Decision Trees

Aarti Singh

Machine Learning 10-701/15-781
Oct 6 , 2010

MACHINE LEARNING DEPARTMENT
Carnegie Mellon.
School of Computer Science

Learning a good prediction rule

- Learn a mapping $f : \mathcal{X} \rightarrow \mathcal{Y}$
- Best prediction rule $f^*(X) = \arg\min_f R(f)$
- Hypothesis space/Function class $\mathcal{F}$
  - Parametric classes (Gaussian, binomial etc.)
  - Conditionally independent class densities (Naïve Bayes)
  - Linear decision boundary (Logistic regression)
  - Nonparametric class (Histograms, nearest neighbor, kernel estimators, **Decision Trees – Today**)
- Given training data, find a hypothesis/function in $\mathcal{F}$ that is close to the best prediction rule.

$$\widehat{f}_n(X) = \arg\min_{\underline{f \in \mathcal{F}}} \widehat{R}_n(f) + C(f)$$

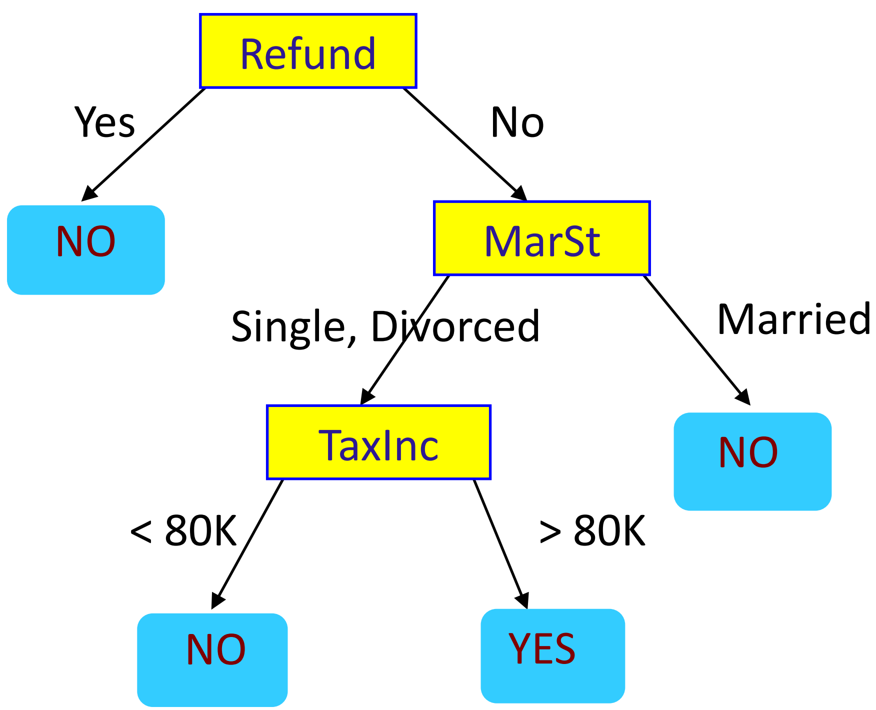
First …

- What does a decision tree represent
- Given a decision tree, how do we assign label to a test point

Decision Tree for Tax Fraud Detection

$\mathcal{F}$ – Decision Trees

$f(X_1, X_2, X_3) \in \mathcal{F}$



Query Data

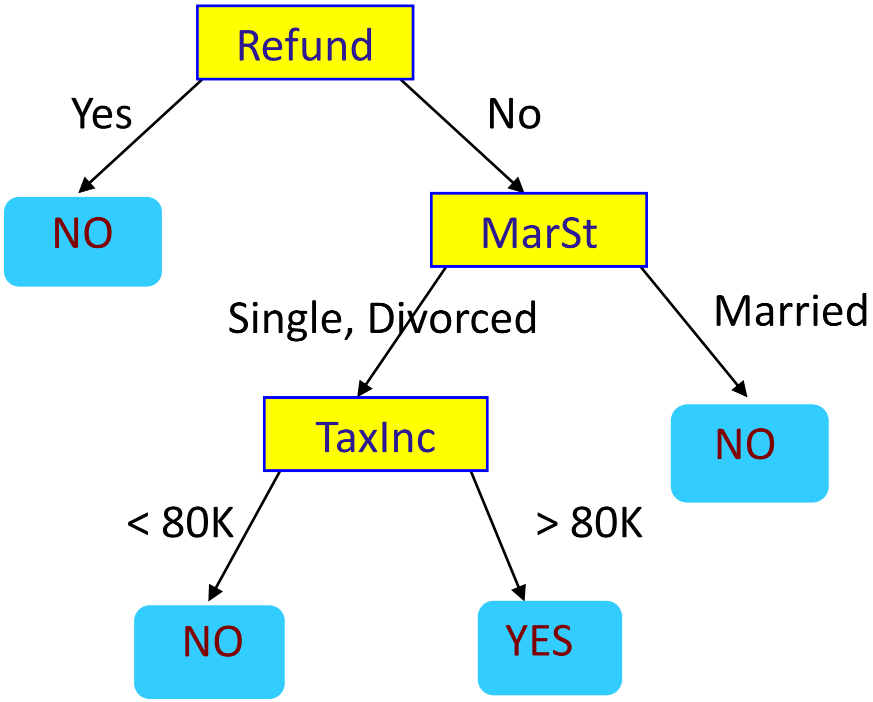
X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Married	80K	?

- Each internal node: test one feature X_i
- Each branch from a node: selects one value for X_i
- Each leaf node: predict Y

Decision Tree for Tax Fraud Detection

$\mathcal{F}$ – Decision Trees

$f(X_1, X_2, X_3) \in \mathcal{F}$



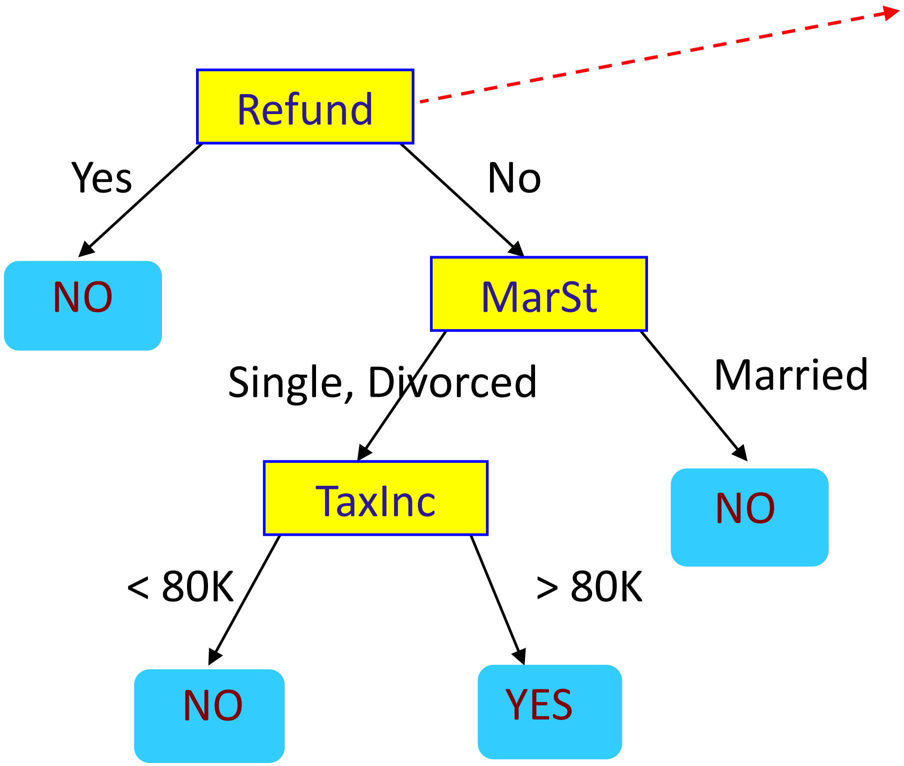
Query Data

X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Married	80K	?

