CS 360: Machine Learning

Prof. Sara Mathieson

Fall 2019

HAVERFORD
COLLEGE

Admin

- Office hours: TODAY 12:30-1:30pm (L310)
  - Also make use of TA office hours Sun/Mon

- Lab 1 due Tues night (TODAY)
  - Make sure to push your final code on git
  - Turn in math portion on paper (slide under door)
  - There is one with no name

- Next reading quiz: Thursday on Duame 1.3-1.6 and chapter 2

Outline for September 10

- Recap featurization discussion
- Begin: Decision Trees
- ID3 Decision Tree algorithm
- Entropy

Outline for September 10

- Recap featurization discussion
- Begin: Decision Trees
- ID3 Decision Tree algorithm
- Entropy

Back to Handout 2

4. Using your response from the previous question, what would the *feature vector* become for $\boldsymbol{x}_1$?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis (y)
$\boldsymbol{x}_1$	Sunny	Hot	High	Weak	No
$\boldsymbol{x}_2$	Sunny	Hot	High	Strong	No
$\boldsymbol{x}_3$	Overcast	Hot	High	Weak	Yes
$\boldsymbol{x}_4$	Rain	Mild	High	Weak	Yes
$\boldsymbol{x}_5$	Rain	Cool	Normal	Weak	Yes
$\boldsymbol{x}_6$	Rain	Cool	Normal	Strong	No
$\boldsymbol{x}_7$	Overcast	Cool	Normal	Strong	Yes
$\boldsymbol{x}_8$	Sunny	Mild	High	Weak	No
$\boldsymbol{x}_9$	Sunny	Cool	Normal	Weak	Yes
$\boldsymbol{x}_{10}$	Rain	Mild	Normal	Weak	Yes

Data from Machine Learning by Tom Mitchell (Table 3.2)

Back to Handout 2

4. Using your response from the previous question, what would the *feature vector* become for $\boldsymbol{x}_1$?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis (y)
$\boldsymbol{x}_1$	Sunny	Hot	High	Weak	No
$\boldsymbol{x}_2$	Sunny	Hot	High	Strong	No
$\boldsymbol{x}_3$	Overcast	Hot	High	Weak	Yes
$\boldsymbol{x}_4$	Rain	Mild	High	Weak	Yes
$\boldsymbol{x}_5$	Rain	Cool	Normal	Weak	Yes
$\boldsymbol{x}_6$	Rain	Cool	Normal	Strong	No
$\boldsymbol{x}_7$	Overcast	Cool	Normal	Strong	Yes
$\boldsymbol{x}_8$	Sunny	Mild	High	Weak	No
$\boldsymbol{x}_9$	Sunny	Cool	Normal	Weak	Yes
$\boldsymbol{x}_{10}$	Rain	Mild	Normal	Weak	Yes

Data from Machine Learning by Tom Mitchell (Table 3.2)

Sunny:	{0,1}	
Overcast:	{0,1}	
Rain:	{0,1}	
Temperature:	{0, 1, 2}	(Cool, Mild, Hot)
Humidity:	{0,1}	(Normal, High)
Wind	{0,1}	(Weak, Strong)

Back to Handout 2

4. Using your response from the previous question, what would the *feature vector* become for $\boldsymbol{x}_1$?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis (y)
$\boldsymbol{x}_1$	Sunny	Hot	High	Weak	No
$\boldsymbol{x}_2$	Sunny	Hot	High	Strong	No
$\boldsymbol{x}_3$	Overcast	Hot	High	Weak	Yes
$\boldsymbol{x}_4$	Rain	Mild	High	Weak	Yes
$\boldsymbol{x}_5$	Rain	Cool	Normal	Weak	Yes
$\boldsymbol{x}_6$	Rain	Cool	Normal	Strong	No
$\boldsymbol{x}_7$	Overcast	Cool	Normal	Strong	Yes
$\boldsymbol{x}_8$	Sunny	Mild	High	Weak	No
$\boldsymbol{x}_9$	Sunny	Cool	Normal	Weak	Yes
$\boldsymbol{x}_{10}$	Rain	Mild	Normal	Weak	Yes

Data from Machine Learning by Tom Mitchell (Table 3.2)

Sunny:	{0,1}	
Overcast:	{0,1}	
Rain:	{0,1}	
Temperature:	{0, 1, 2}	(Cool, Mild, Hot)
Humidity:	{0,1}	(Normal, High)
Wind	{0,1}	(Weak, Strong)

	Sunny	Overcast	Rain	Temp	Humidity	Wind
$\boldsymbol{x}_1$	1	0	0	2	1	0

Featurization (rule of thumb)

- Real-valued features get copied directly. *Duame, Chap 3*
- Binary features become 0 (for false) or 1 (for true).
- Categorical features with V possible values get mapped to V-many binary indicator features.

