BLG 454E Learning from Data

FALL 2022-2023

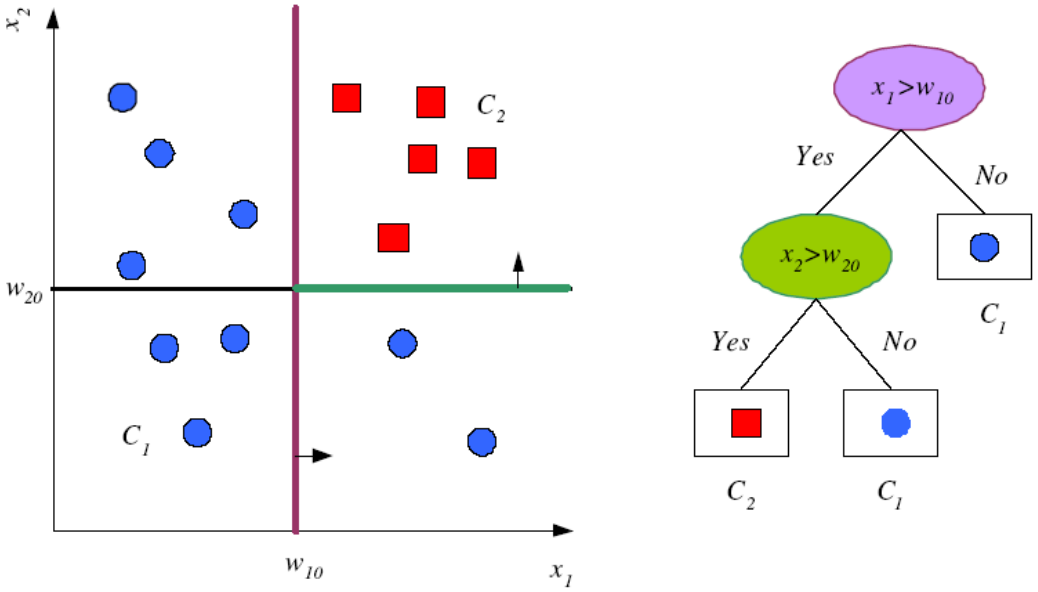
Assoc. Prof. Yusuf Yaslan

Trees

Lecture Notes from Alpaydın 2010 Introduction to Machine Learning 2e © The MIT Press AND
S. Theodoridis, Machine Learning: A Bayesian and Optimization Perspective, Elsevier 2nd Edition AND
P. Flach, Chapter 5 Trees, Machine Learning: Making Sense of Data 2013 AND
N. De Freitas, Machine Learning-Decision Trees, Random Forest, 2013 Youtube Lecture AND
A. Criminisi et al. Decision Forests for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning, 2011, Microsoft Technical Report

Trees/Decision Trees

- Each internal node is labelled with a feature
- Each edge is labelled with a literal
  - Set of literals: Split
- Leaves are
  - Classification: class labels
  - Regression: numeric, average, or local fit



Lecture Notes for E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

Trees/Decision Trees

- Univariate case
  - Single attribute x_1
  - In case of numeric, binary split $x_1 > m$
  - In case of discrete, n-way split
- Multivariate case
  - All attributes, x
- Learning is *greedy*
  - *Finding the best split (and feature) recursively*

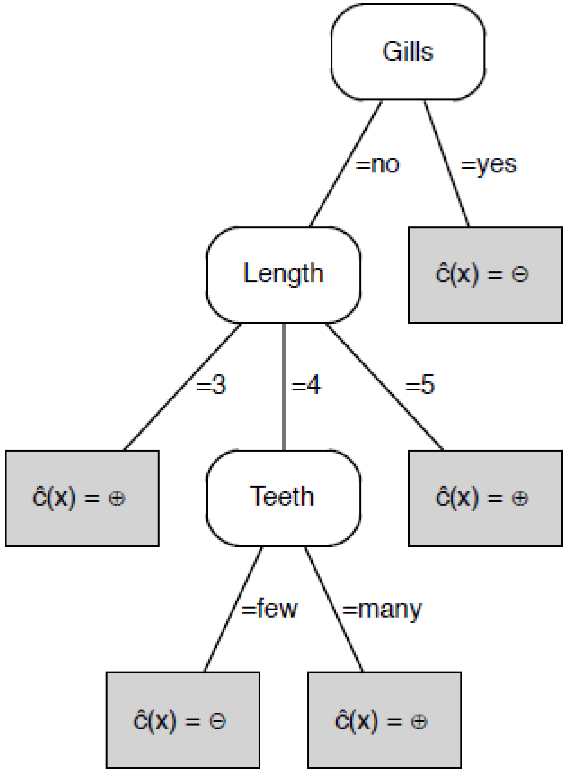


Figure from P. Flach, Chapter 5 Trees, Machine Learning: Making Sense of Data 2013

