DS 4400

Machine Learning and Data Mining I

Alina Oprea
Associate Professor, CCIS
Northeastern University

February 12 2019

Logistics

- TA office hours moved to 5:45pm today
- HW3 will be out today or tomorrow
  - Due on Friday, February 22
- Project proposal due on Tuesday 02/26
  - 1 page description of your project, including problem statement, dataset, and ML algorithms
- Week of February 25
  - Lecture on 02/26 taught by Lisa Friedland
  - Lecture on 02/28 canceled

Review

- Metrics for evaluating classifiers
  - Accuracy, error, precision, recall, F1 score
  - AUC (area under the ROC curve) measures performance of classifier for different thresholds
- Feature selection methods
  - Filters decide on each feature individually
  - Wrappers select a subset of features by search strategy (fixing model and evaluating with cross-validation)
  - Embedded methods (e.g., regularization) are part of training
- Decision trees are interpretable, non-linear models
  - Greedy algorithm to train Decision Trees
  - Works on categorical and numerical data
  - Node splitting done by highest Information Gain

Outline

- Decision trees
- How to split nodes
  - Definitions of entropy, conditional entropy, information gain
- ID3 algorithm
  - Training
  - Pruning
  - Interpretability
- Lab
- Ensemble learning

Sample Dataset

- Columns denote features X_i
- Rows denote labeled instances $\langle x^{(i)}, y^{(i)} \rangle$
- Class label denotes whether a tennis game was played

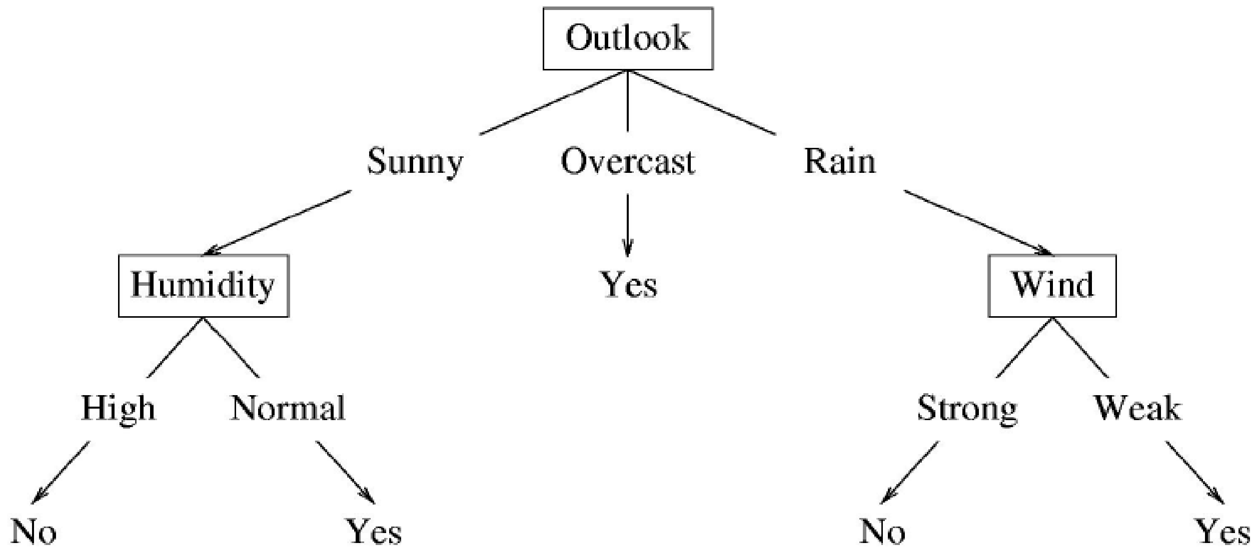
Predictors				Response
Outlook	**Temperature**	**Humidity**	**Wind**	**Class**
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

$\langle x^{(i)}, y^{(i)} \rangle$

Categorical data

Decision Tree

- A possible decision tree for the data:

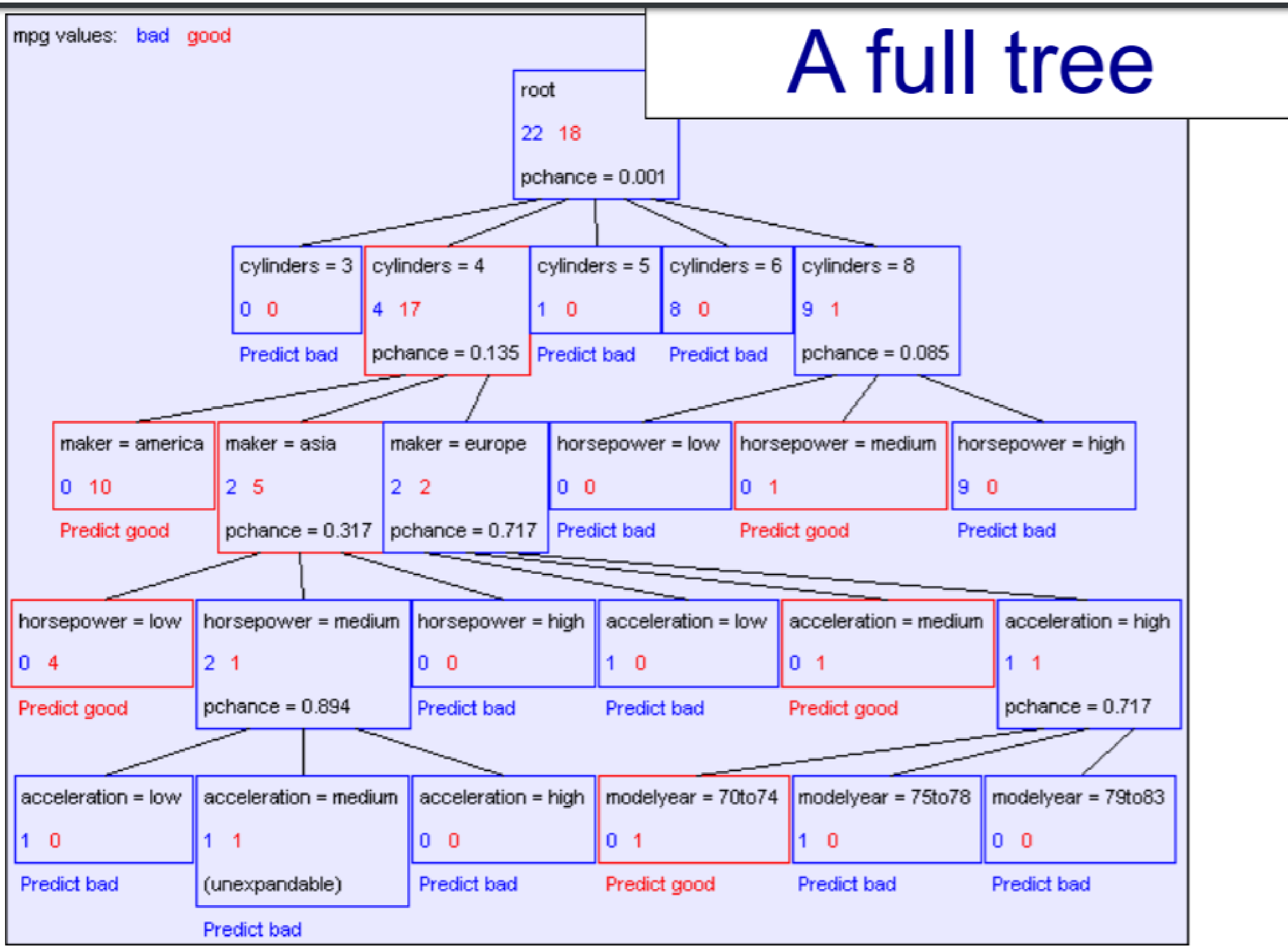


- What prediction would we make for
  <outlook=sunny, temperature=hot, humidity=high, wind=weak> ?

Learning Decision Trees

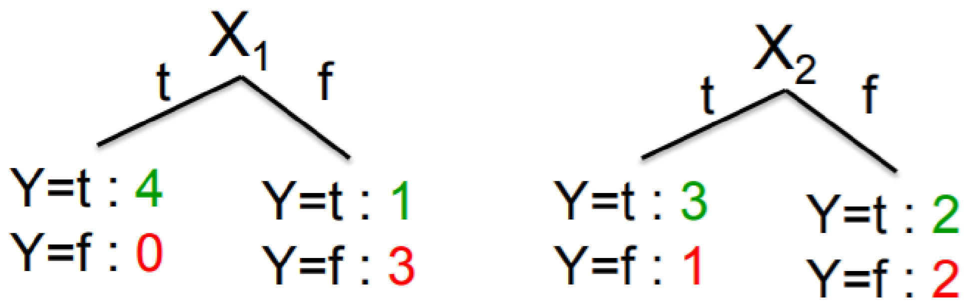
- Learning the simplest (smallest) decision tree is an NP-complete problem [Hyafil & Rivest '76]
- Resort to a greedy heuristic:
  - Start from empty decision tree
  - Split on **next best attribute (feature)**
  - Recurse

Full Tree


mpg values: bad good
A full tree
root
22 18
pchance = 0.001
cylinders = 3
0 0
Predict bad
cylinders = 4
4 17
pchance = 0.135
cylinders = 5
1 0
Predict bad
cylinders = 6
8 0
Predict bad
cylinders = 8
9 1
pchance = 0.085
maker = america
0 10
Predict good
maker = asia
2 5
pchance = 0.317
maker = europe
2 2
pchance = 0.717
horsepower = low
0 0
Predict bad
horsepower = medium
0 1
Predict good
horsepower = high
9 0
Predict bad
horsepower = low
0 4
Predict good
horsepower = medium
2 1
pchance = 0.894
horsepower = high
0 0
Predict bad
acceleration = low
1 0
Predict bad
acceleration = medium
0 1
Predict good
acceleration = high
1 1
pchance = 0.717
acceleration = low
1 0
Predict bad
acceleration = medium
1 1
(unexpandable)
Predict bad
acceleration = high
0 0
Predict bad
modelyear = 70to74
0 1
Predict good
modelyear = 75to78
1 0
Predict bad
modelyear = 79to83
0 0
Predict bad

Splitting

Would we prefer to split on X_1 or X_2?