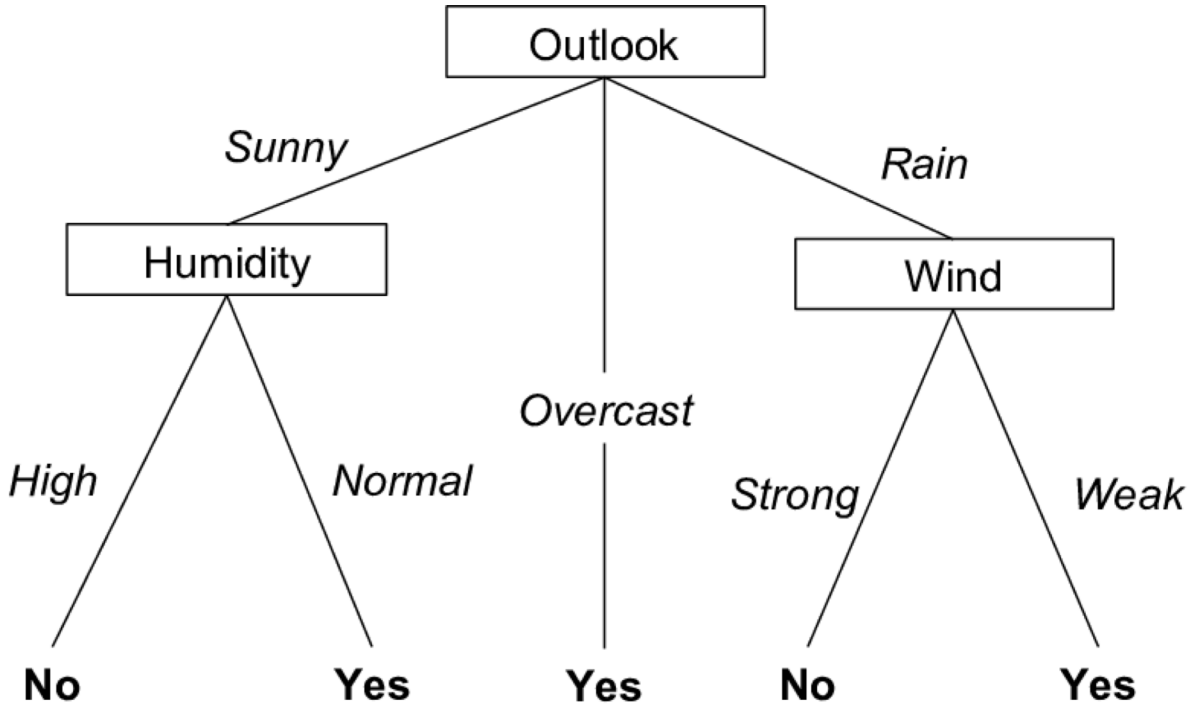
Haven't discussed:

-normalization

Outline for September 10

- Recap featurization discussion
- Begin: Decision Trees
- ID3 Decision Tree algorithm
- Entropy