Remarks on Trees

- \+ Trees can naturally treat mixtures of numeric and categorical variables.
- \+ They scale well with large data sets.
- \+ They can treat missing variables in an effective way.
- \+ They are easily interpretable (expressive).
- \- Prediction performance may not be as good as other classifiers such as SVM or NNs
- \- Trees are sensitive to changes in training data – unstable
- **Bagging/Boosting** can reduce variance and improve generalization performance
- **Random forests** use bagging and often have very good prediction accuracy.

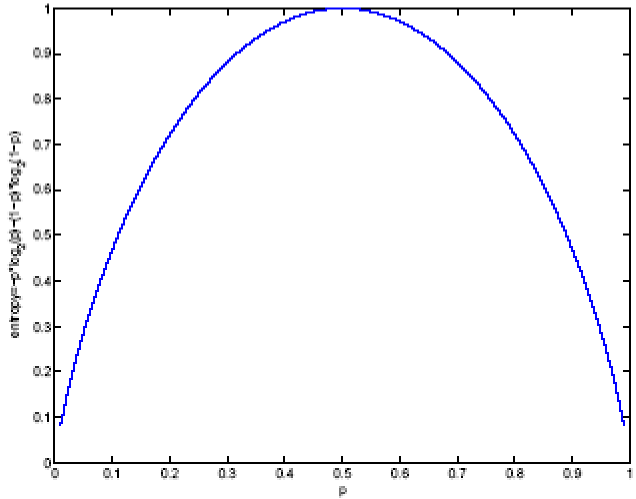
Classification Trees (ID3, CART, C4.5)

- For node *m*, N_m instances reach *m*, N^i_m belong to C_i

$$\hat{P}(C_i \mid \boldsymbol{x}, m) \equiv p^i_m = \frac{N^i_m}{N_m}$$

- Node *m* is pure if p^i_m is 0 or 1
- Measure of impurity is entropy

$$\mathcal{I}_m = -\sum_{i=1}^{K} p^i_m \log_2 p^i_m$$



Impurity

- Every split from X_t to X_{tL} and X_{tR} must generate class-homogeneous sets compared to X_t.
  - Easier to distinguish classes in the new subsets.
- Choose the feature that maximizes the decrease in node impurity
  - Higher **Information Gain**

$$I = H(\mathcal{S}) - \sum_{i \in \{1,2\}} \frac{|\mathcal{S}^i|}{|\mathcal{S}|} H(\mathcal{S}^i)$$

- Besides *entropy*
  - *Gini Index:*

$$I_m = -\sum_{i=1}^{K} p_m^i (1 - p_m^i)$$

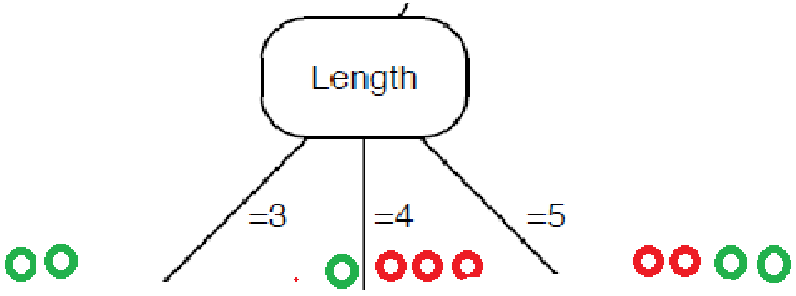
Suppose we have the following five positive examples (the first three are the same as in Example 4.1):