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

Idea: use counts at leaves to define probability distributions, so we can measure uncertainty!

Use entropy-based measure (Information Gain)

Transmitting Bits

You are watching a set of independent random samples of X

You see that X has four possible values

P(X=A) = 1/4	P(X=B) = 1/4	P(X=C) = 1/4	P(X=D) = 1/4

So you might see: BAACBADCDADDDA…

You transmit data over a binary serial link. You can encode each reading with two bits (e.g. A = 00, B = 01, C = 10, D = 11)

0100001001001110110011111100…

Use Fewer Bits

Someone tells you that the probabilities are not equal

P(X=A) = 1/2	P(X=B) = 1/4	P(X=C) = 1/8	P(X=D) = 1/8

It's possible…

…to invent a coding for your transmission that only uses 1.75 bits on average per symbol. How?

Use Fewer Bits

Someone tells you that the probabilities are not equal

P(X=A) = 1/2	P(X=B) = 1/4	P(X=C) = 1/8	P(X=D) = 1/8

It's possible…

…to invent a coding for your transmission that only uses 1.75 bits on average per symbol. How?

A	0
B	10
C	110
D	111

(This is just one of several ways)

General case

Suppose X can have one of *m* values… $V_1, V_2, \ldots V_m$

$P(X=V_1) = p_1$	$P(X=V_2) = p_2$	….	$P(X=V_m) = p_m$

What's the smallest possible number of bits, on average, per symbol, needed to transmit a stream of symbols drawn from X's distribution? It's

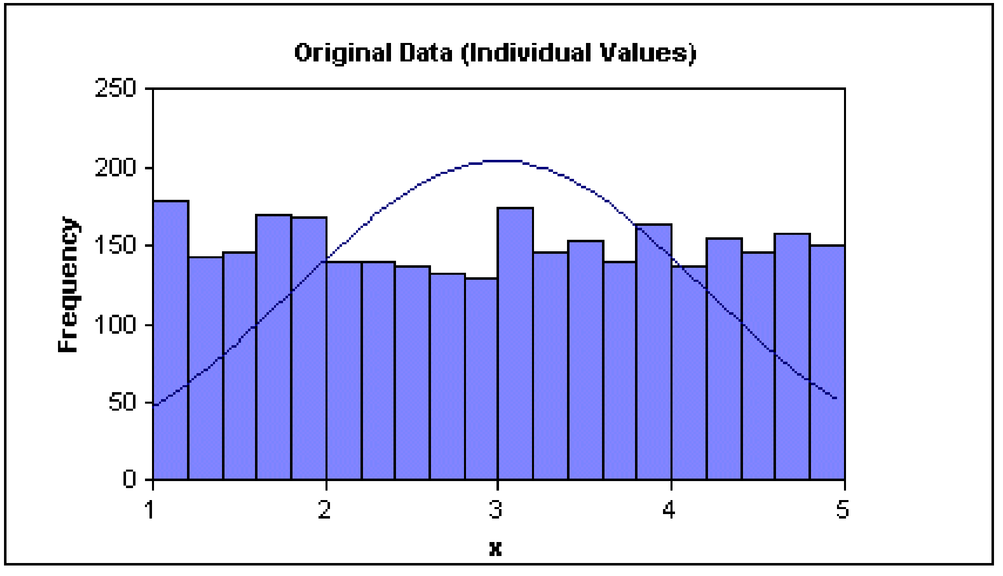
$$H(X) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \ldots - p_m \log_2 p_m$$
$$= -\sum_{j=1}^{m} p_j \log_2 p_j$$

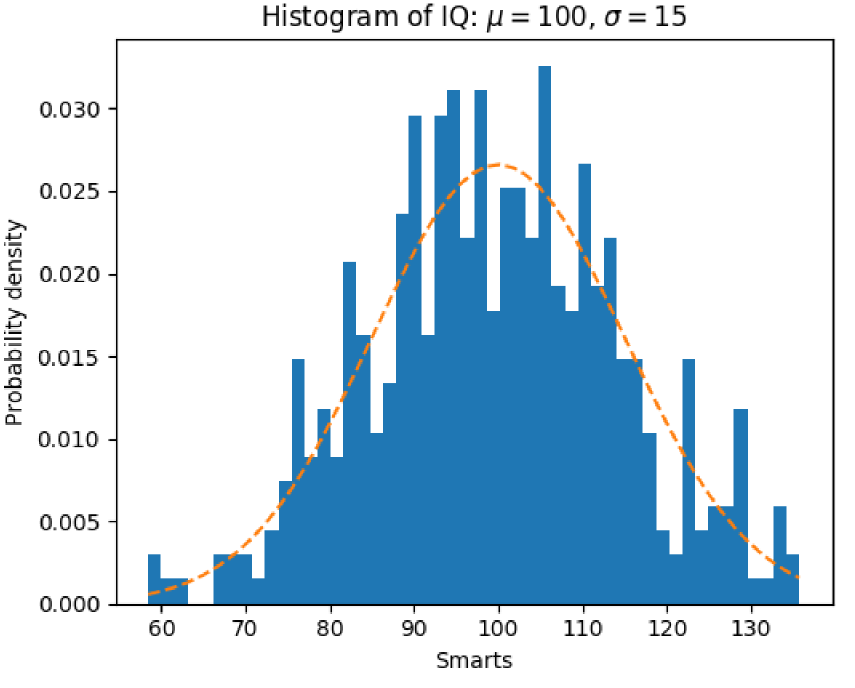
H(X) = The entropy of X

- "High Entropy" means X is from a uniform (boring) distribution
- "Low Entropy" means X is from varied (peaks and valleys) distribution

High/Low Entropy

Which distribution has high entropy?





High

Low

Conditional Entropy

Suppose I'm trying to predict output Y and I have input X

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Let's assume this reflects the true probabilities

E.G. From this data we estimate

- *P(LikeG = Yes) = 0.5*
- *P(Major = Math & LikeG = No) = 0.25*
- *P(Major = Math) = 0.5*
- *P(LikeG = Yes | Major = History) = 0*

Note:

- *H(X) = 1.5*
- *H(Y) = 1*

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Specific Conditional Entropy:

$H(Y|X=v)$ = **The entropy of** Y **among only those records in which** X **has value** v

Example:

- $H(Y|X=Math) = 1$
- $H(Y|X=History) = 0$
- $H(Y|X=CS) = 0$

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Conditional Entropy:

$H(Y|X)$ = The average specific conditional entropy of Y

= if you choose a record at random what will be the conditional entropy of Y, conditioned on that row's value of X

= Expected number of bits to transmit Y if both sides will know the value of X

$= \Sigma_j \, Prob(X=v_j) \; H(Y \mid X = v_j)$

Conditional Entropy

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Conditional Entropy:

$H(Y|X)$ = The average conditional entropy of Y

$= \Sigma_j Prob(X=v_j)\ H(Y \mid X = v_j)$

Example:

v_j	$Prob(X=v_j)$	$H(Y \mid X = v_j)$
Math	0.5	1
History	0.25	0
CS	0.25	0

$H(Y|X) = 0.5 * 1 + 0.25 * 0 + 0.25 * 0 = 0.5$

Information Gain

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Definition of Information Gain:

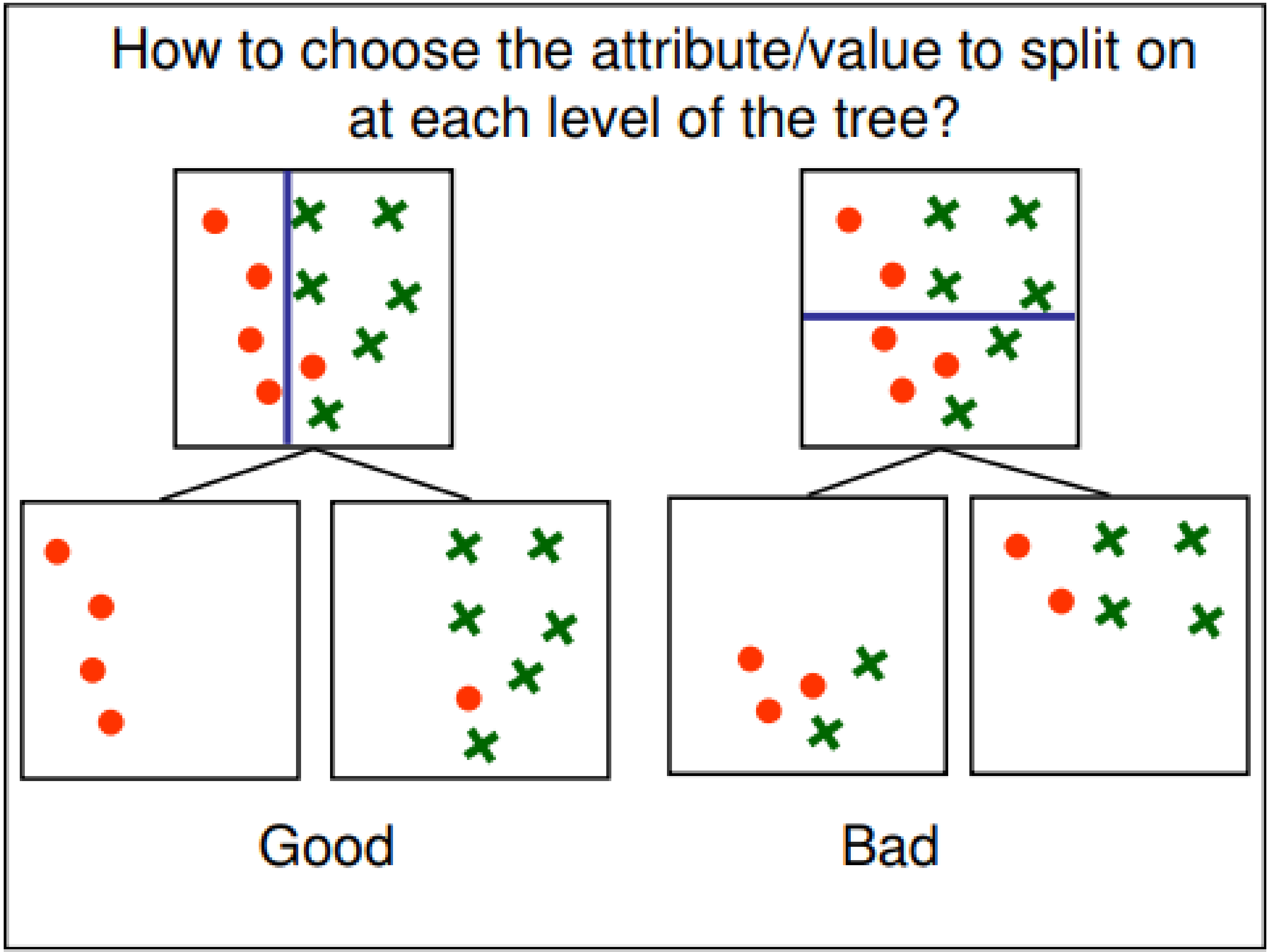
IG(Y|X) = **I must transmit** *Y*. **How many bits on average would it save me if both ends of the line knew** *X***?**

$IG(Y|X) = H(Y) - H(Y|X)$

Example:

- **H(Y) = 1**
- **H(Y|X) = 0.5**
- **Thus IG(Y|X) = 1 – 0.5 = 0.5**

Example

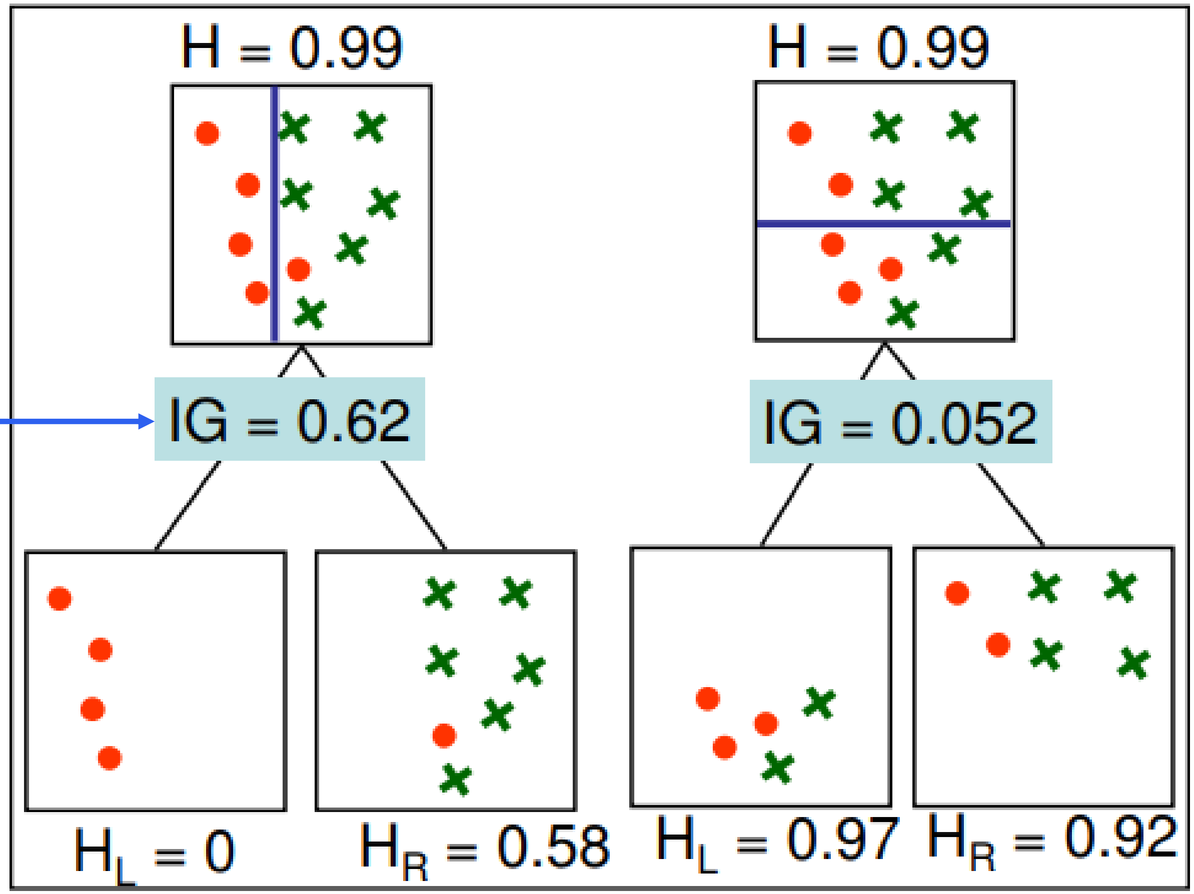


How to choose the attribute/value to split on
at each level of the tree?
Good
Bad

Example Information Gain

Max Information Gain

Pure node



Learning Decision Trees

- Start from empty decision tree
- Split on **next best attribute (feature)**
  - Use, for example, information gain to select attribute:

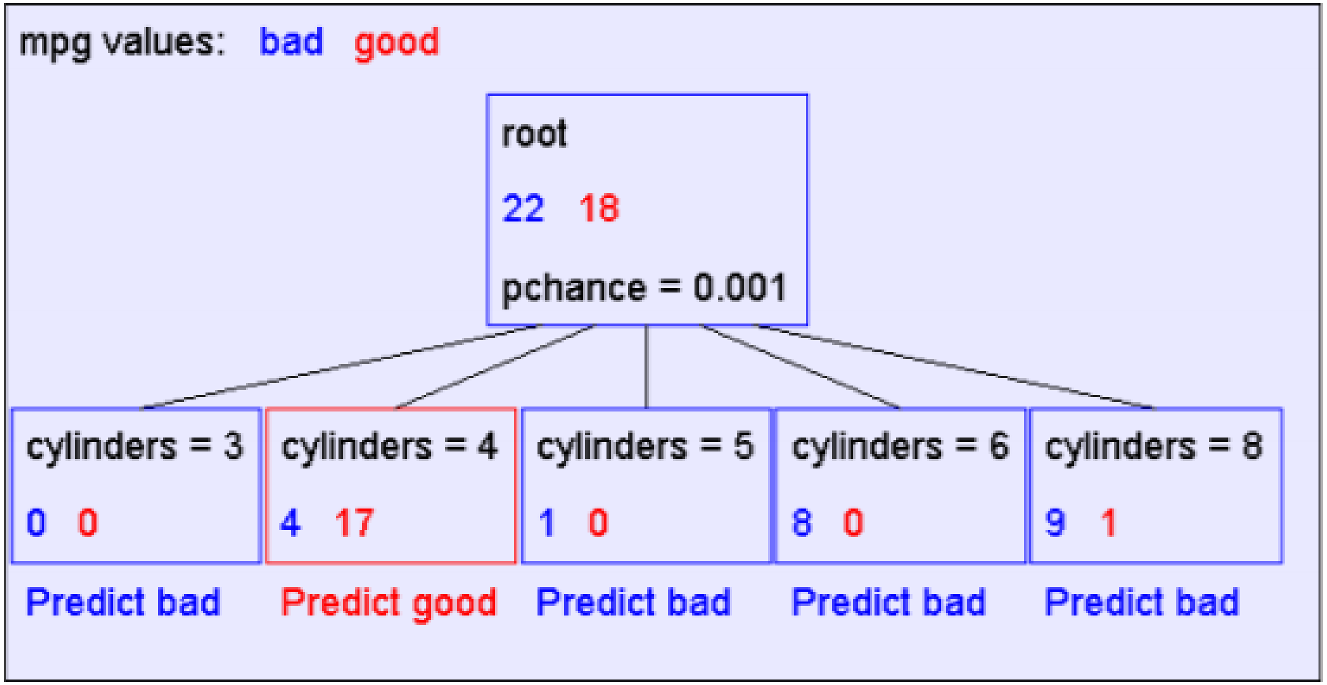
$$\arg\max_i IG(X_i) = \arg\max_i H(Y) - H(Y \mid X_i)$$

