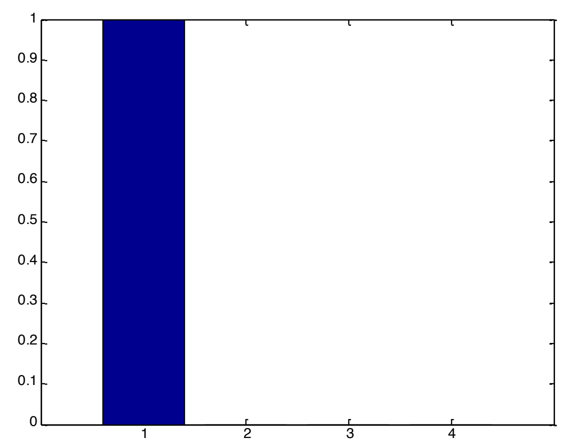
$H(x) = .75 \log 4/3 + .25 \log 4$
$\approx .8133$ bits

Entropy and Information

- Entropy $H(x) \equiv \mathbb{E}[\log 1/p(x)] = \sum p(x) \log 1/p(x)$
  - Log base two, units of entropy are "bits"
- Examples:







$H(x) = .25 \log 4 + .25 \log 4 + .25 \log 4 + .25 \log 4 = \log 4 = 2$ bits

Max entropy for 4 outcomes

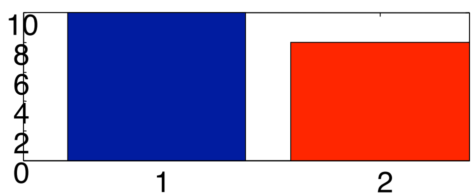
$H(x) = .75 \log 4/3 + .25 \log 4 \approx .8133$ bits

$H(x) = 1 \log 1 = 0$ bits

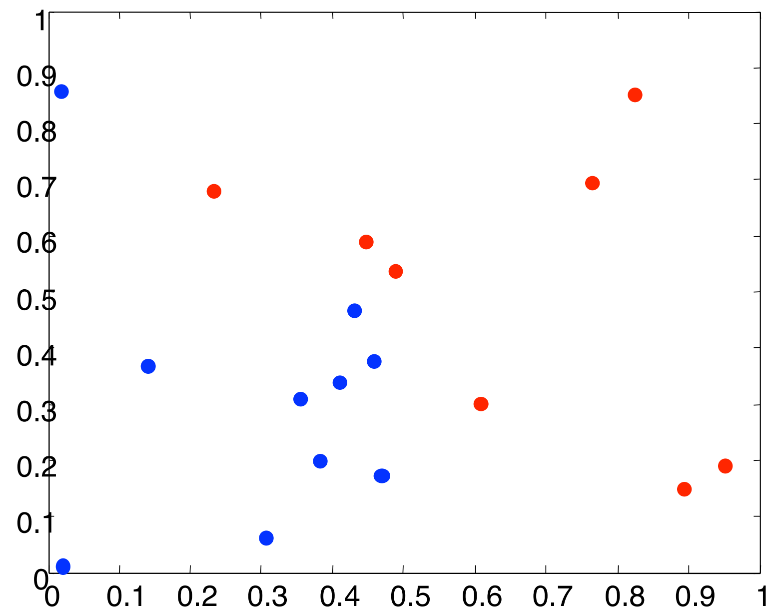
Min entropy

Entropy and Information

- Information gain
  - How much is entropy reduced by measurement?
- Information: expected information gain

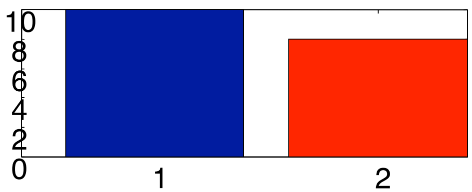


H = . 99 bits

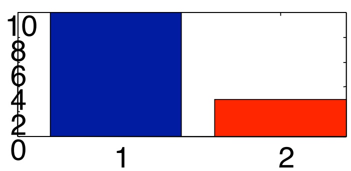


Entropy and Information

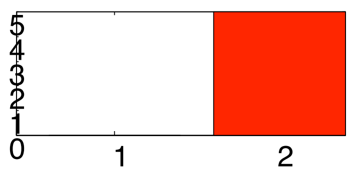
- Information gain
  - How much is entropy reduced by measurement?
- Information: expected information gain