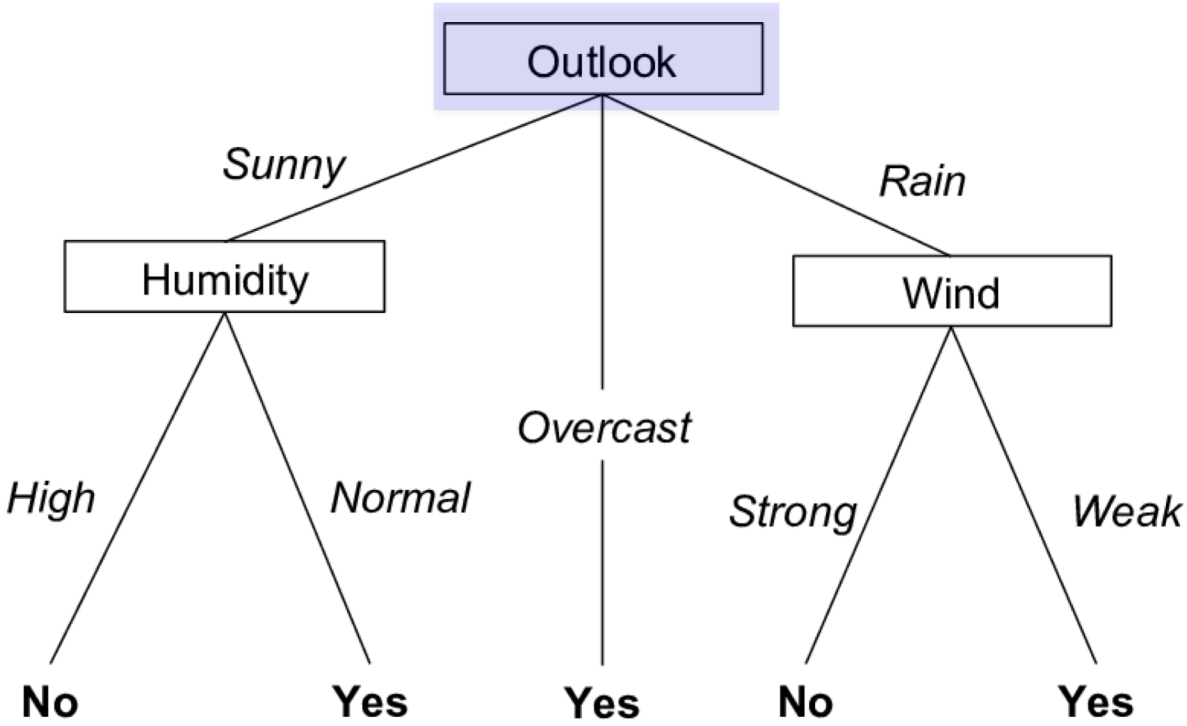
Decision Tree example (tennis data)



- Each internal node: test one feature
- Each branch from node: selects one value of the feature
- Each leaf node: predict *y*

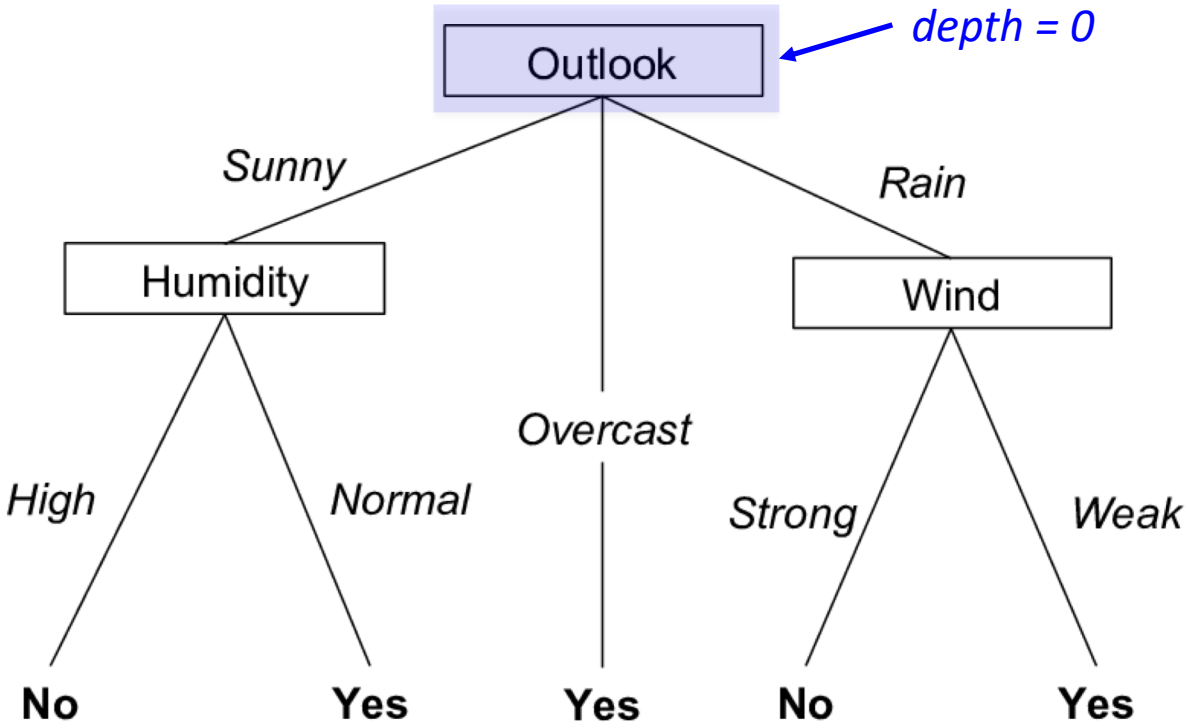
Based on slides by Jessica Wu and Eric Eaton
[originally by Tom Mitchell]