- Recurse

ID3 algorithm uses Information Gain

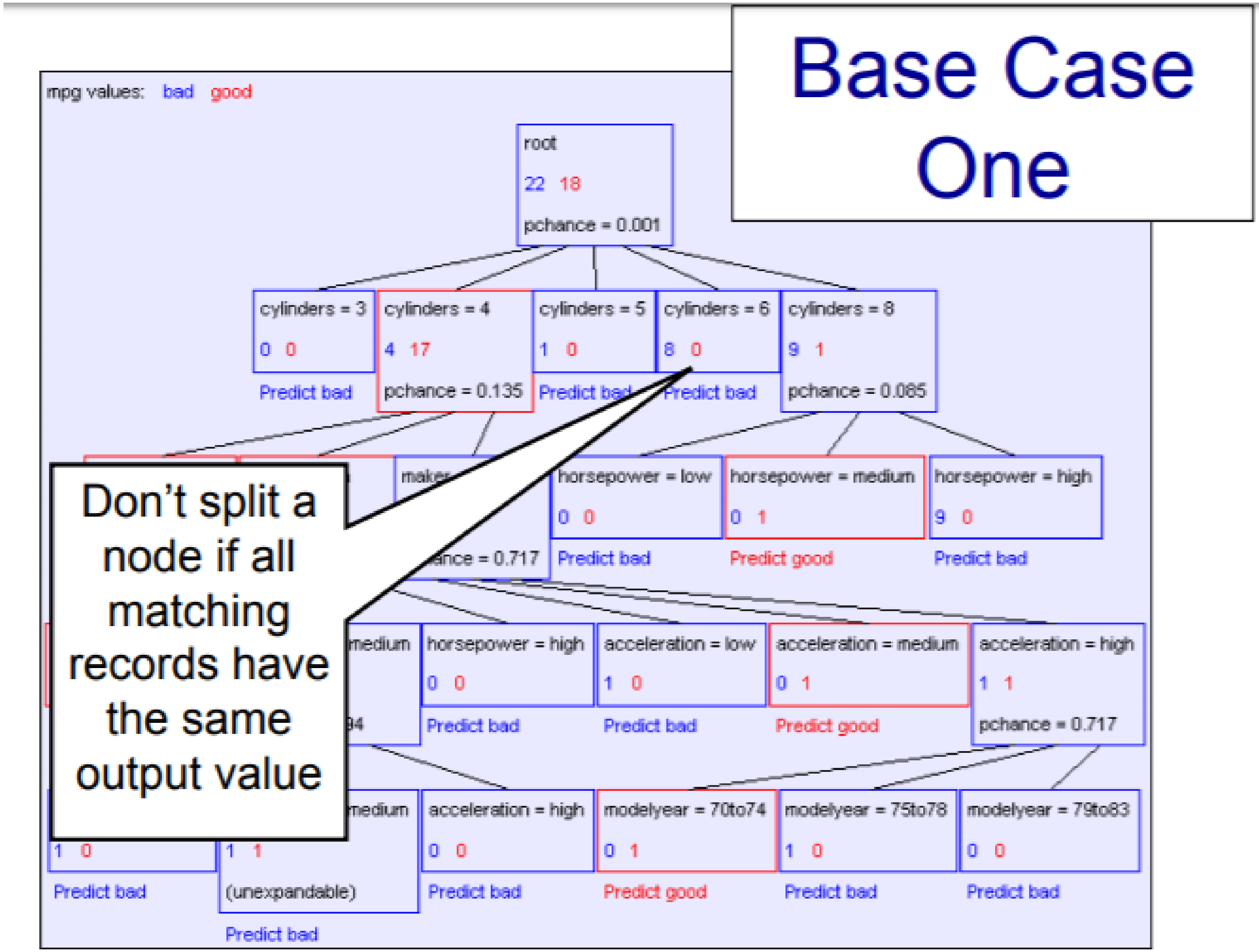
Information Gain reduces uncertainty on Y

When to stop?



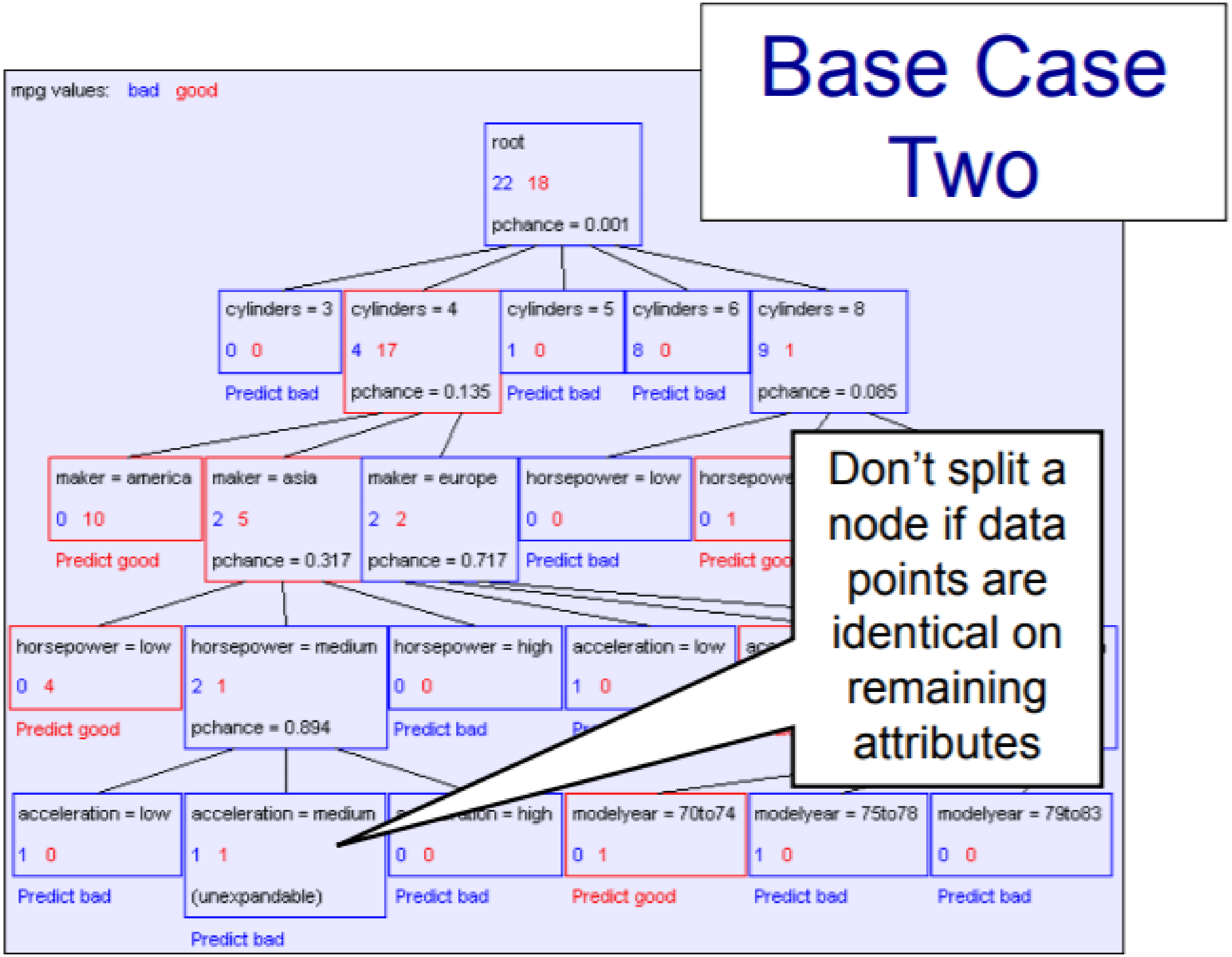
First split looks good! But, when do we stop?