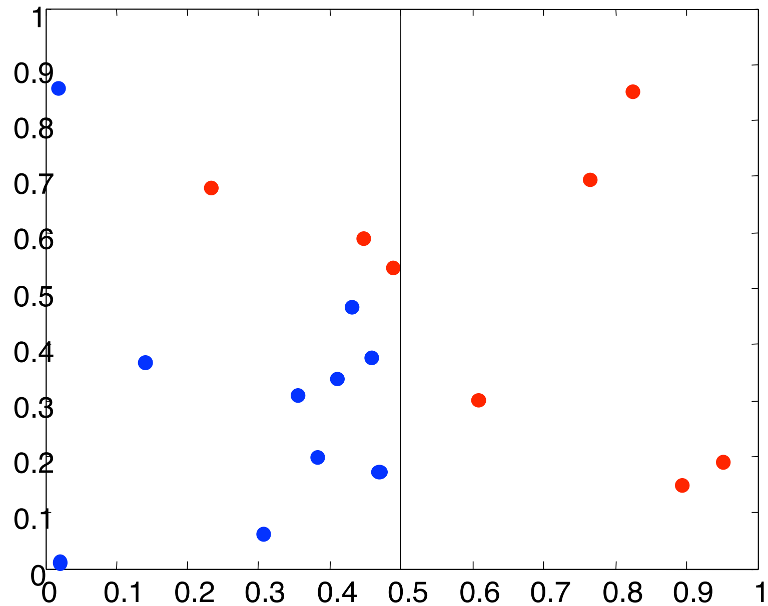
H = . 99 bits



H = .77 bits
Prob = 13/18

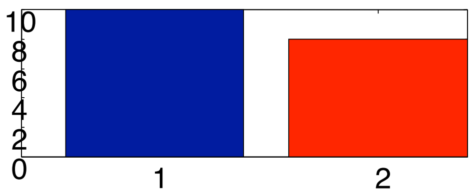


H=0
Prob = 5/18

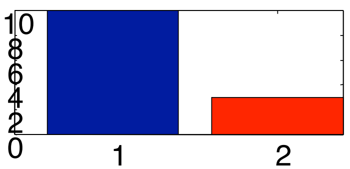


Entropy and Information

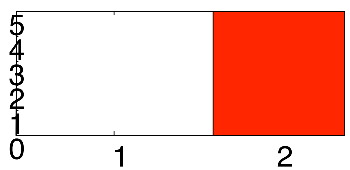
- Information gain
  - How much is entropy reduced by measurement?
- Information: expected information gain



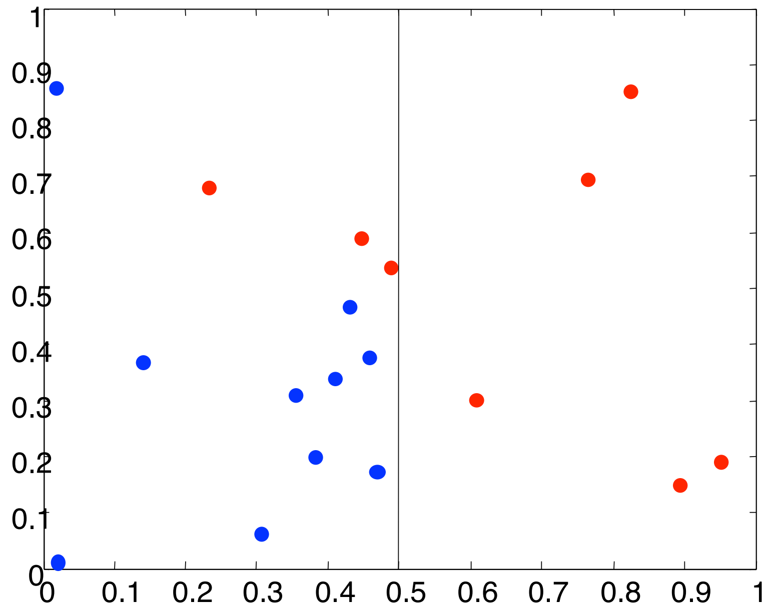
H = . 99 bits



H = .77 bits
Prob = 13/18



H=0
Prob = 5/18



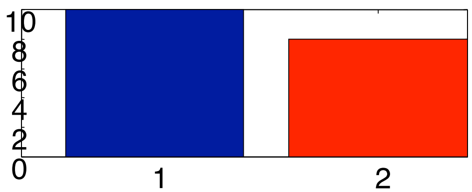
Information = 13/18 * (.99-.77) + 5/18 * (.99 – 0)