Decision Tree for Tax Fraud Detection

$\mathcal{F}$ – Decision Trees

$f(X_1, X_2, X_3) \in \mathcal{F}$



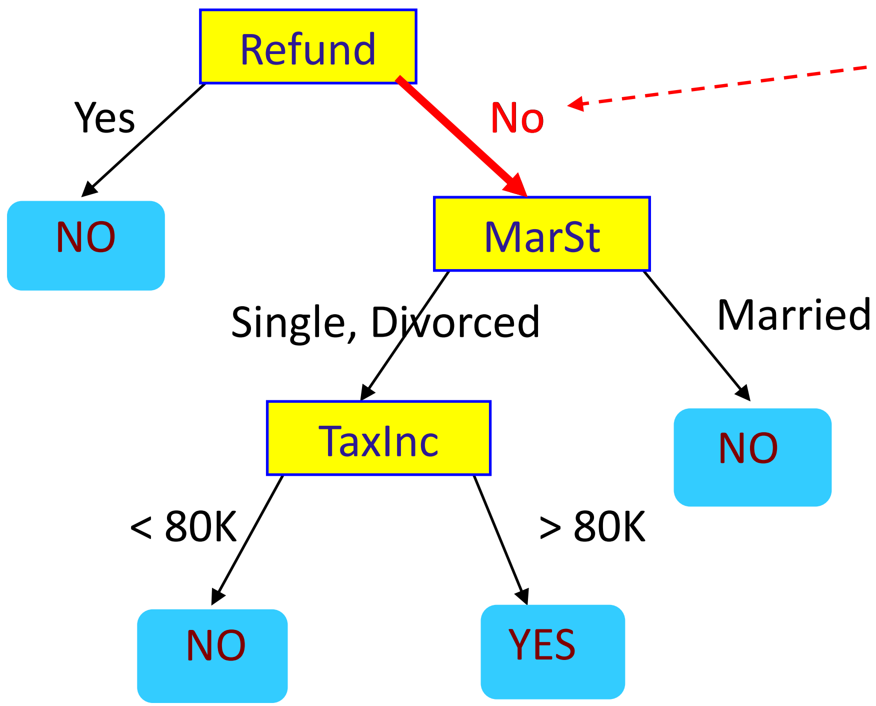
Query Data

X_1	X_2	X_3	Y
Refund	**Marital Status**	**Taxable Income**	**Cheat**
No	Married	80K	**?**

Decision Tree for Tax Fraud Detection

$\mathcal{F}$ – Decision Trees

$$f(X_1, X_2, X_3) \in \mathcal{F}$$



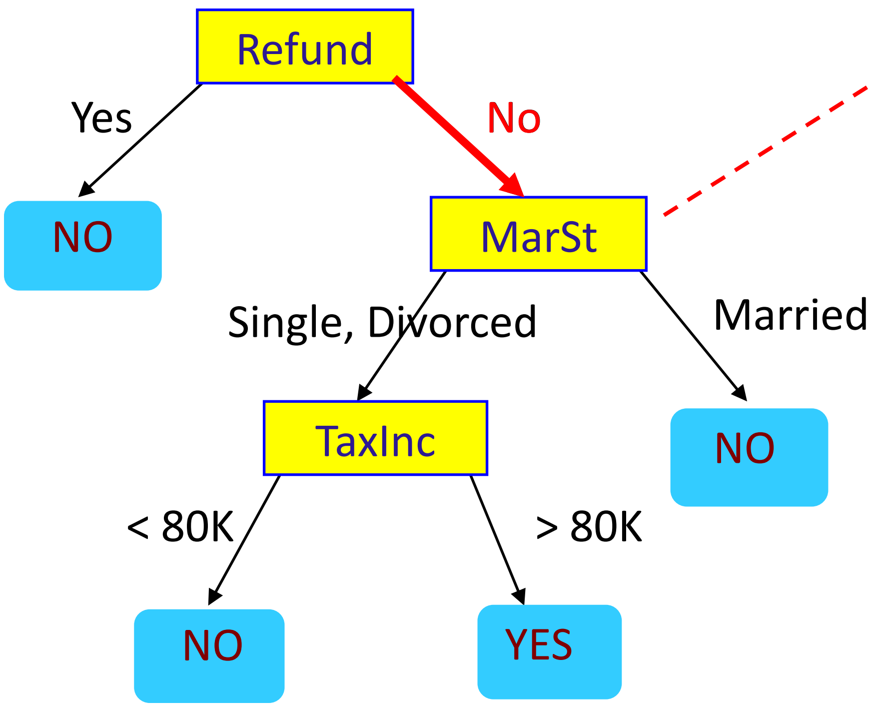
Query Data

X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Married	80K	?

Decision Tree for Tax Fraud Detection

$$\mathcal{F} - \text{Decision Trees}$$

$$f(X_1, X_2, X_3) \in \mathcal{F}$$



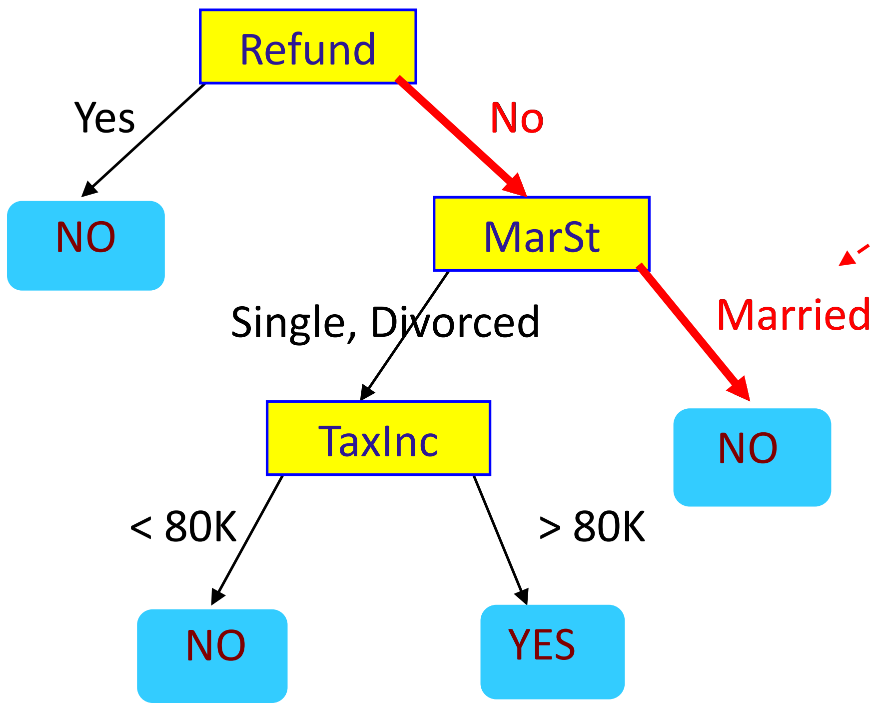
Query Data

X_1	X_2	X_3	Y
Refund	**Marital Status**	**Taxable Income**	**Cheat**
No	Married	80K	**?**