Case 1

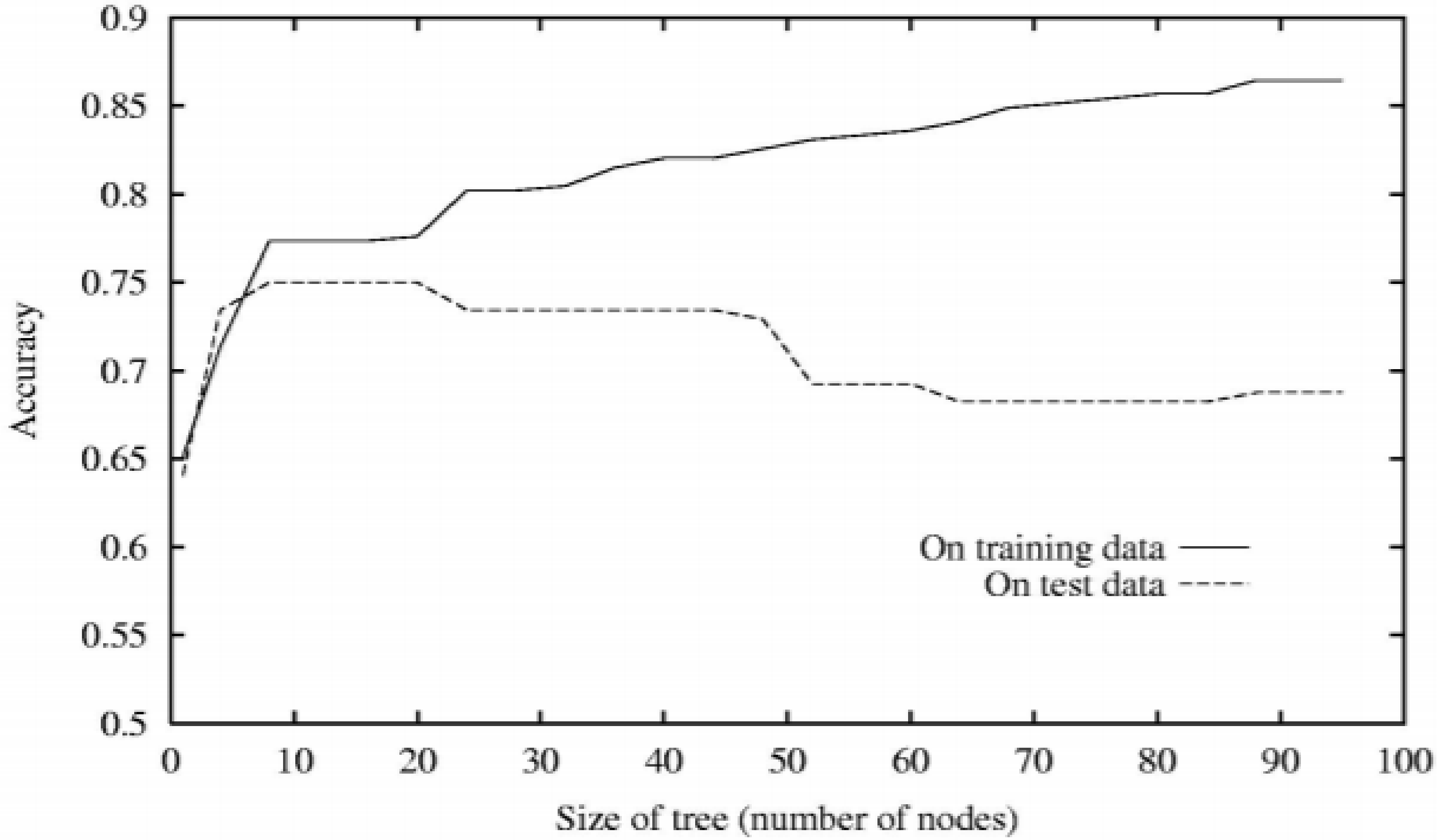


Base Case One
Don't split a node if all matching records have the same output value
mpg values: bad good
root
22 18
pchance = 0.001
cylinders = 3
0 0
Predict bad
cylinders = 4
4 17
pchance = 0.135
cylinders = 5
1 0
Predict bad
cylinders = 6
8 0
Predict bad
cylinders = 8
9 1
pchance = 0.085
horsepower = low
0 0
Predict bad
horsepower = medium
0 1
Predict good
horsepower = high
9 0
Predict bad
horsepower = high
0 0
Predict bad
acceleration = low
1 0
Predict bad
acceleration = medium
0 1
Predict good
acceleration = high
1 1
pchance = 0.717
acceleration = high
0 0
Predict bad
modelyear = 70to74
0 1
Predict good
modelyear = 75to78
1 0
Predict bad
modelyear = 79to83
0 0
Predict bad
1 0
Predict bad
1 1
(unexpandable)
Predict bad

Case 2


mpg values: bad good
root
22 18
pchance = 0.001
cylinders = 3
0 0
Predict bad
cylinders = 4
4 17
pchance = 0.135
cylinders = 5
1 0
Predict bad
cylinders = 6
8 0
Predict bad
cylinders = 8
9 1
pchance = 0.085
maker = america
0 10
Predict good
maker = asia
2 5
pchance = 0.317
maker = europe
2 2
pchance = 0.717
horsepower = low
0 0
Predict bad
0 1
horsepower = low
0 4
Predict good
horsepower = medium
2 1
pchance = 0.894
horsepower = high
0 0
Predict bad
acceleration = low
1 0
acceleration = low
1 0
Predict bad
acceleration = medium
1 1
(unexpandable)
Predict bad
0 0
Predict bad
modelyear = 70to74
0 1
Predict good
modelyear = 75to78
1 0
Predict bad
modelyear = 79to83
0 0
Predict bad
Base Case Two
Don't split a node if data points are identical on remaining attributes

Overfitting



Solutions against Overfitting

- Standard decision trees have no learning bias
  - Training set error is always zero!
    - (If there is no label noise)
  - Lots of variance
  - Must introduce some bias towards simpler trees
- Many strategies for picking simpler trees
  - Fixed depth
  - Minimum number of samples per leaf
- Pruning

Pruning Decision Trees

Split data into *training* and *validation* sets

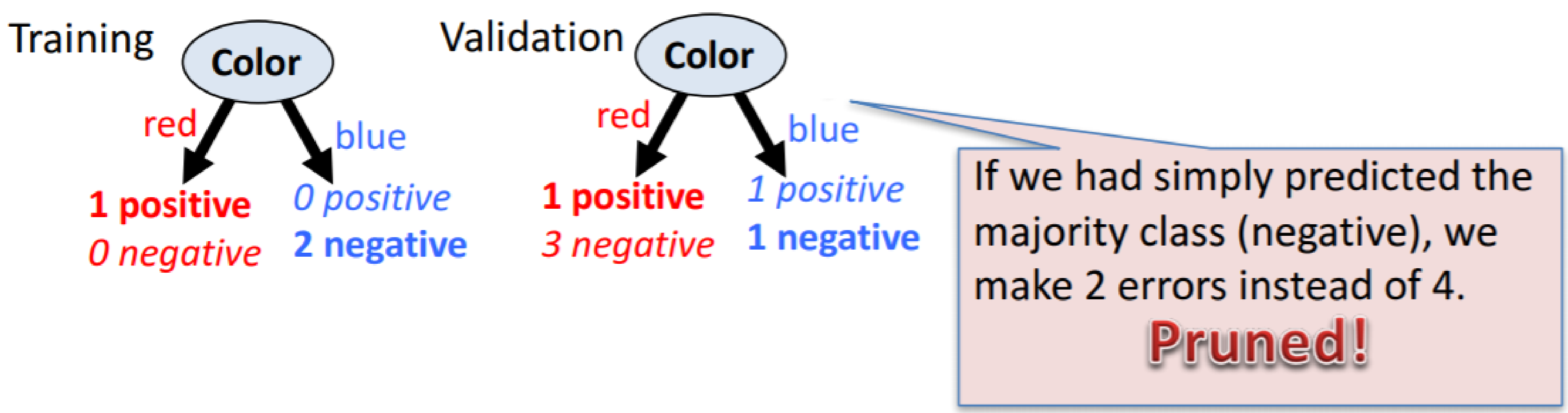
Grow tree based on *training set*

Do until further pruning is harmful:

1. Evaluate impact on validation set of pruning each possible node (plus those below it)
2. Greedily remove the node that most improves *validation set* accuracy

Pruning Decision Trees

- Pruning of the decision tree is done by replacing a whole subtree by a leaf node.
- The replacement takes place if a decision rule establishes that the expected error rate in the subtree is greater than in the single leaf.
- For example,



Real-Valued Inputs

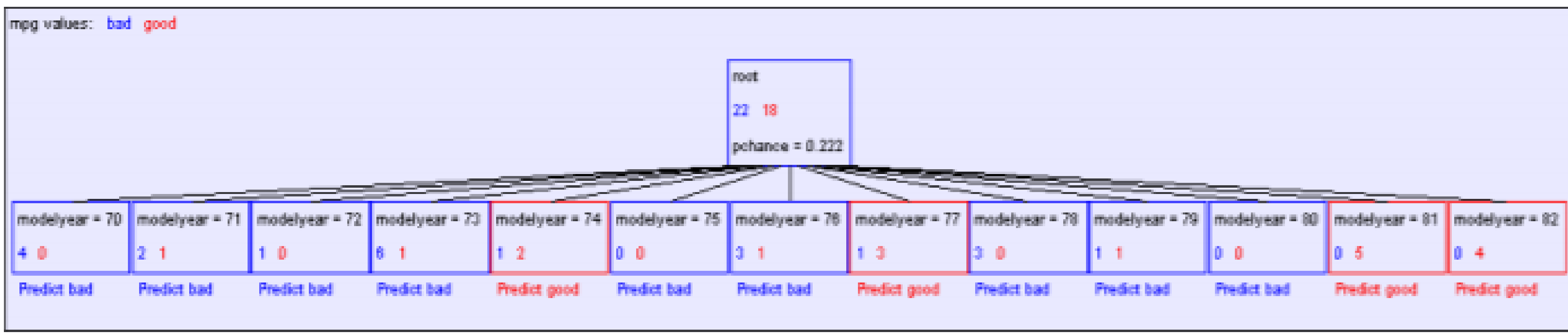
What should we do if some of the inputs are real-valued?