Decision Tree example (tennis data)



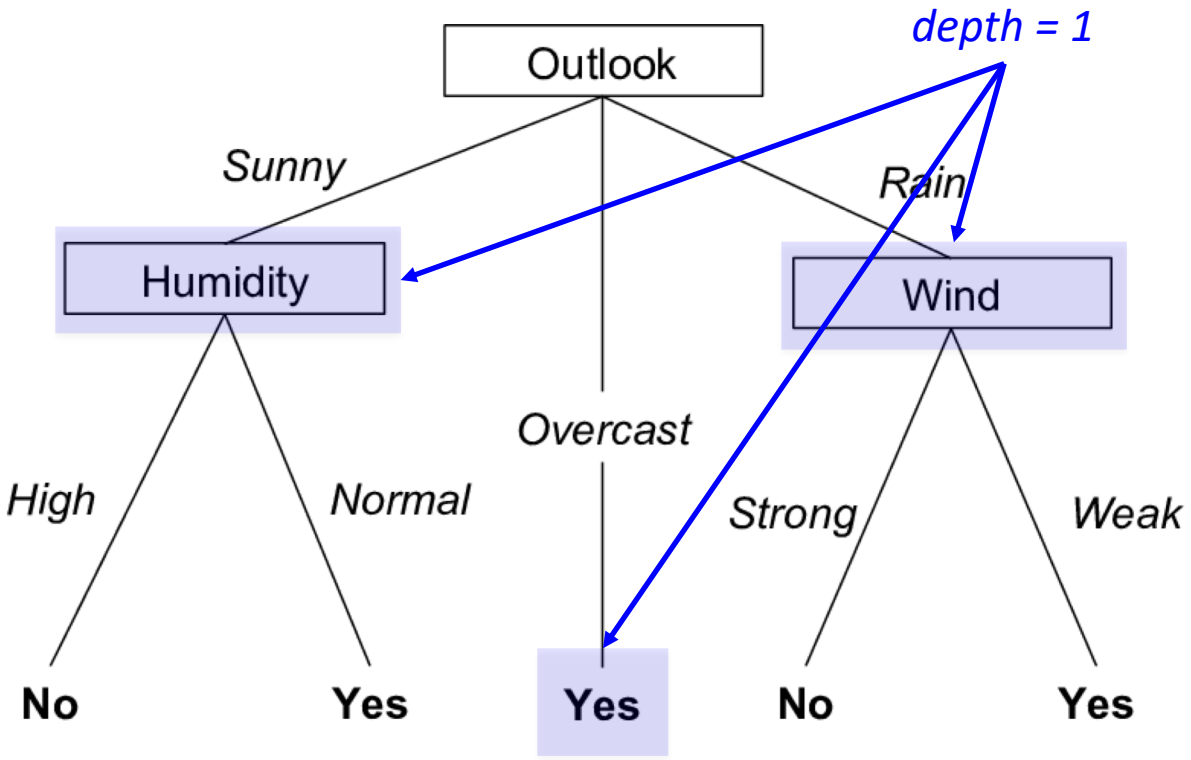
Key term: *depth*

Decision Tree example (tennis data)