Decision Tree for Tax Fraud Detection

$\mathcal{F}$ – Decision Trees

$f(X_1, X_2, X_3) \in \mathcal{F}$



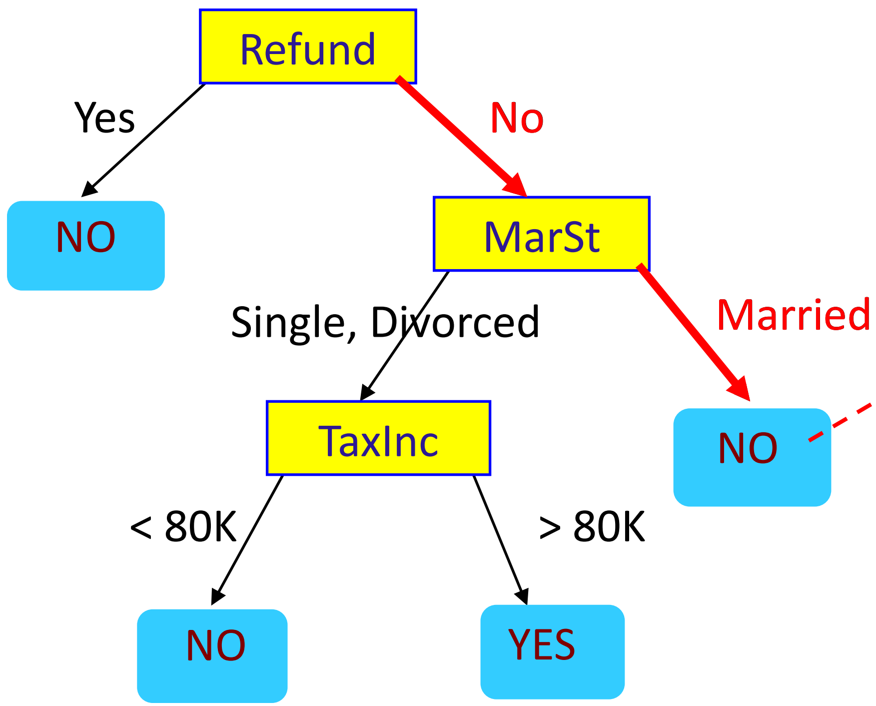
Query Data

X_1	X_2	X_3	Y
Refund	**Marital Status**	**Taxable Income**	**Cheat**
No	Married	80K	**?**

Decision Tree for Tax Fraud Detection

$\mathcal{F}$ – Decision Trees

$f(X_1, X_2, X_3) \in \mathcal{F}$

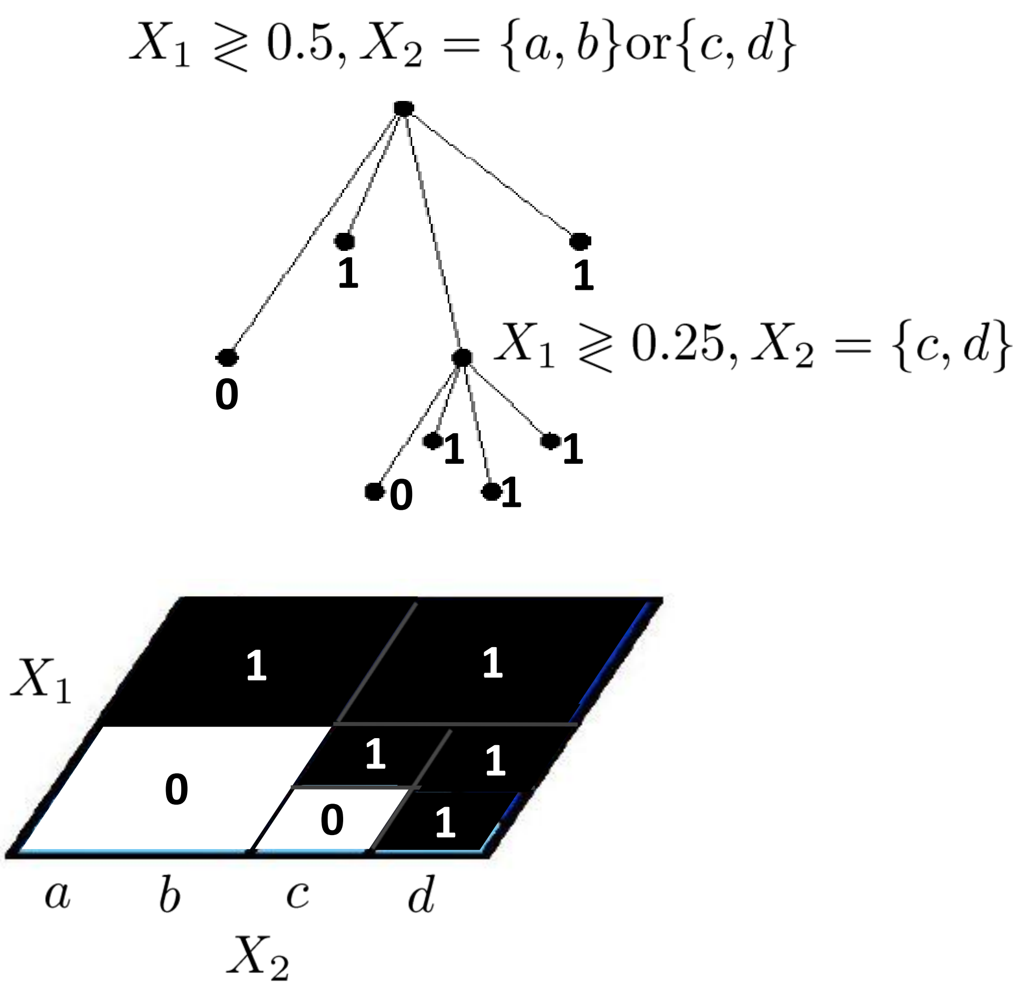


Query Data

X_1 Refund	X_2 Marital Status	X_3 Taxable Income	Y Cheat
No	Married	80K	?

Assign Cheat to "No"

Decision Tree more generally…



- Features can be discrete, continuous or categorical
- Each internal node: test some set of features $\{X_i\}$
- Each branch from a node: selects a set of value for $\{X_i\}$
- Each leaf node: predict Y

So far…

- What does a decision tree represent
- Given a decision tree, how do we assign label to a test point

Now …

- How do we learn a decision tree from training data
- What is the decision on each leaf

So far…

- What does a decision tree represent
- Given a decision tree, how do we assign label to a test point

Now …

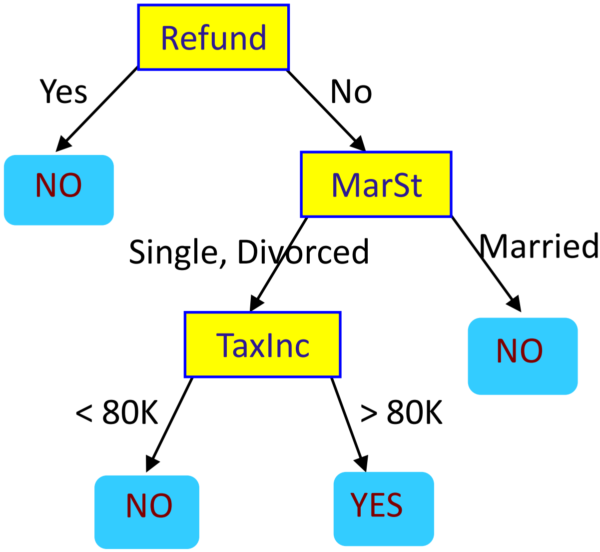
- How do we learn a decision tree from training data
- What is the decision on each leaf

How to learn a decision tree

- Top-down induction [ID3, C4.5, CART, …]

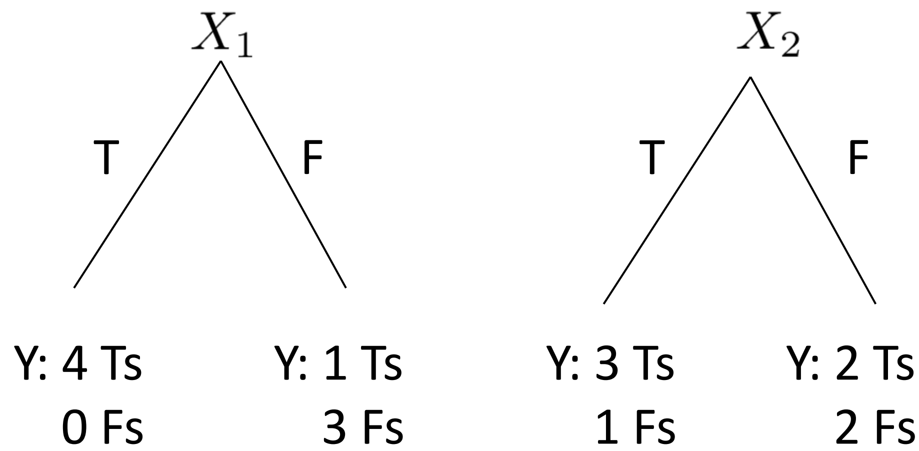
Main loop:

1. $X \leftarrow$ the "best" decision attribute for next *node*
2. Assign X as decision attribute for *node*
3. For each value of X, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes



Which feature is best to split?