Infinite number of possible split values!!!

mpg	cylinders	displacemen	horsepower	weight	acceleration	modelyear	maker
good	4	97	75	2265	18.2	77	asia
bad	6	199	90	2648	15	70	america
bad	4	121	110	2600	12.8	77	europe
bad	8	350	175	4100	13	73	america
bad	6	198	95	3102	16.5	74	america
bad	4	108	94	2379	16.5	73	asia
bad	4	113	95	2228	14	71	asia
bad	8	302	139	3570	12.8	78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
good	4	120	79	2625	18.6	82	america
bad	8	455	225	4425	10	70	america
good	4	107	86	2464	15.5	76	europe
bad	5	131	103	2830	15.9	78	europe

Naïve Approach

"One branch for each numeric value" idea:

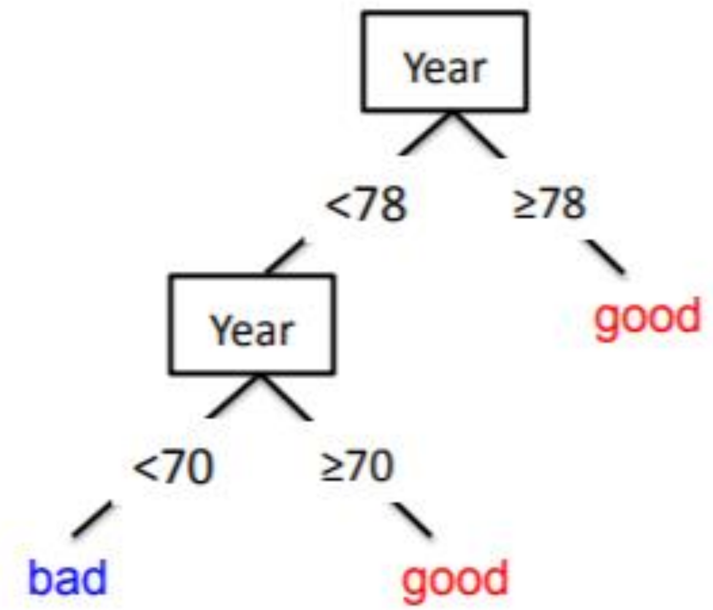


Hopeless: hypothesis with such a high branching factor will shatter *any* dataset and overfit

Threshold Splits

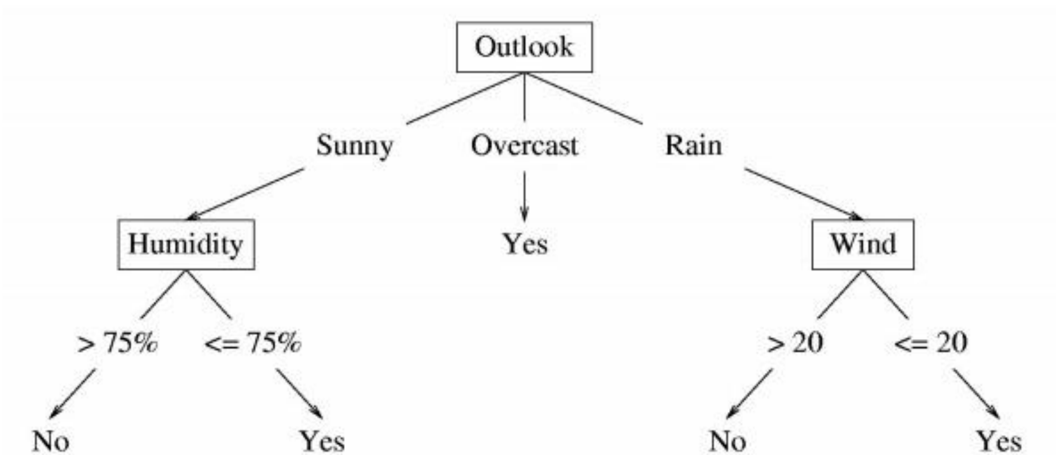
- Binary tree: split on attribute X at value t
  - One branch: X < t
  - Other branch: X ≥ t

- Requires small change
  - Allow repeated splits on same variable **along a path**



Information Gain metric can be extended to numerical attributes

Real-valued Features



- Change to binary splits by choosing a threshold
- One method:
  - Sort instances by value, identify adjacencies with different classes

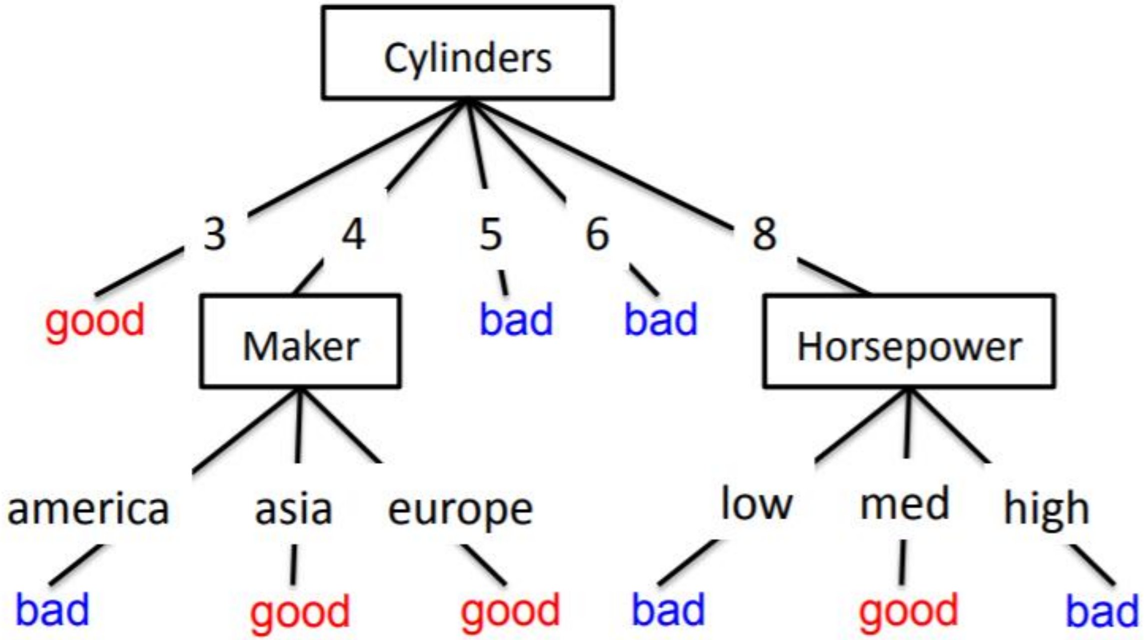
Temperature:	40	48	60	72	80	90
PlayTennis:	No	No	Yes	Yes	Yes	No

candidate splits

  - Choose among splits by InfoGain()

Interpretability

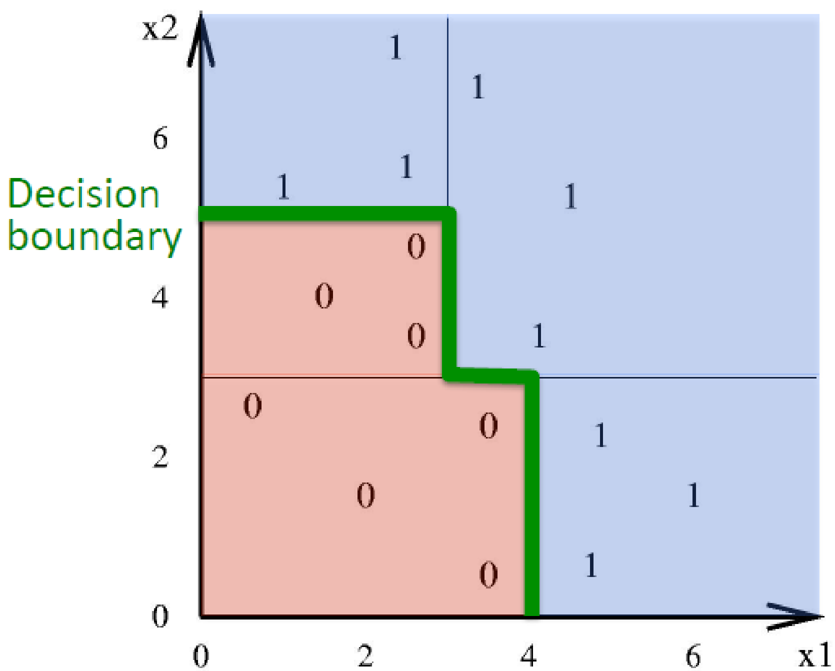
- Each internal node tests an attribute x_i
- One branch for each possible attribute value $x_i = v$
- Each leaf assigns a class y
- To classify input x: traverse the tree from root to leaf, output the labeled y

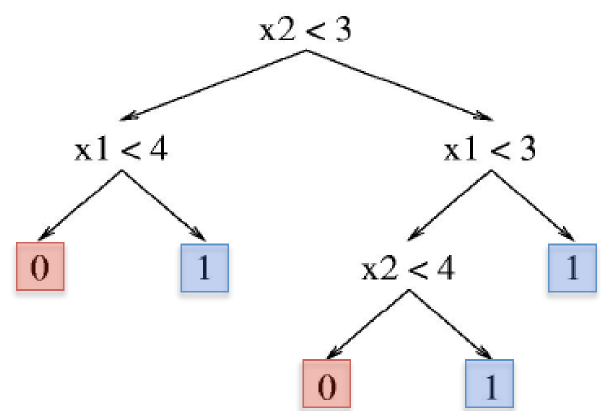


Human interpretable!