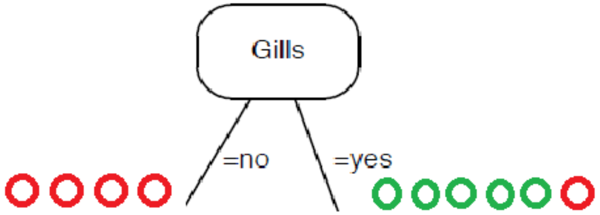
$$
\begin{aligned}
&\text{p1: } \mathsf{Length} = 3 \;\wedge\; \mathsf{Gills} = \mathsf{no} \;\wedge\; \mathsf{Beak} = \mathsf{yes} \;\wedge\; \mathsf{Teeth} = \mathsf{many}\\
&\text{p2: } \mathsf{Length} = 4 \;\wedge\; \mathsf{Gills} = \mathsf{no} \;\wedge\; \mathsf{Beak} = \mathsf{yes} \;\wedge\; \mathsf{Teeth} = \mathsf{many}\\
&\text{p3: } \mathsf{Length} = 3 \;\wedge\; \mathsf{Gills} = \mathsf{no} \;\wedge\; \mathsf{Beak} = \mathsf{yes} \;\wedge\; \mathsf{Teeth} = \mathsf{few}\\
&\text{p4: } \mathsf{Length} = 5 \;\wedge\; \mathsf{Gills} = \mathsf{no} \;\wedge\; \mathsf{Beak} = \mathsf{yes} \;\wedge\; \mathsf{Teeth} = \mathsf{many}\\
&\text{p5: } \mathsf{Length} = 5 \;\wedge\; \mathsf{Gills} = \mathsf{no} \;\wedge\; \mathsf{Beak} = \mathsf{yes} \;\wedge\; \mathsf{Teeth} = \mathsf{few}
\end{aligned}
$$

and the following negatives (the first one is the same as in Example 4.2):

$$
\begin{aligned}
&\text{n1: } \mathsf{Length} = 5 \;\wedge\; \mathsf{Gills} = \mathsf{yes} \;\wedge\; \mathsf{Beak} = \mathsf{yes} \;\wedge\; \mathsf{Teeth} = \mathsf{many}\\
&\text{n2: } \mathsf{Length} = 4 \;\wedge\; \mathsf{Gills} = \mathsf{yes} \;\wedge\; \mathsf{Beak} = \mathsf{yes} \;\wedge\; \mathsf{Teeth} = \mathsf{many}\\
&\text{n3: } \mathsf{Length} = 5 \;\wedge\; \mathsf{Gills} = \mathsf{yes} \;\wedge\; \mathsf{Beak} = \mathsf{no} \;\wedge\; \mathsf{Teeth} = \mathsf{many}\\
&\text{n4: } \mathsf{Length} = 4 \;\wedge\; \mathsf{Gills} = \mathsf{yes} \;\wedge\; \mathsf{Beak} = \mathsf{no} \;\wedge\; \mathsf{Teeth} = \mathsf{many}\\
&\text{n5: } \mathsf{Length} = 4 \;\wedge\; \mathsf{Gills} = \mathsf{no} \;\wedge\; \mathsf{Beak} = \mathsf{yes} \;\wedge\; \mathsf{Teeth} = \mathsf{few}
\end{aligned}
$$

Example





Similar calculations for Beak and Teeth would give 0.76 and 0.97 respectively.

Pick 'Gills' as the first feature!

Length = [3, 4, 5]	[2+, 0−][1+, 3−][2+, 2−]
Gills = [yes, no]	[0+, 4−][5+, 1−]
Beak = [yes, no]	[5+, 3−][0+, 2−]
Teeth = [many, few]	[3+, 4−][2+, 1−]

Impurity of the root is 5+,5- so it is 1.
We will find the feature that maximizes the decrease in impurity.

Impurity of using 'Length' is

-2/2 * log1 – 0/2*log0/2 = 0
-1/4 * log(1/4) – ¾*log(3/4) = 0.81
-2/4 * log(2/4) – 2/4*log(2/4) = 1
Total entropy is weighted average:
2/10 * 0 + 4/10 * 0.81 + 4/10 * 1 = 0.72

Impurity of 'Gills' is

-0/4 * log 0 – 4/4 * log1 = 0
-5/6*log(5/6) – 1/6*log(1/6) =
Total entropy is
4/10*0 + 6/10 * (-5/6*log(5/6) – 1/6*log(1/6)) = 0.39

Example cont'd

- After the first feature is selected, the right side of the tree reaches the final leaf node. NEGATIVE
- Perform similar operations to decide on the second feature, then third, and forth until you reach the class labels in the leaf nodes.