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F



Absolutely sure

Kind of sure

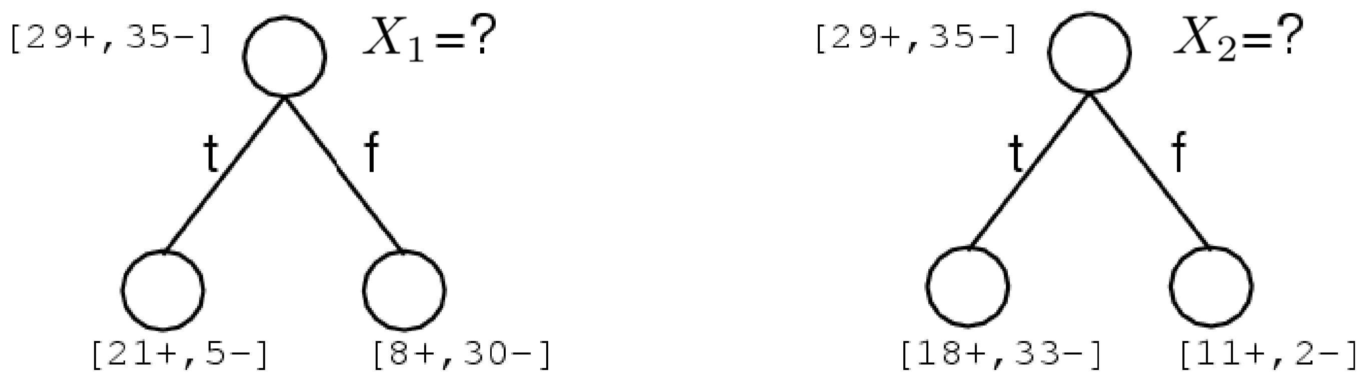
Kind of sure

Absolutely unsure

Good split if we are more certain about classification after split –

Uniform distribution of labels is bad

Which feature is best to split?



Pick the attribute/feature which yields maximum information gain:

$$\arg\max_i I(Y, X_i) = \arg\max_i [H(Y) - H(Y|X_i)]$$

H(Y) – entropy of Y   H(Y|X_i) – conditional entropy of Y

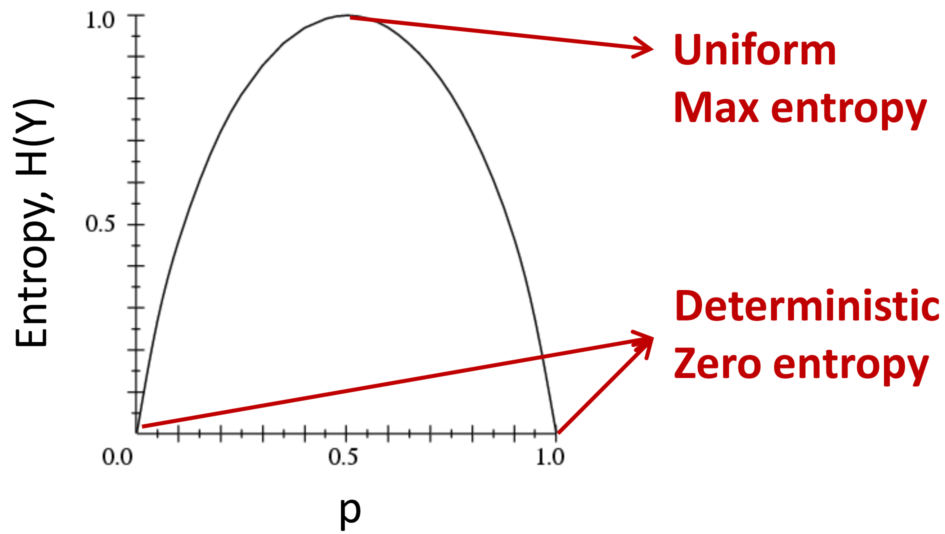
Entropy

- Entropy of a random variable Y

$$H(Y) = -\sum_y P(Y=y)\log_2 P(Y=y)$$

More uncertainty, more entropy!

Y ~ Bernoulli(p)



Information Theory interpretation: $H(Y)$ is the expected number of bits needed to encode a randomly drawn value of Y (under most efficient code)

Andrew Moore's Entropy in a Nutshell

Low Entropy

..the values (locations of soup) sampled entirely from within the soup bowl

High Entropy

..the values (locations of soup) unpredictable... almost uniformly sampled throughout our dining room

Information Gain

- Advantage of attribute = decrease in uncertainty
  - Entropy of Y before split

$$H(Y) = -\sum_y P(Y=y)\log_2 P(Y=y)$$

  - Entropy of Y after splitting based on X_i
    - Weight by probability of following each branch

$$\begin{aligned} H(Y \mid X_i) &= -\sum_x P(X_i = x) H(Y \mid X_i = x) \\ &= -\sum_x P(X_i = x) \sum_y P(Y = y \mid X_i = x) \log_2 P(Y = y \mid X_i = x) \end{aligned}$$

- Information gain is difference

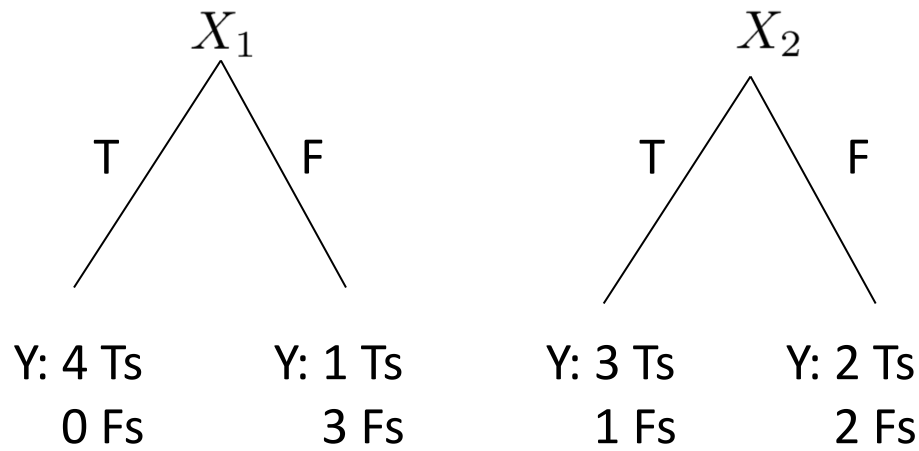
$$I(Y, X_i) = H(Y) - H(Y \mid X_i)$$

Max Information gain = min conditional entropy

Information Gain

$$H(Y \mid X_i) \;=\; -\sum_x P(X_i = x) \sum_y P(Y = y \mid X_i = x) \log_2 P(Y = y \mid X_i = x)$$

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F



$$\widehat{H}(Y|X_1) = -\cancel{\frac{1}{2}[1\log_2 1 + 0\log_2 0]} - \frac{1}{2}[\frac{1}{4}\log_2\frac{1}{4} + \frac{3}{4}\log_2\frac{3}{4}]$$

$$\widehat{H}(Y|X_2) = -\frac{1}{2}[\frac{3}{4}\log_2\frac{3}{4} + \frac{1}{4}\log_2\frac{1}{4}] \underbrace{- \frac{1}{2}[\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}]}_{>0}$$

$$\widehat{H}(Y|X_1) < \widehat{H}(Y|X_2)$$

Which feature is best to split?