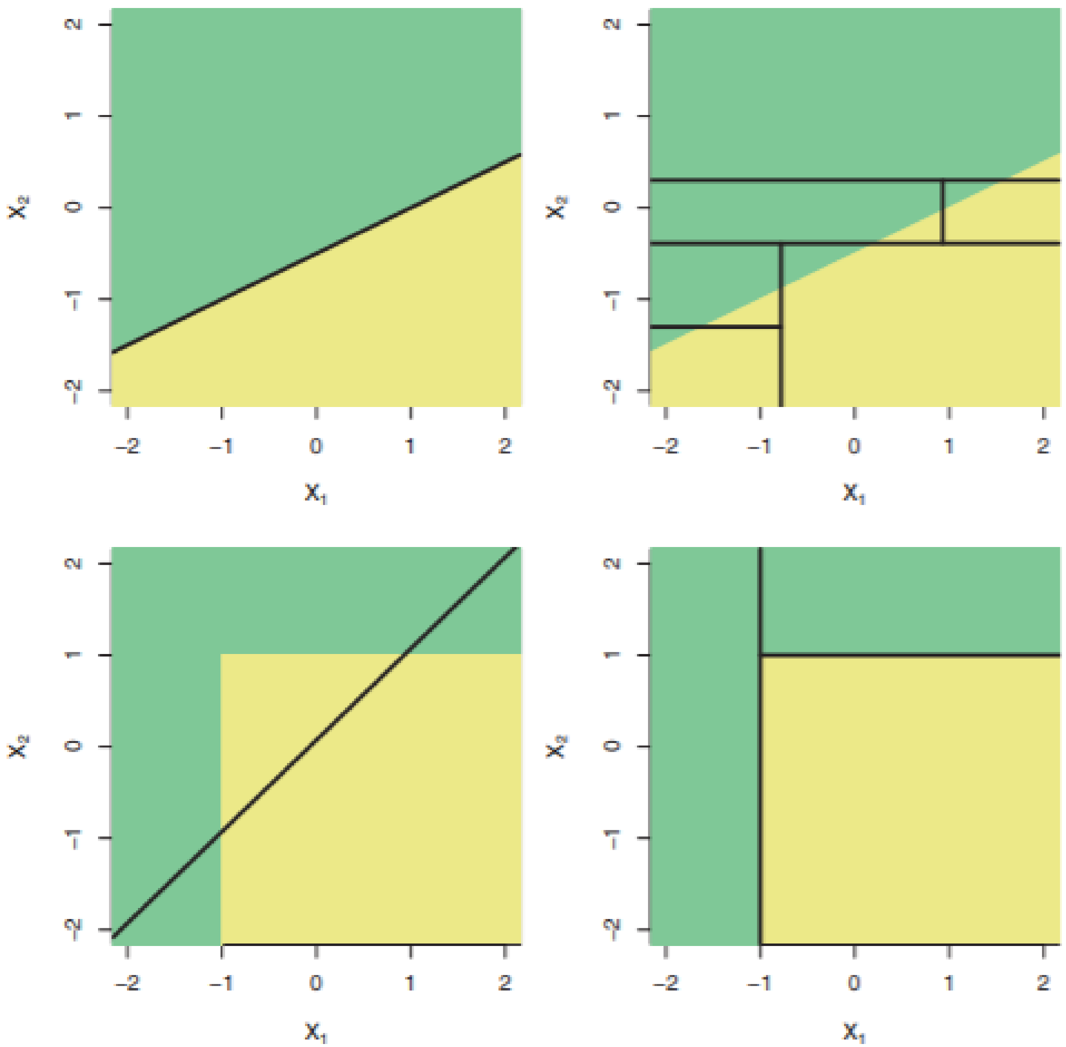
Decision Boundary

- Decision trees divide the feature space into axis-parallel (hyper-)rectangles
- Each rectangular region is labeled with one label
  - or a probability distribution over labels





Decision Trees vs Linear Models



Linear model | Decision tree

Lab

```
> library(tree)
> library(ISLR)
> fix(Carseats)
```

	Sales	CompPrice	Income	Advertising	Population	Price	ShelveLoc	Age
1	9.5	138	73	11	276	120	Bad	42
2	11.22	111	48	16	260	83	Good	65
3	10.06	113	35	10	269	80	Medium	59
4	7.4	117	100	4	466	97	Medium	55
5	4.15	141	64	3	340	128	Bad	38
6	10.81	124	113	13	501	72	Bad	78
7	6.63	115	105	0	45	108	Medium	71
8	11.85	136	81	15	425	120	Good	67
9	6.54	132	110	0	108	124	Medium	76
10	4.69	132	113	0	131	124	Medium	76
11	9.01	121	78	9	150	100	Bad	26
12	11.96	117	94	4	503	94	Good	50
13	3.98	122	35	2	393	136	Medium	62
14	10.96	115	28	11	29	86	Good	53
15	11.17	107	117	11	148	118	Good	52

Lab

Add Label "High" is Sales > 8

```
> High=ifelse(Sales<=8,"No","Yes")
> Carseats=data.frame(Carseats,High)
> head(Carseats)
  Sales CompPrice Income Advertising Population Price ShelveLoc Age Education Urban  US High
1  9.50       138     73          11        276   120       Bad  42        17   Yes Yes  Yes
2 11.22       111     48          16        260    83      Good  65        10   Yes Yes  Yes
3 10.06       113     35          10        269    80    Medium  59        12   Yes Yes  Yes
4  7.40       117    100           4        466    97    Medium  55        14   Yes Yes   No
5  4.15       141     64           3        340   128       Bad  38        13   Yes  No   No
6 10.81       124    113          13        501    72       Bad  78        16    No Yes  Yes
> 
```

Lab

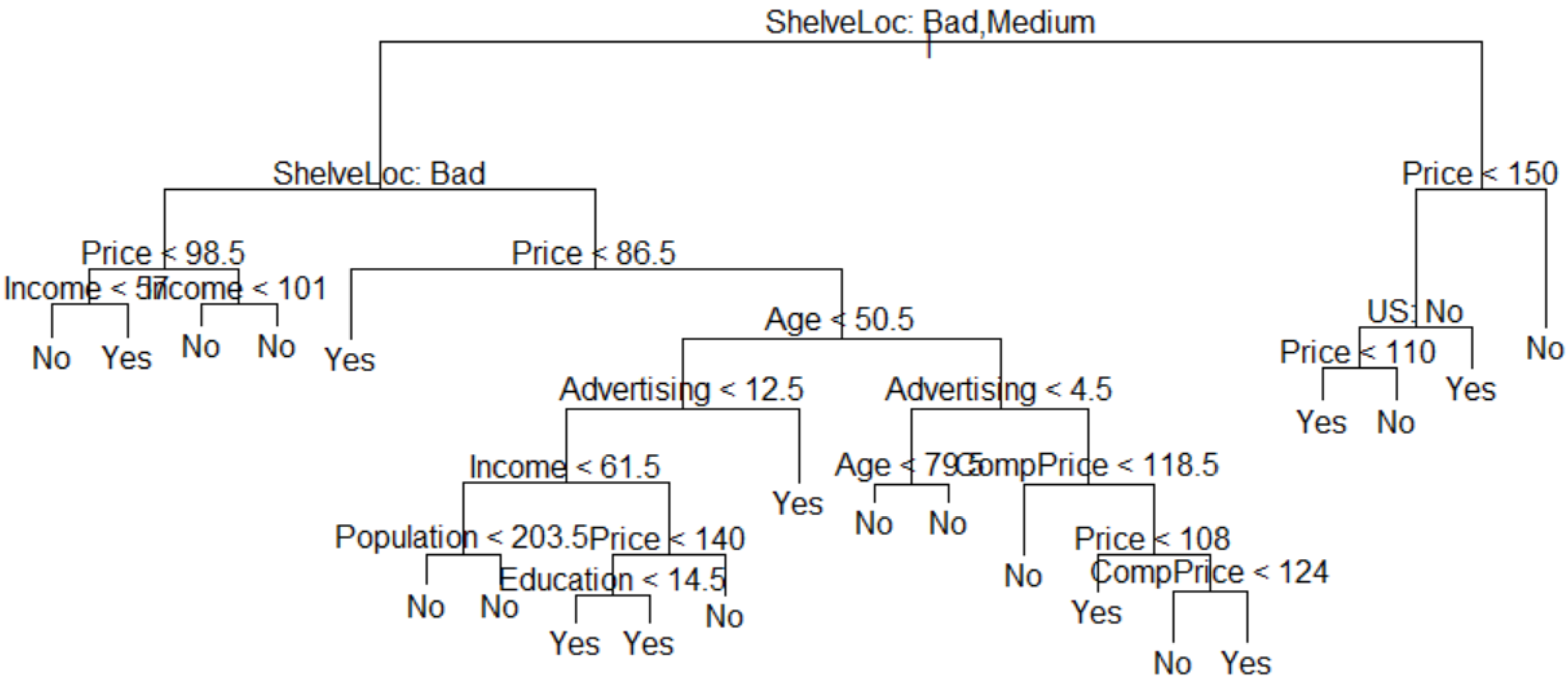
Train and Test

```
> train=sample(1:nrow(Carseats), 200)
> Carseats.test=Carseats[-train,]
> High.test=High[-train]
> tree.carseats=tree(High~.-Sales,Carseats,subset=train)
> tree.pred=predict(tree.carseats,Carseats.test,type="class")
> table(tree.pred,High.test)
         High.test
tree.pred No Yes
      No  85  22
      Yes 34  59
> mean(tree.pred==High.test)
[1] 0.72
>
```