Equivalent: $\sum p(s,c) \log [\, p(s,c) / p(s)\, p(c) \,]$

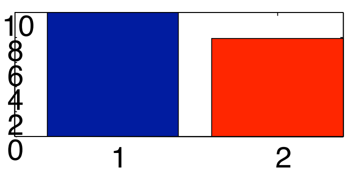
$= 10/18 \log[\, (10/18) / (13/18)\, (10/18)] + 3/18 \log[\, (3/18)/(13/18)(8/18) + \ldots$

Entropy and Information

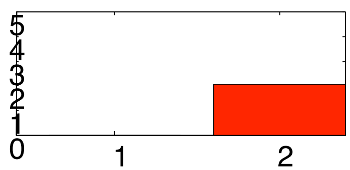
- Information gain
  - How much is entropy reduced by measurement?
- Information: expected information gain



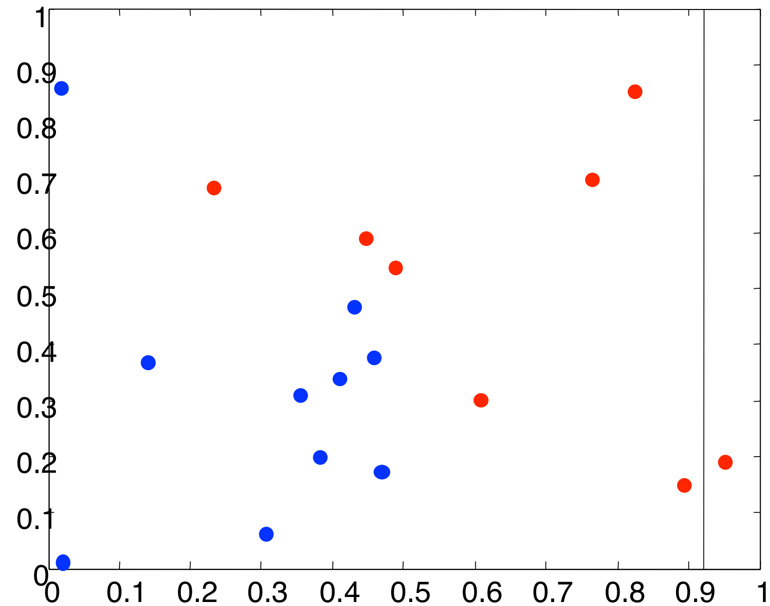
H = . 99 bits



H = .97 bits
Prob = 17/18

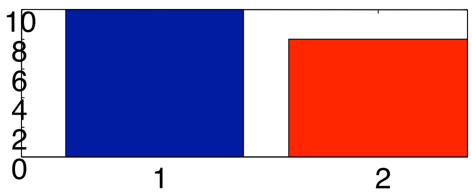


H=0
Prob = 1/18

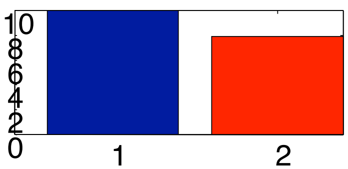


Entropy and Information

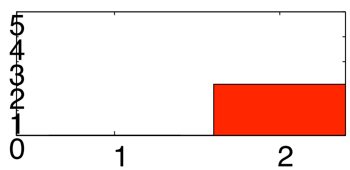
- Information gain
  – How much is entropy reduced by measurement?
- Information: expected information gain