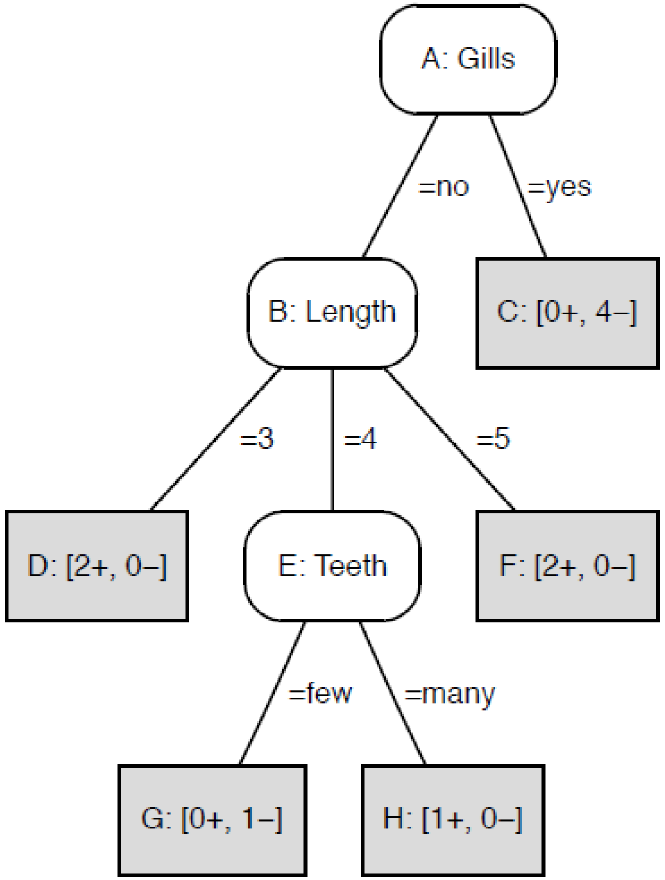


Figure from P. Flach, Chapter 5 Trees, Machine Learning: Making Sense of Data 2013

Another Example

- Let's compare *Pat* (Patrons) and *Type*
  - I(Pat) = 0.541, I(Type) = 0

Example	Input Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$

Figure 18.3 Examples for the restaurant domain.

Example from S. Russell and P. Norvig, Chapter 18, Artificial Intelligence: A Modern Approach, 3rd Ed. 2010

Regression Trees

- Error at node m:

$$b_m(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in \mathcal{X}_m : \boldsymbol{x} \text{ reaches node } m \\ 0 & \text{otherwise} \end{cases}$$