Pick the attribute/feature which yields maximum information gain:

$$\arg\max_i I(Y, X_i) = \arg\max_i [H(Y) - H(Y|X_i)]$$

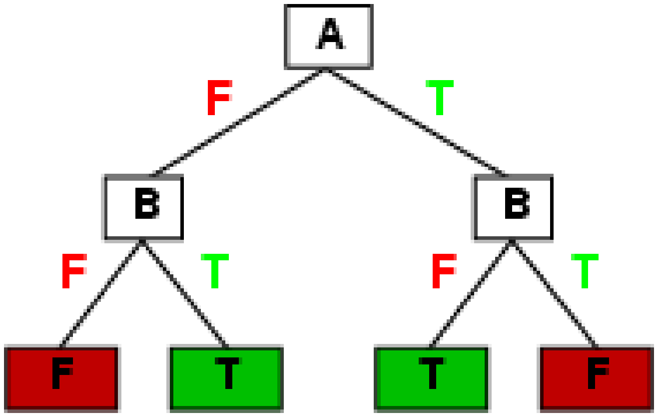
$H(Y)$ – entropy of Y $H(Y|X_i)$ – conditional entropy of Y

Feature which yields maximum reduction in entropy provides maximum information about Y

Expressiveness of Decision Trees

- Decision trees can express any function of the input features.
- E.g., for Boolean functions, truth table row → path to leaf:

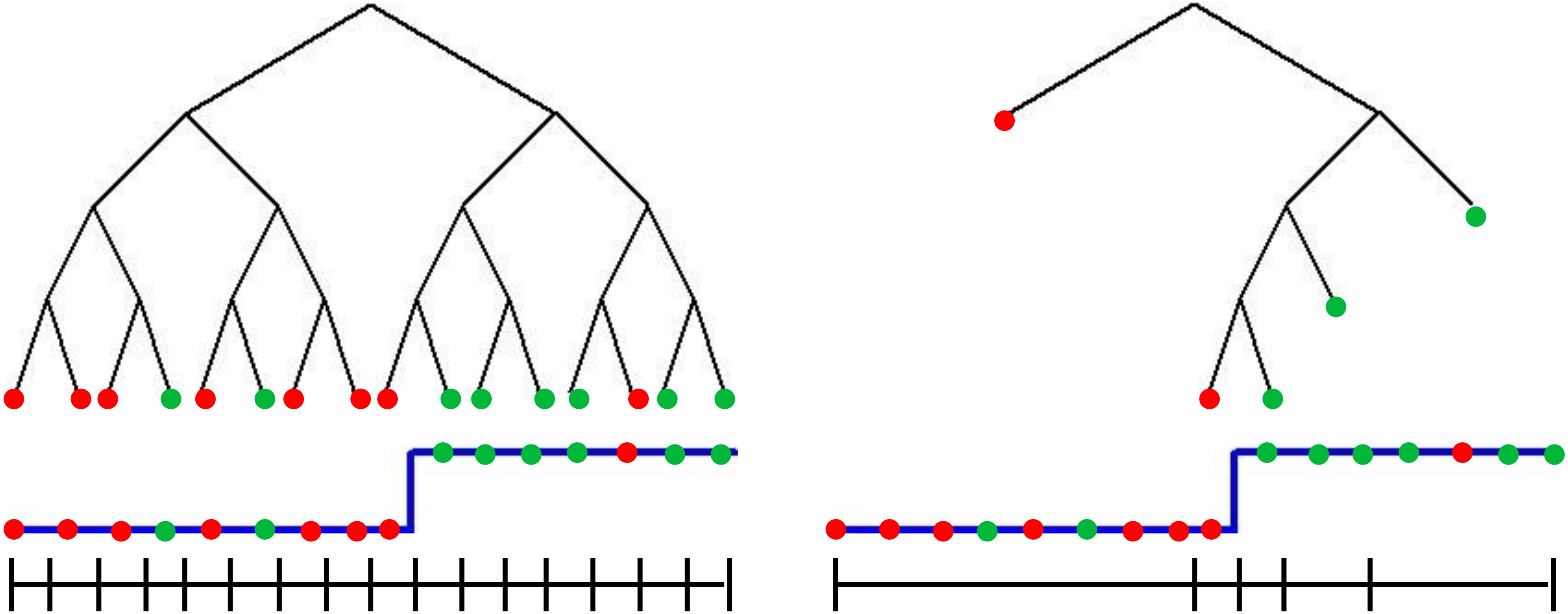
A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



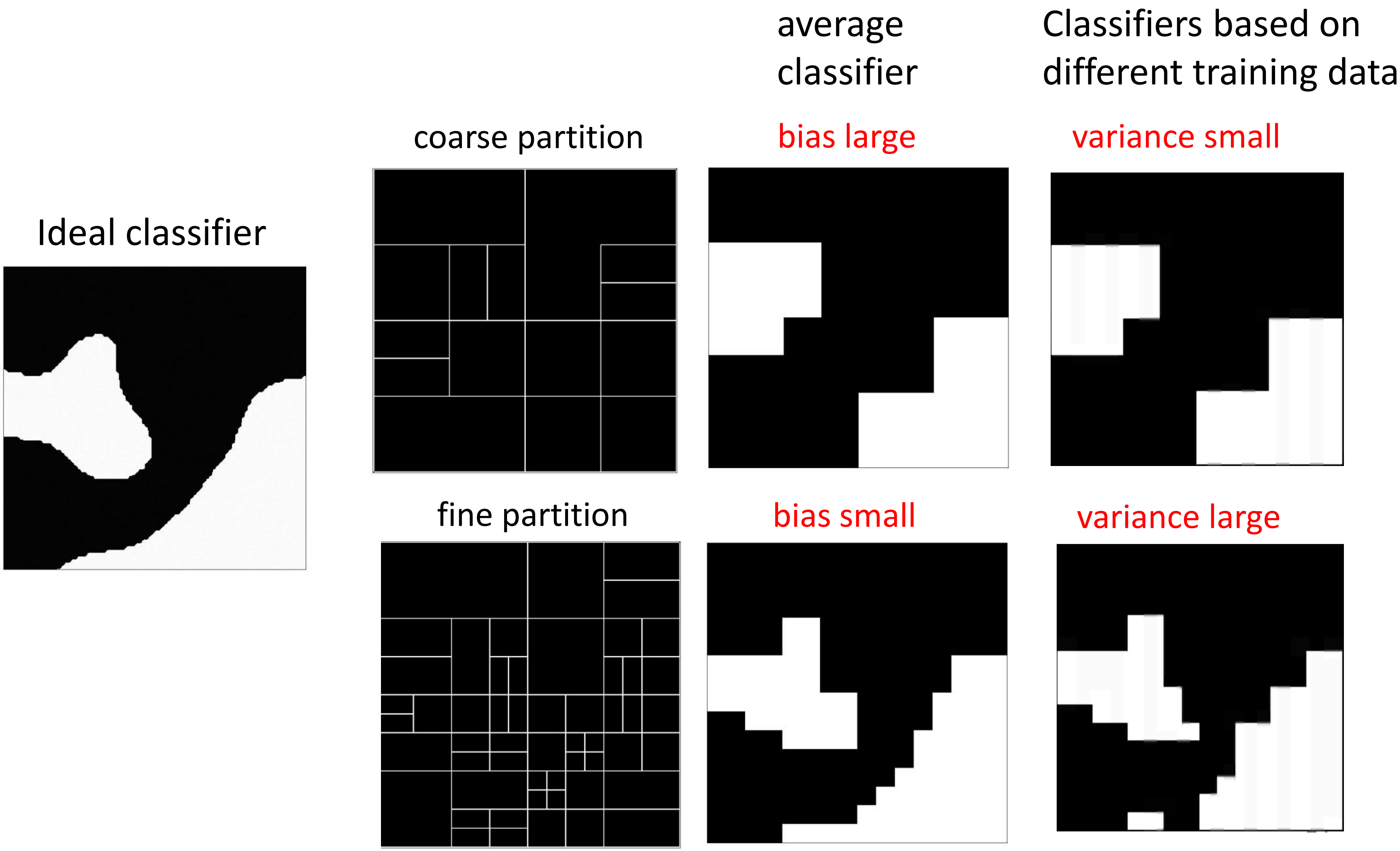
- There is a decision tree which perfectly classifies a training set with one path to leaf for each example
- But it won't generalize well to new examples - prefer to find more compact decision trees