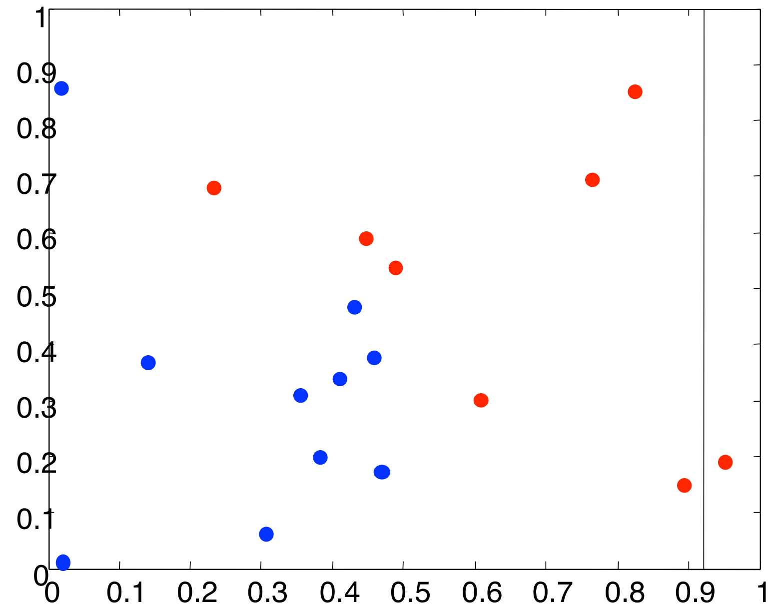
H = . 99 bits



H = .97 bits
Prob = 17/18



H=0
Prob = 1/18

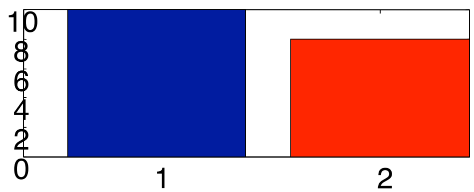


Information = 17/18 * (.99-.97) + 1/18 * (.99 – 0)

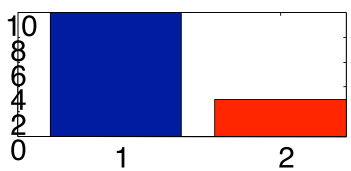
Less information reduction – a less desirable split of the data

Gini index / impurity

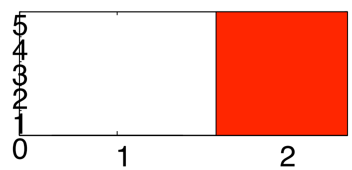
- An alternative to information gain
  - Measures variance in the allocation (instead of entropy)
- $H_{gini} = \sum_c p(c)\,(1\text{-}p(c))$ vs. $H_{ent} = -\sum_c p(c) \log p(c)$