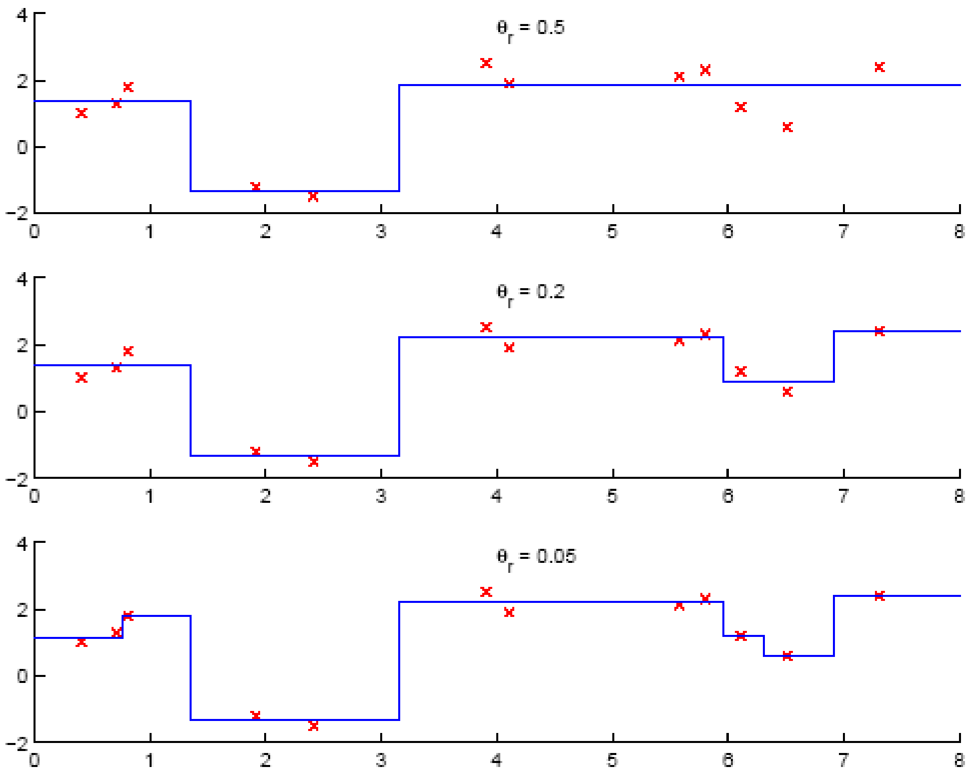
$$E_m = \frac{1}{N_m} \sum_t \left(r^t - g_m\right)^2 b_m\left(\boldsymbol{x}^t\right) \qquad g_m = \frac{\sum_t b_m\left(\boldsymbol{x}^t\right) r^t}{\sum_t b_m\left(\boldsymbol{x}^t\right)}$$

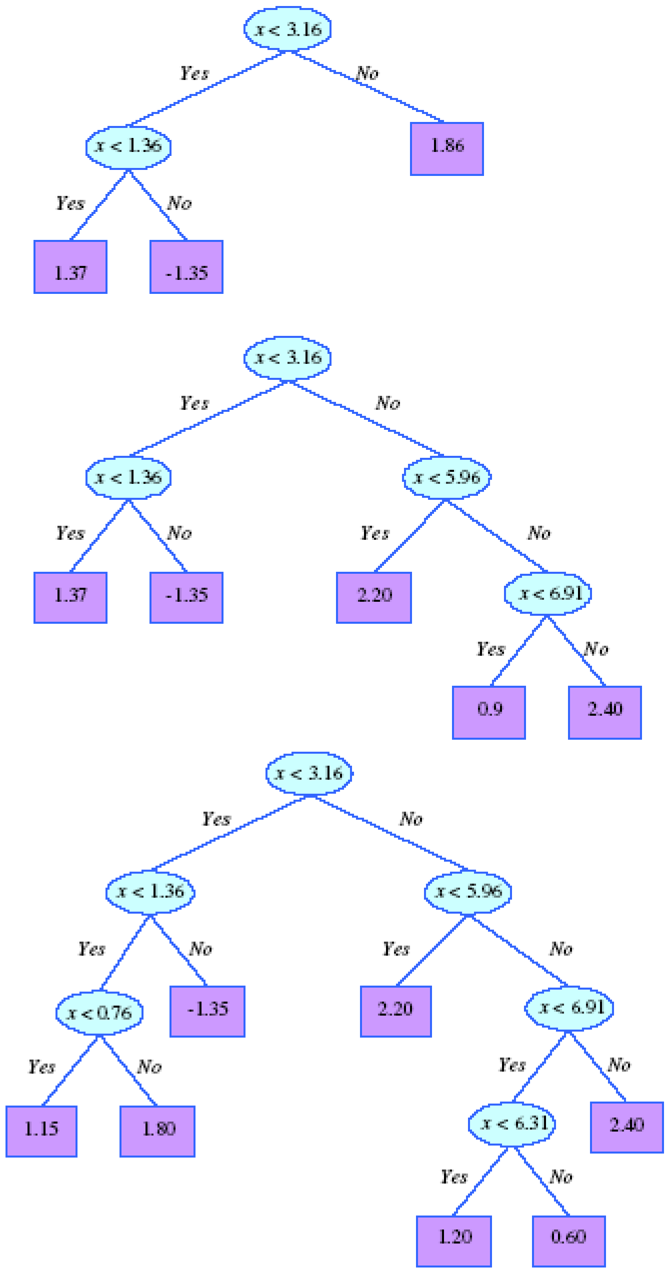
- After splitting:

$$b_{mj}(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in \mathcal{X}_{mj} : \boldsymbol{x} \text{ reaches node } m \text{ and branch } j \\ 0 & \text{otherwise} \end{cases}$$

$$E'_m = \frac{1}{N_m} \sum_j \sum_t \left(r^t - g_{mj}\right)^2 b_{mj}\left(\boldsymbol{x}^t\right) \qquad g_{mj} = \frac{\sum_t b_{mj}\left(\boldsymbol{x}^t\right) r^t}{\sum_t b_{mj}\left(\boldsymbol{x}^t\right)}$$