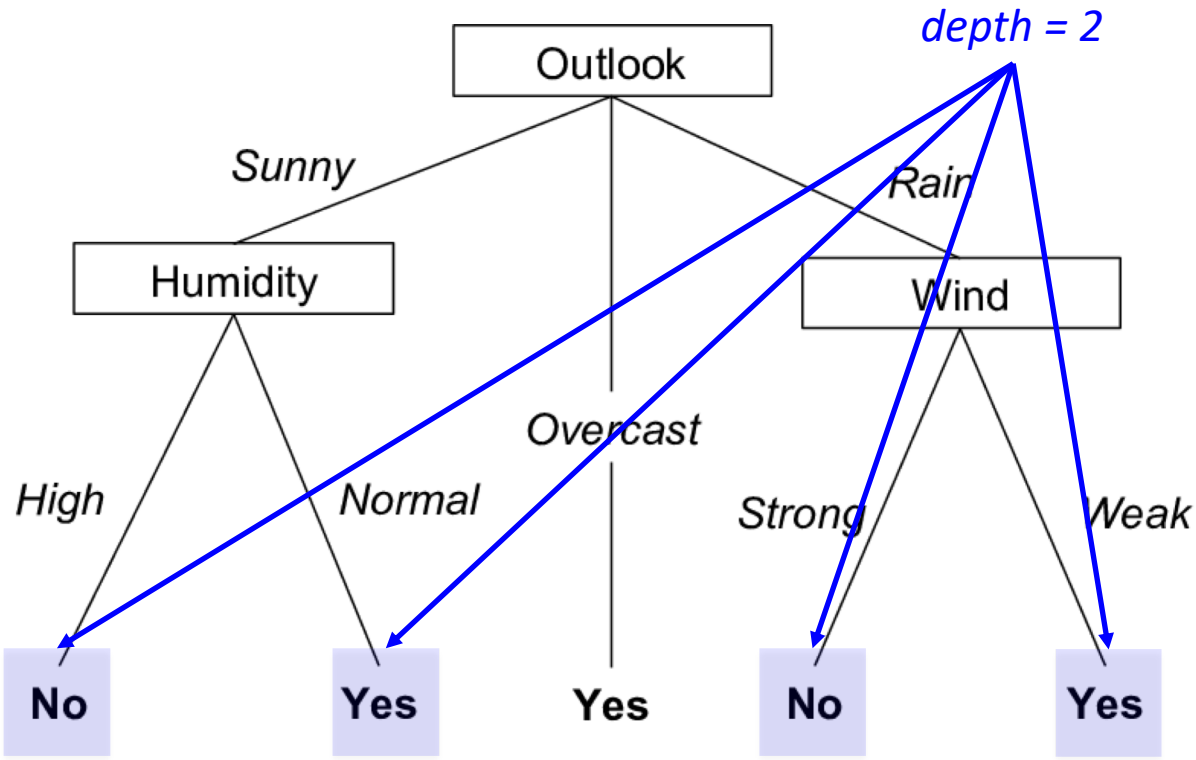
Key term: *depth*

Decision Tree example (tennis data)