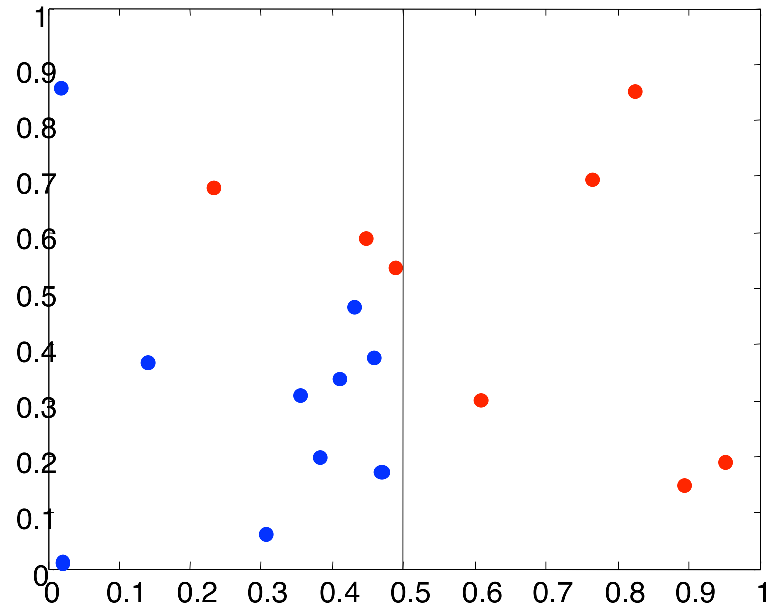
Hg = . 247



Hg = .198
Prob = 13/18



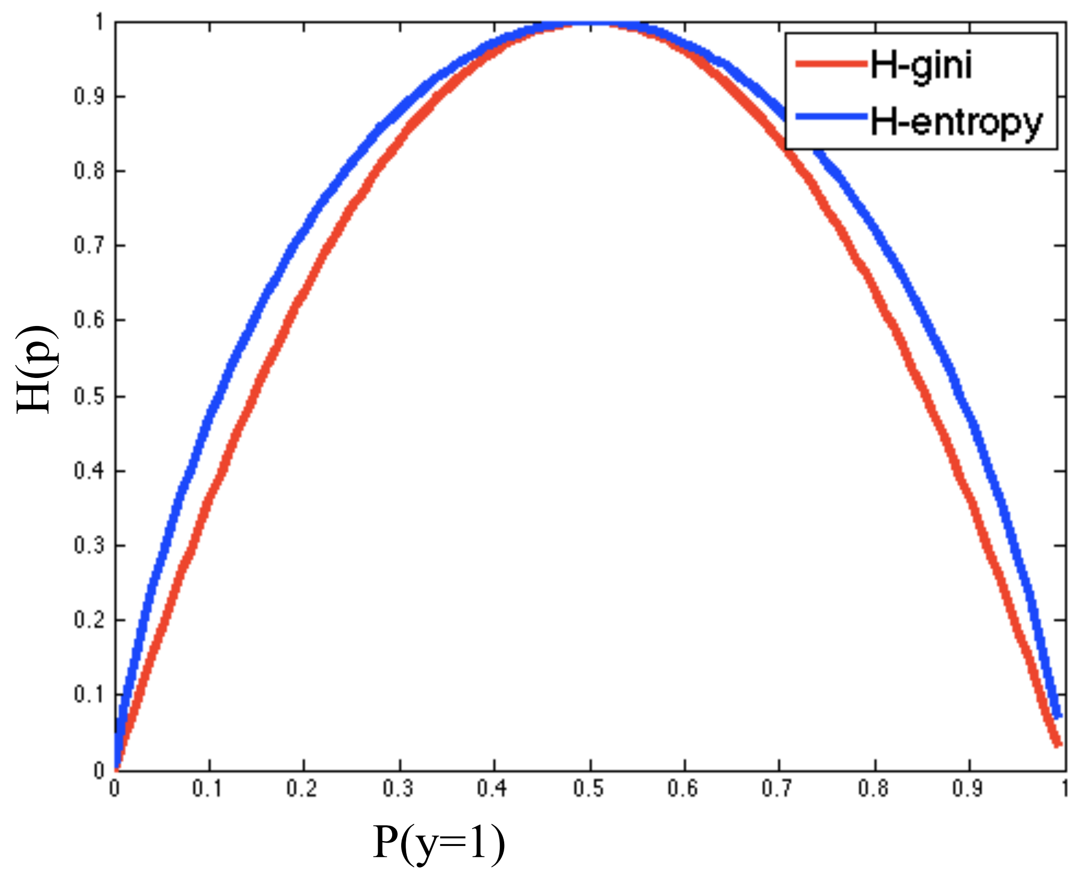
Hg = 0
Prob = 5/18



Gini Index = 13/18 * (.247-.198) + 5/18 * (.247 – 0)

Entropy vs Gini index

- The two are nearly the same…
  - Pick whichever one you like



Tree-structured splitting

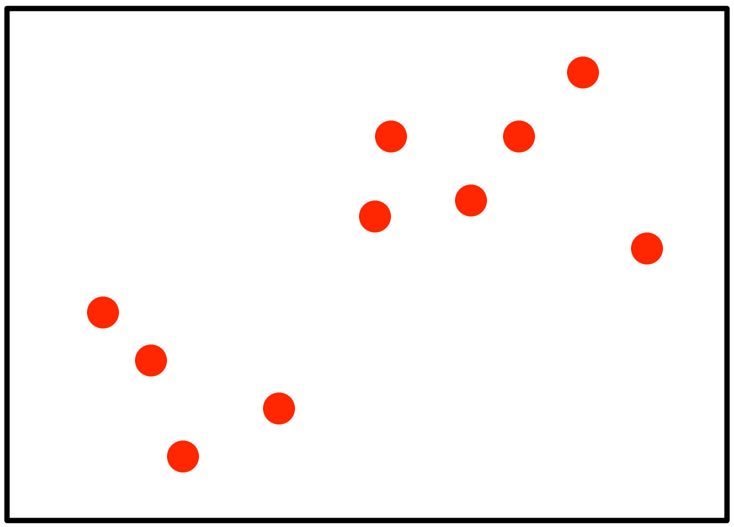
- "CART" = classification and regression trees
  - A particular algorithm, but many similar variants
  - See e.g. http://en.wikipedia.org/wiki/Classification_and_regression_tree
  - Also ID3 and C4.5 algorithms