Decision Trees - Overfitting

One training example per leaf – overfits, need compact/pruned decision tree

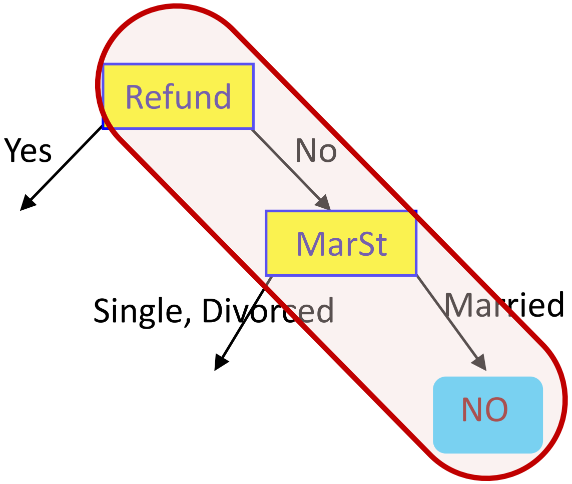


Bias-Variance Tradeoff



When to Stop?

- Many strategies for picking simpler trees:
  - Pre-pruning
    - Fixed depth
    - Fixed number of leaves



  - Post-pruning
    - Chi-square test
      - Convert decision tree to a set of rules
      - Eliminate variable values in rules which are independent of label (using chi-square test for independence)
      - Simplify rule set by eliminating unnecessary rules
  - Information Criteria: MDL(Minimum Description Length)

Information Criteria

- Penalize complex models by introducing cost

$$\widehat{f} = \arg\min_{T} \left\{ \underbrace{\frac{1}{n}\sum_{i=1}^{n} \text{loss}(\widehat{f}_T(X_i), Y_i)}_{\text{log likelihood}} + \underbrace{\text{pen}(T)}_{\text{cost}} \right\}$$

$$\begin{aligned} \text{loss}(\widehat{f}_T(X_i), Y_i) &= (\widehat{f}_T(X_i) - Y_i)^2 && \text{regression} \\ &= \mathbf{1}_{\widehat{f}_T(X_i) \neq Y_i} && \text{classification} \end{aligned}$$

$\text{pen}(T) \propto |T|$ penalize trees with more leaves

Information Criteria - MDL

Penalize complex models based on their **information content**.

$$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \left\{ \widehat{R}_n(f) + C(f) \right\}$$

bits needed to describe *f*
(description length)

MDL (Minimum Description Length)

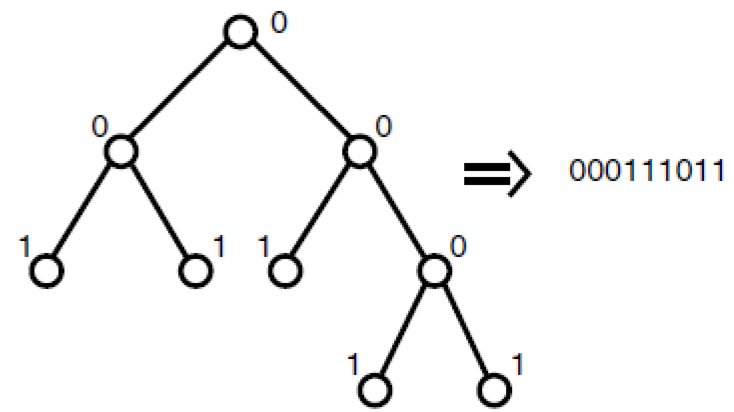
Example: Binary Decision trees $\mathcal{F}_k^T = \{\text{tree classifiers with } k \text{ leafs}\}$

prefix encode each element f of $\mathcal{F}_k^T$

$$C(f) = 3k - 1 \text{ bits}$$

k leaves => 2k – 1 nodes

2k – 1 bits to encode tree structure
+ k bits to encode label of each leaf (0/1)



5 leaves => 9 bits to encode structure

So far…

- What does a decision tree represent
- Given a decision tree, how do we assign label to a test point

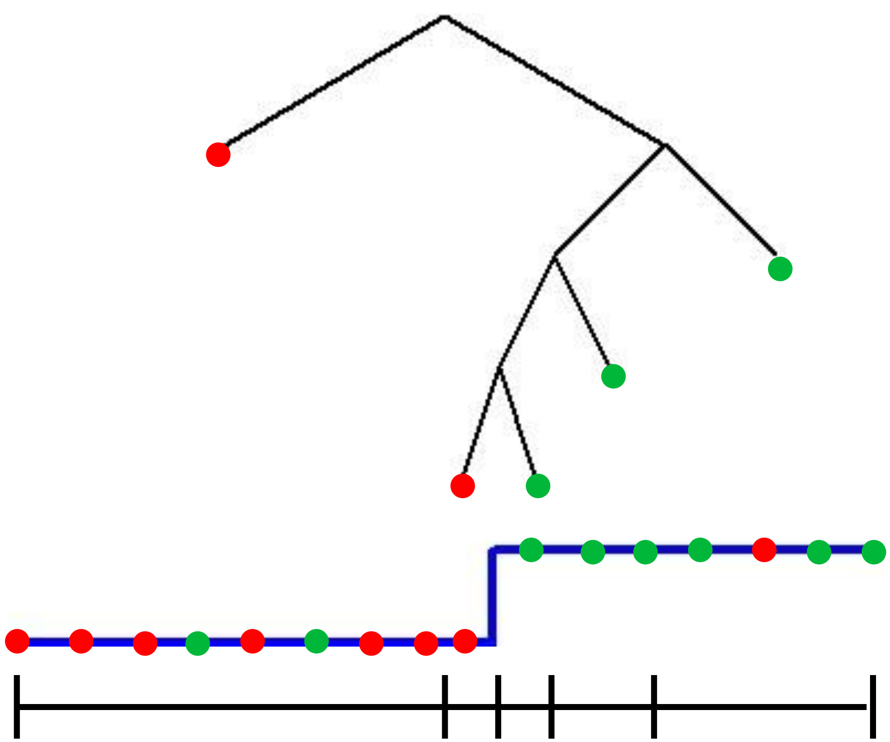
Now …

- How do we learn a decision tree from training data
- What is the decision on each leaf

How to assign label to each leaf

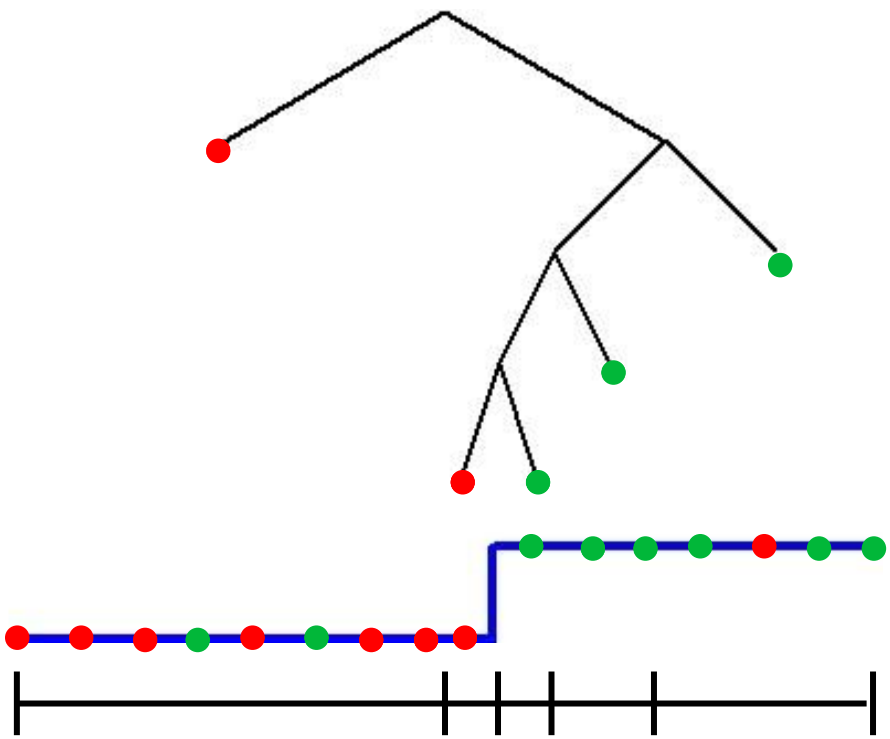
Classification – Majority vote

Regression – ?