Model Selection in Trees:



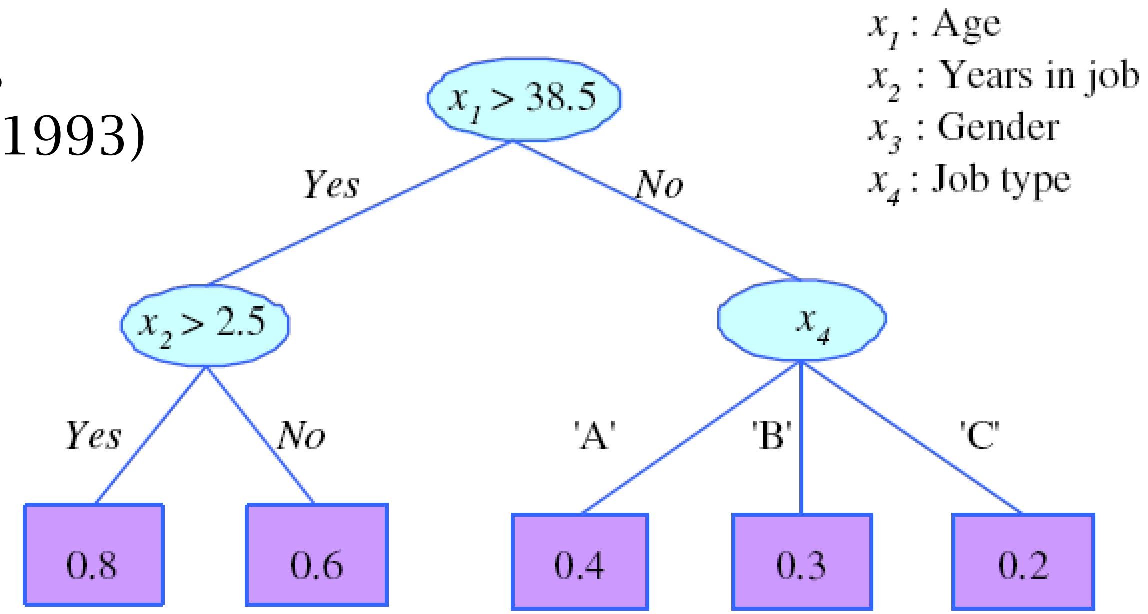


Pruning Trees

- Remove subtrees for better generalization (decrease variance)
  - Prepruning: Early stopping
  - Postpruning: Grow the whole tree then prune subtrees which overfit on the pruning set
- Prepruning is faster, postpruning is more accurate (requires a separate pruning set)

Rule Extraction from Trees

C4.5Rules
(Quinlan, 1993)



R1: IF (age>38.5) AND (years-in-job>2.5) THEN y =0.8
R2: IF (age>38.5) AND (years-in-job≤2.5) THEN y =0.6
R3: IF (age≤38.5) AND (job-type='A') THEN y =0.4
R4: IF (age≤38.5) AND (job-type='B') THEN y =0.3
R5: IF (age≤38.5) AND (job-type='C') THEN y =0.2

Random Trees
Random Forest

- How to build a random tree?
  - Randomly select two features, pick the best one in terms of impurity
  - Continue to grow the tree until you reach the leaf nodes.
- How to avoid the impact of randomization?
  - Execute this many (T) times:
    - Pick a sample Z of size N from the training data (**Bagging**)
    - Build a random tree by recursively following the steps
      - Select m random features, pick the best split, split the node
    - Output the ensemble
  - Final prediction: Average over all trees

$$p(c|\mathbf{v}) = \frac{1}{T}\sum_{t=1}^{T} p_t(c|\mathbf{v})$$

- The first example of **ensemble of classifiers** in this course