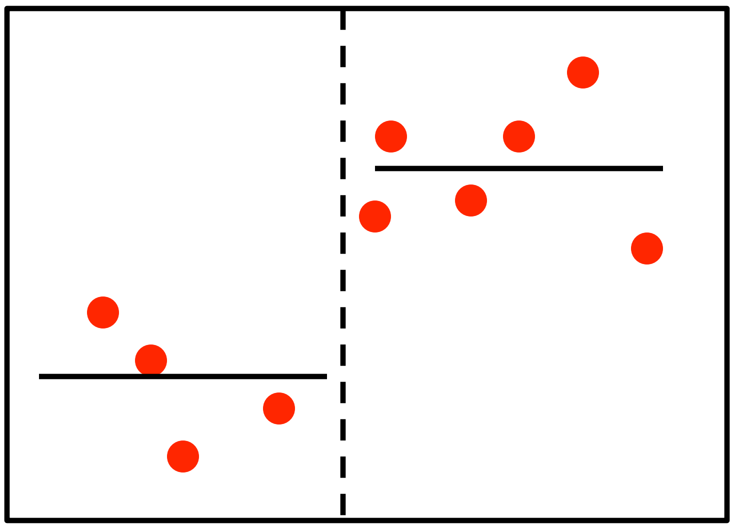
- Classification
  - Union of rectangular decision regions
  - Split criterion, e.g., information gain (or "cross-entropy")
  - Alternative: "Gini index" (similar properties)

- Regression
  - Divide input space ("x") into regions
  - Each region has its own regression function
  - Split criterion, e.g., predictive improvement

For regression

- Most common is to measure variance reduction
  - Equivalent to “information gain” in a Gaussian model…





Var reduction = 4/10 * (.25-.1) + 6/10 * (.25 – .2)

Building a decision tree

- Pseudo-code

```
decisionTreeSplitData(X,Y)
 if (stopping condition) return decision for this node
 For each possible feature
  For each possible split
   (for cts features: sort & compute split points)
   Score the split  (e.g. information gain)
 Pick the feature & split with the best score
 Split the data at that point
 Recurse on each subset
```

Stopping conditions:

* # of data < K
* Depth > D
* All data indistinguishable (discrete features)
* Prediction sufficiently accurate

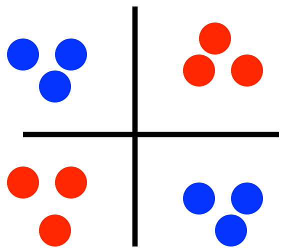
Building a decision tree

- Pseudo-code

```
decisionTreeSplitData(X,Y)
 if (stopping condition) return decision for this node
 For each possible feature
  For each possible split
   (for cts features: sort & compute split points)
   Score the split  (e.g. information gain)
 Pick the feature & split with the best score
 Split the data at that point
 Recurse on each subset
```

Stopping criteria:

- Information gain threshold? Often not a good idea…

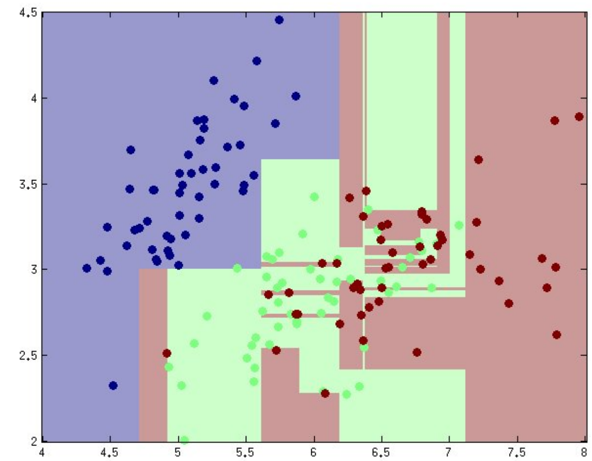


No single split improves performance, but two splits together is accurate

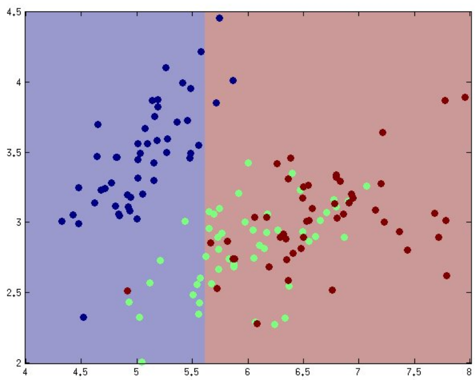
Instead: grow a large tree and prune back, using training or validation data