Accuracy

Lab

```
> plot(tree.carseats)
> text(tree.carseats,pretty=0)
>
```



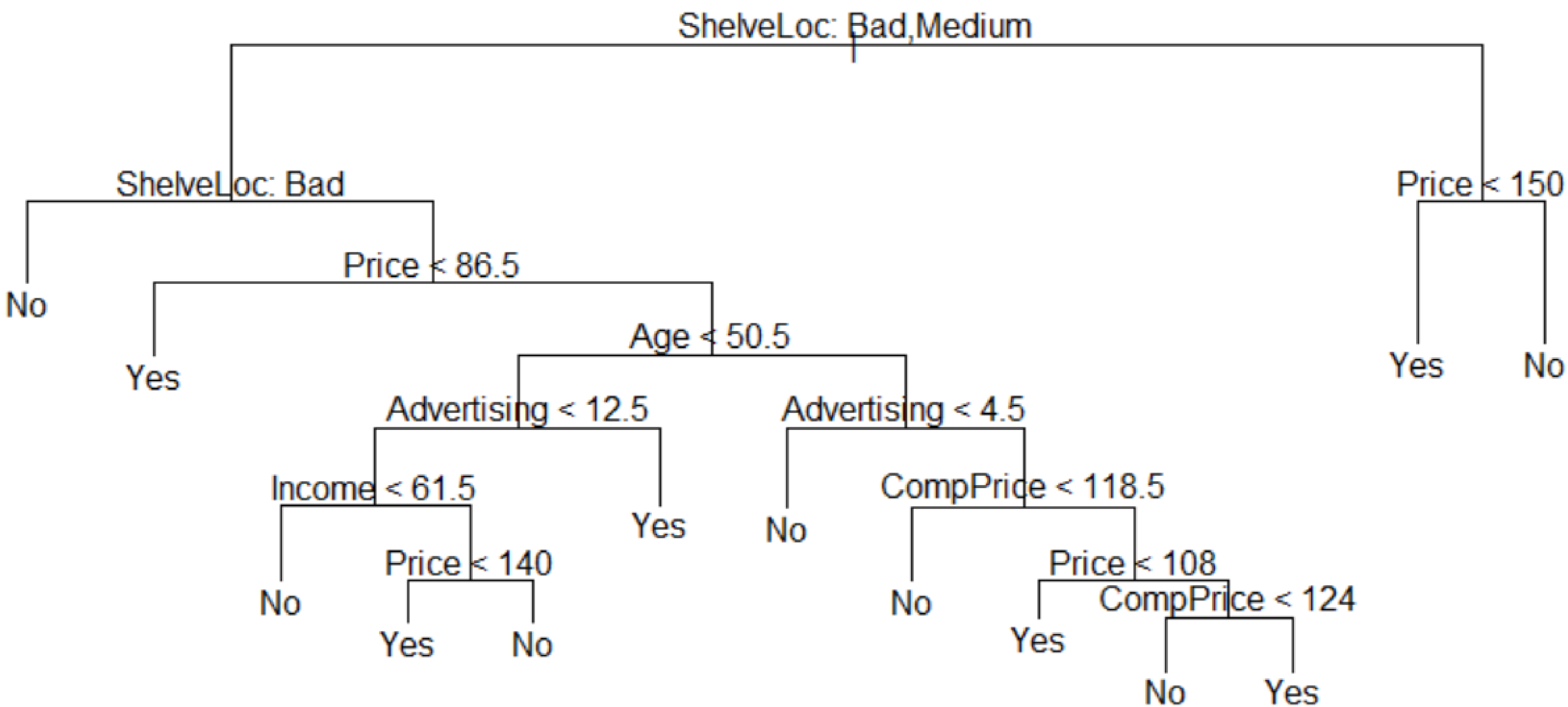
```
> tree.carseats
node), split, n, deviance, yval, (yprob)
      * denotes terminal node

  1) root 200 271.500 No ( 0.58500 0.41500 )
    2) ShelveLoc: Bad,Medium 157 196.500 No ( 0.68153 0.31847 )
      4) ShelveLoc: Bad 46  31.630 No ( 0.89130 0.10870 )
        8) Price < 98.5 13  16.050 No ( 0.69231 0.30769 )
         16) Income < 57 6   0.000 No ( 1.00000 0.00000 ) *
         17) Income > 57 7   9.561 Yes ( 0.42857 0.57143 ) *
        9) Price > 98.5 33   8.962 No ( 0.96970 0.03030 )
         18) Income < 101 28   0.000 No ( 1.00000 0.00000 ) *
         19) Income > 101 5   5.004 No ( 0.80000 0.20000 ) *
      5) ShelveLoc: Medium 111 149.900 No ( 0.59459 0.40541 )
       10) Price < 86.5 7   0.000 Yes ( 0.00000 1.00000 ) *
       11) Price > 86.5 104 136.500 No ( 0.63462 0.36538 )
         22) Age < 50.5 47  64.620 Yes ( 0.44681 0.55319 )
           44) Advertising < 12.5 37  50.620 No ( 0.56757 0.43243 )
             88) Income < 61.5 17  12.320 No ( 0.88235 0.11765 )
              176) Population < 203.5 5   6.730 No ( 0.60000 0.40000 ) *
              177) Population > 203.5 12   0.000 No ( 1.00000 0.00000 ) *
             89) Income > 61.5 20  24.430 Yes ( 0.30000 0.70000 )
              178) Price < 140 15  11.780 Yes ( 0.13333 0.86667 )
                356) Education < 14.5 10   0.000 Yes ( 0.00000 1.00000 )
                357) Education > 14.5 5   6.730 Yes ( 0.40000 0.60000 ) *
              179) Price > 140 5   5.004 No ( 0.80000 0.20000 ) *
           45) Advertising > 12.5 10   0.000 Yes ( 0.00000 1.00000 ) *
         23) Age > 50.5 57  58.670 No ( 0.78947 0.21053 )
           46) Advertising < 4.5 31   8.835 No ( 0.96774 0.03226 )
             92) Age < 79.5 25   0.000 No ( 1.00000 0.00000 ) *
             93) Age > 79.5 6   5.407 No ( 0.83333 0.16667 ) *
           47) Advertising > 4.5 26  35.430 No ( 0.57692 0.42308 )
             94) CompPrice < 118.5 9   0.000 No ( 1.00000 0.00000 ) *
             95) CompPrice > 118.5 17  22.070 Yes ( 0.35294 0.64706 )
```

Pruning

```
> set.seed(3)
> cv.carseats=cv.tree(tree.carseats,FUN=prune.misclass)
> prune.carseats=prune.misclass(tree.carseats,best=12)
> plot(prune.carseats)
> text(prune.carseats,pretty=0)
>
```

- Cross-validation for pruning
- FUN = prune.misclass indicates that classification error is metric to minimize

Pruning



ShelveLoc: Bad,Medium
ShelveLoc: Bad
Price < 150
No
Price < 86.5
Yes
Age < 50.5
Yes
No
Advertising < 12.5
Advertising < 4.5
Income < 61.5
Yes
No
CompPrice < 118.5
No
Price < 140
No
Price < 108
Yes
No
CompPrice < 124
Yes
No
Yes

Summary Decision Trees

- Representation: decision trees
- Bias: prefer small decision trees
- Search algorithm: greedy
- Heuristic function: information gain or information content or others
- Overfitting / pruning

Strengths
- Fast to evaluate
- Interpretable
- Generate rules
- Supports categorical and numerical data

Weaknesses
- Overfitting
- Splitting method might not be optimal
- Accuracy is not always high
- Batch learning

Acknowledgements

- Slides made using resources from:
  - Andrew Ng
  - Eric Eaton
  - David Sontag
- Thanks!