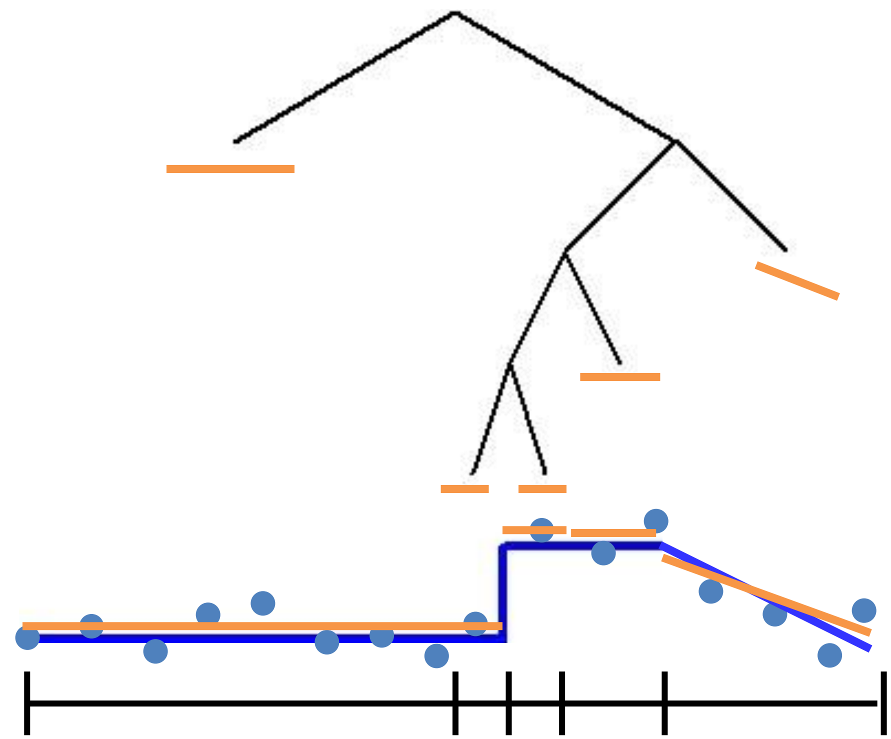


How to assign label to each leaf

Classification – Majority vote



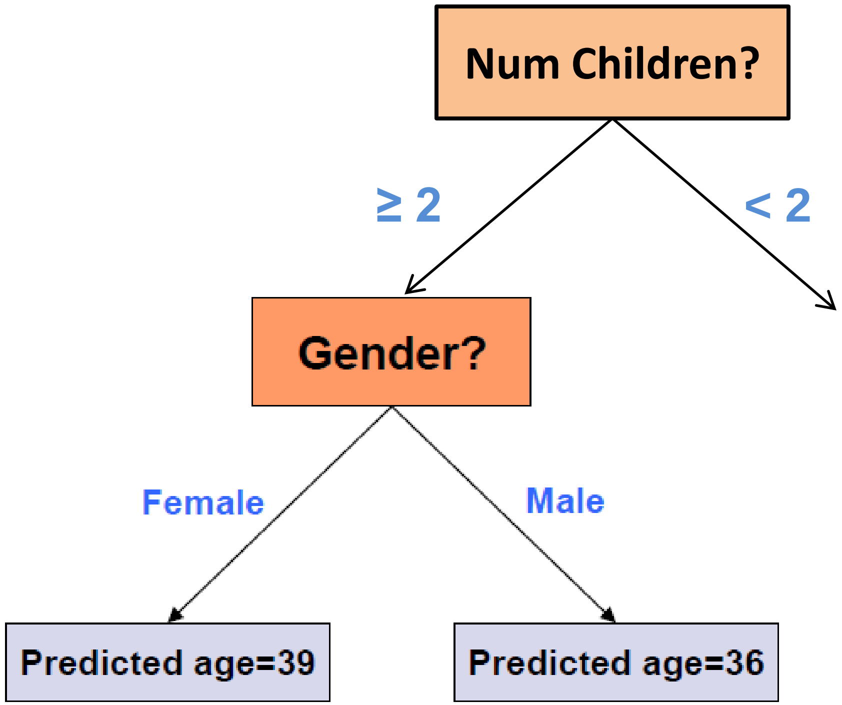
Regression – Constant/ Linear/Poly fit



Regression trees

$X^{(1)}$ $\ldots$ $X^{(p)}$ Y

Gender	Rich?	Num. Children	# travel per yr.	Age
F	No	2	5	38
M	No	0	2	25
M	Yes	1	0	72
:	:	:	:	:

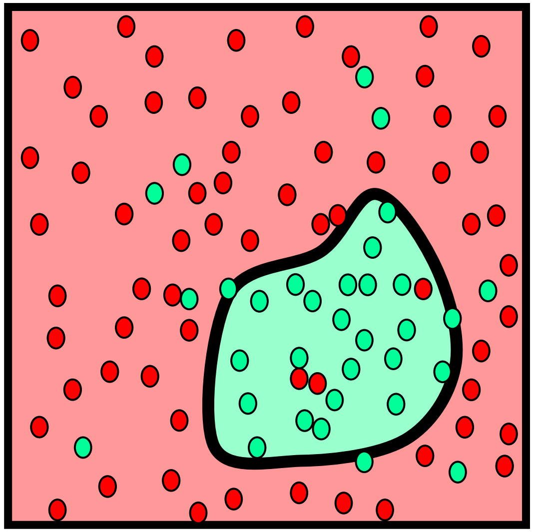


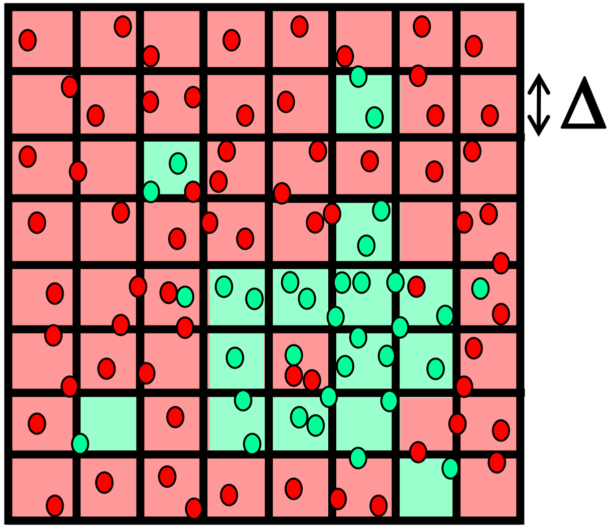
Average (fit a constant) using training data at the leaves

Connection between nearest neighbor/histogram classifiers and decision trees

Local prediction

Histogram, kernel density estimation, k-nearest neighbor classifier, kernel regression