Controlling complexity

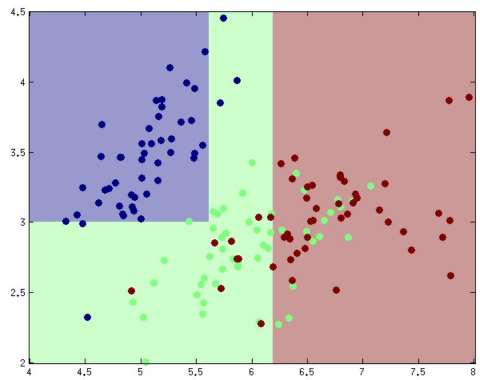
- Maximum depth cutoff



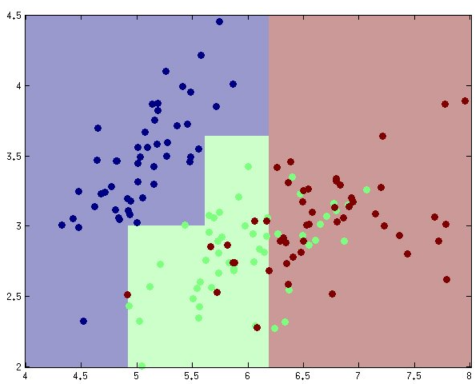
No limit



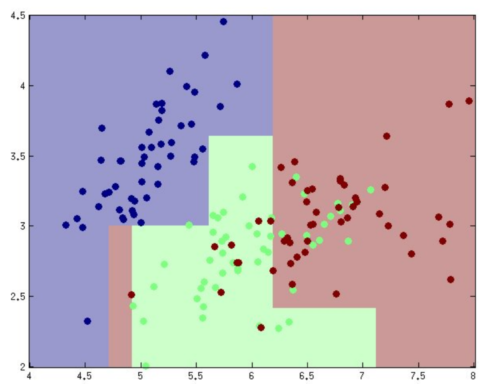
Depth 1



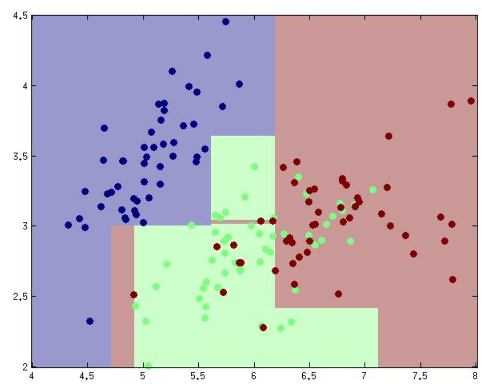
Depth 2



Depth 3



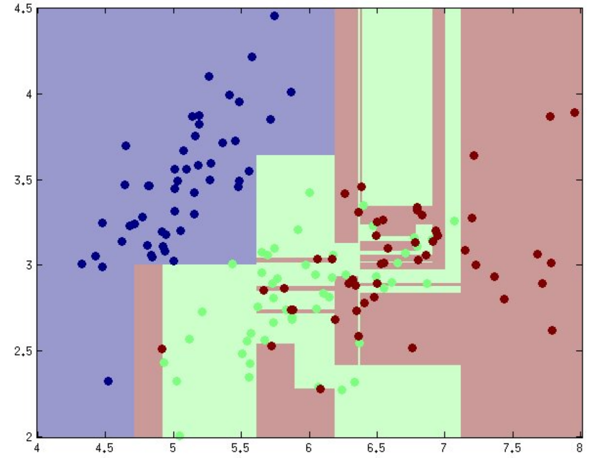
Depth 4



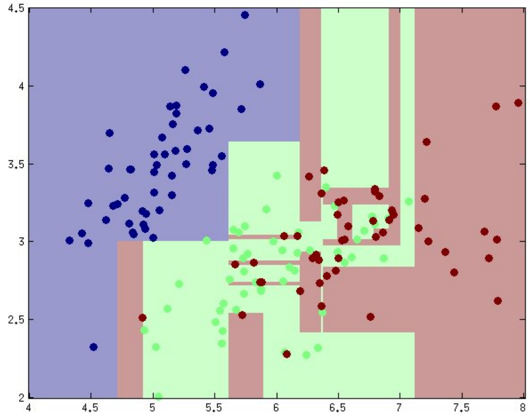
Depth 5

Controlling complexity

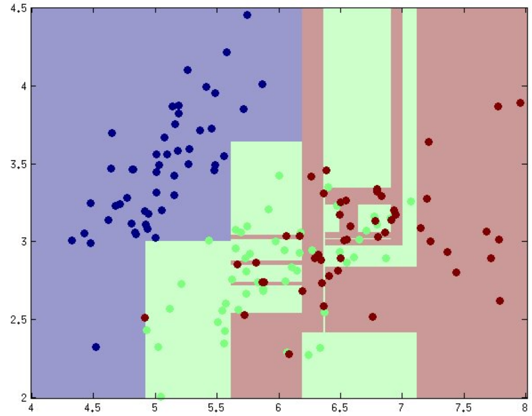
- Minimum # parent data



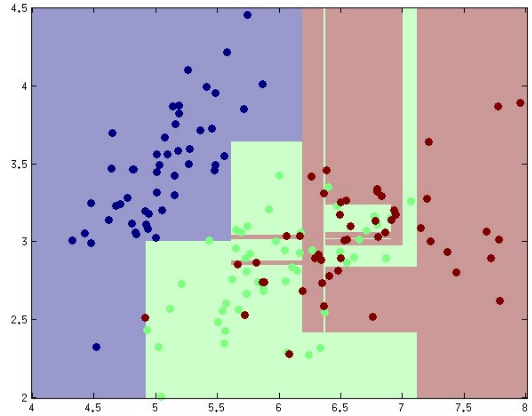
minParent 1



minParent 3



minParent 5



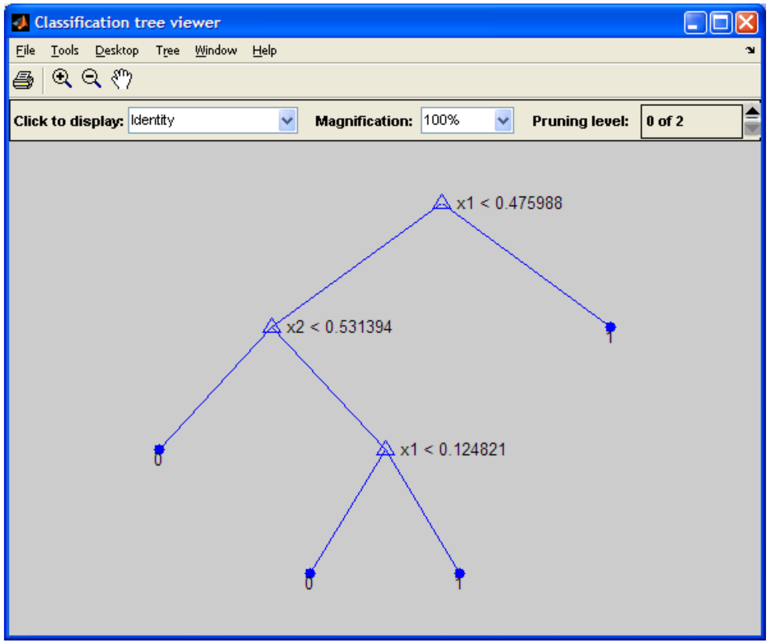
minParent 10

Decision trees in Matlab

- Stats toolbox
  - "classregtree" (older versions: "treefit")
  - If Y is a double array, regression
  - If Y is a logical array (or other things), classify
    - Uses Gini index by default

```
T=classregtree(x,logical(y),'splitmin',1);
% was T=treefit(...)
T,
Decision tree for classification
1   If x1 < 0.475 then node 2 else node 3
2   If x2 < 0.531 then node 4 else nod 5
3   class = 1
4   class = 0
5   If x1 < .125 then node 6 else node 7
6   class = 0
7   class = 1

view(T);    % was treedisp(T)
% NOTE: test returns 1 or 2, not 0 or 1...
```



Summary

- Decision trees
  - Flexible functional form
  - At each level, pick a variable and split condition
  - At leaves, predict a value

- Learning decision trees
  - Score all splits & pick best
    - Classification: Information gain, Gini index
    - Regression: Expected variance reduction
  - Stopping criteria

- Complexity depends on depth
  - Decision stumps: very simple classifiers