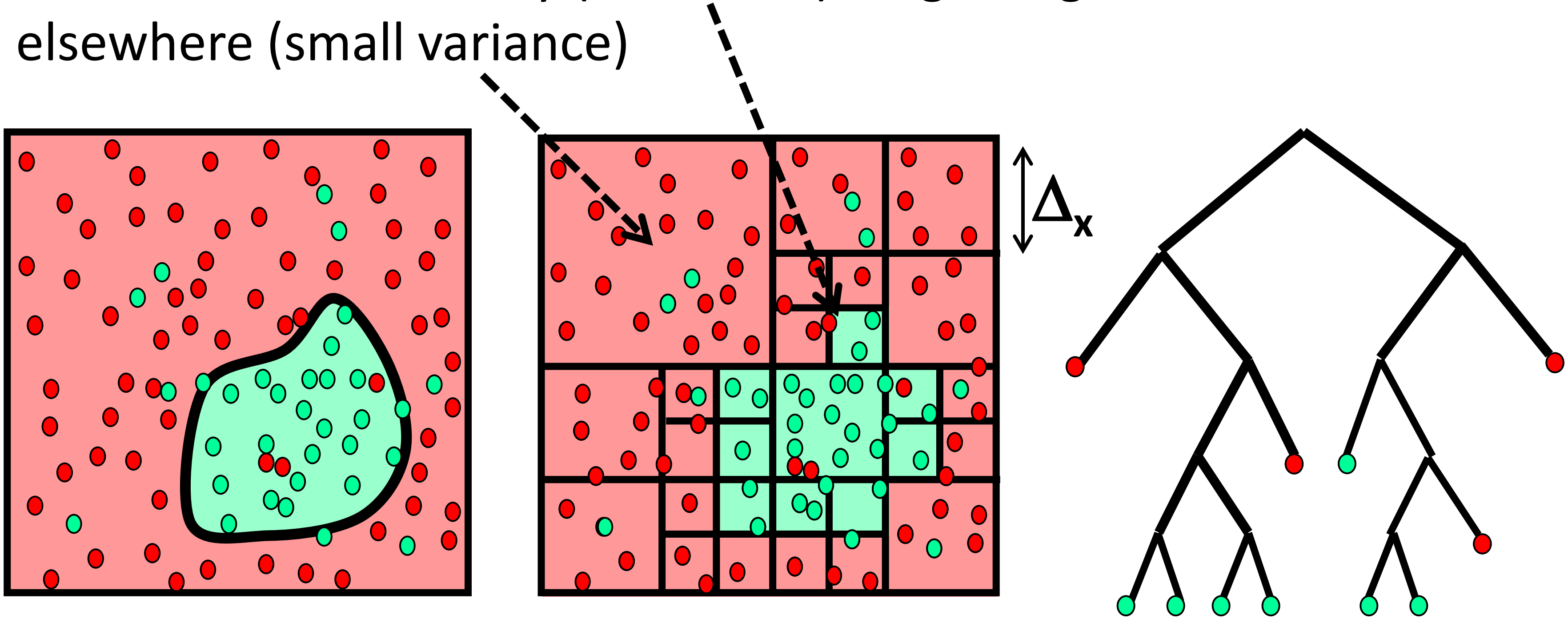




Histogram Classifier

Local Adaptive prediction

Let neighborhood size adapt to data – small neighborhoods near decision boundary (small bias), large neighborhoods elsewhere (small variance)

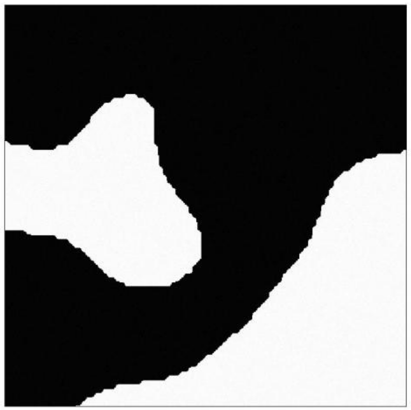


Majority vote
at each leaf

Decision Tree Classifier

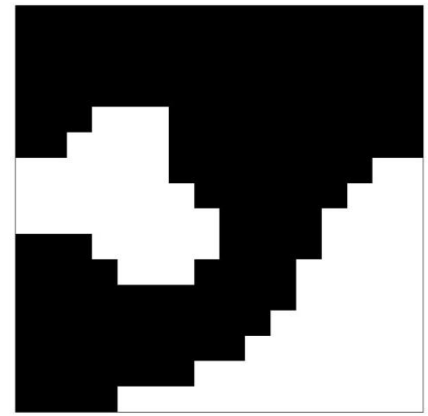
Histogram Classifier vs Decision Trees

Ideal classifier

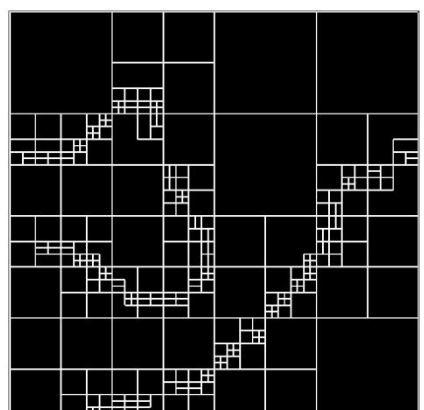


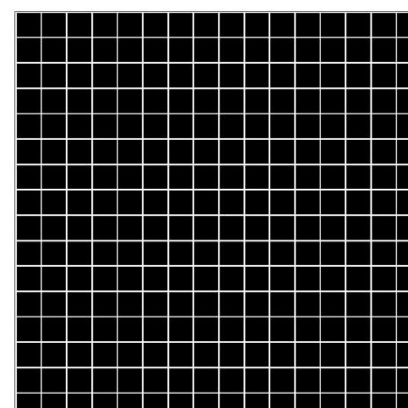
Decision tree

histogram

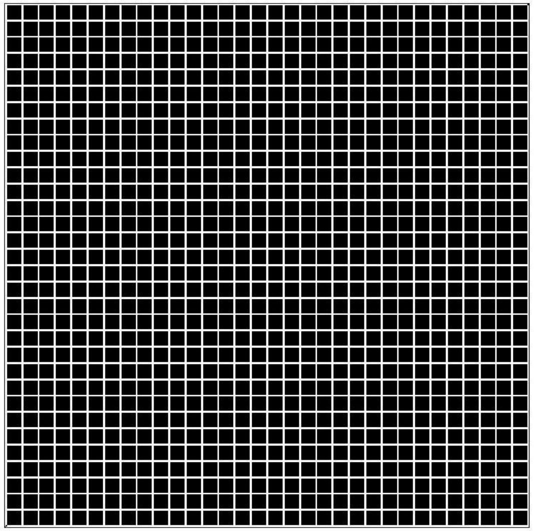


256 cells in
each partition

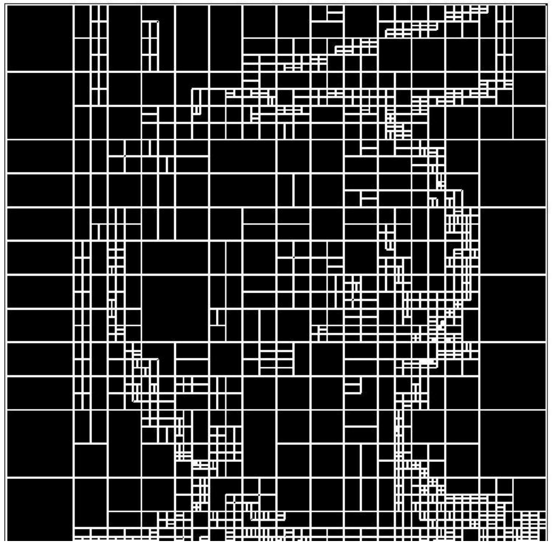




Application to Image Coding



1024 cells in
each partition



Application to Image Coding

JPEG 0.125 bpp
non-adaptive partitioning

JPEG 2000 0.125 bpp
adaptive partitioning

What you should know

- Decision trees are one of the most popular data mining tools
  - Simplicity of design
  - Interpretability
  - Ease of implementation
  - Good performance in practice (for small dimensions)
- Information gain to select attributes (ID3, C4.5,…)
- Can be used for classification, regression and density estimation too
- Decision trees will overfit!!!
  - Must use tricks to find “simple trees”, e.g.,
    - Pre-Pruning: Fixed depth/Fixed number of leaves
    - Post-Pruning: Chi-square test of independence
    - Complexity Penalized/MDL model selection