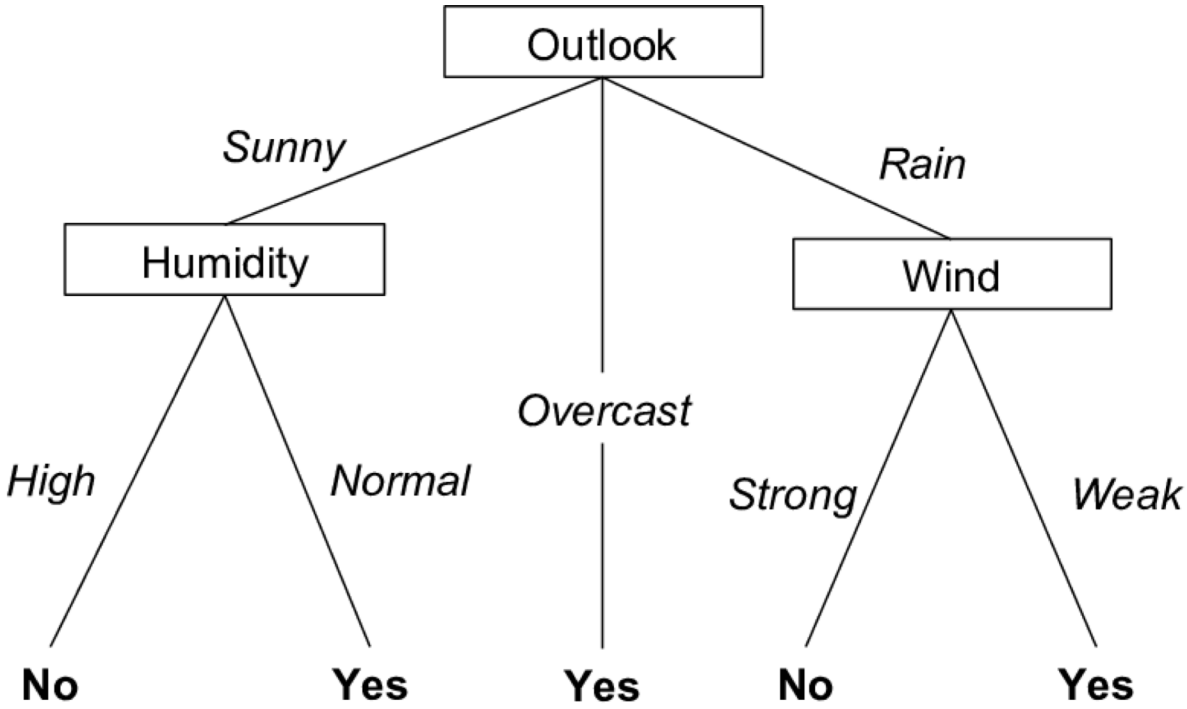
Key term: *depth*

Decision Tree example (tennis data)



Key term: *depth*

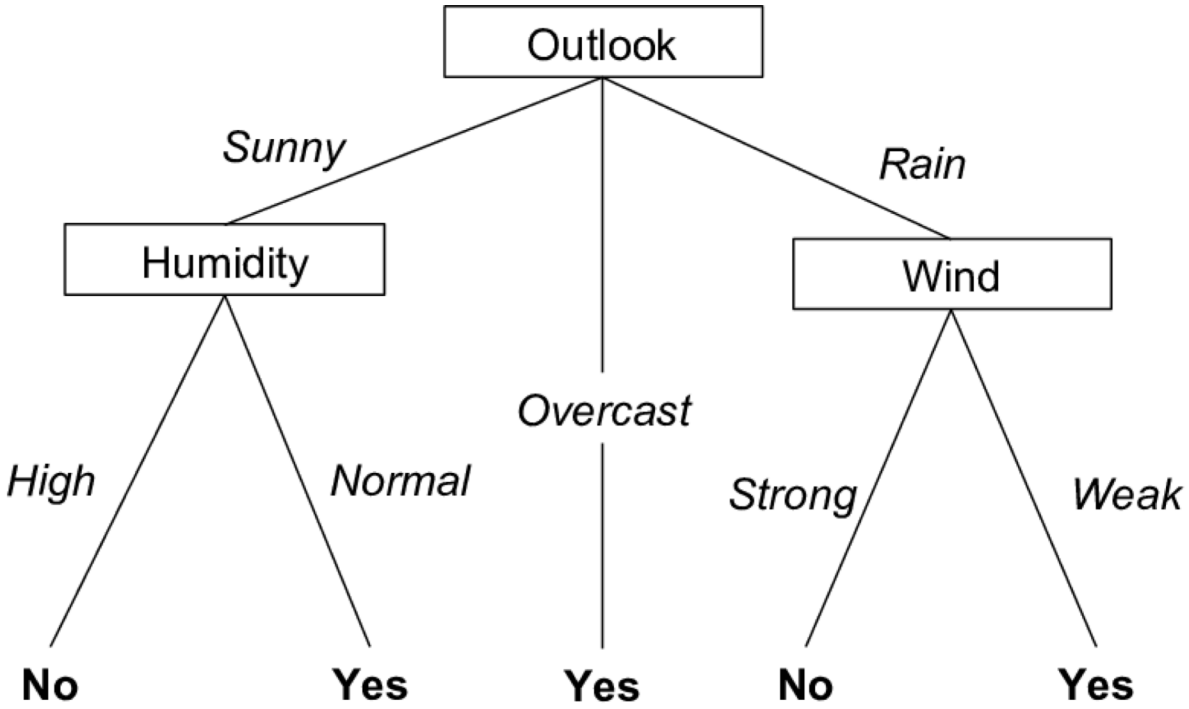
Decision Tree example (tennis data)



(test example) ***x*** =

Outlook	Temp	Humidity	Wind
Rain	Hot	High	Strong

Decision Tree example (tennis data)

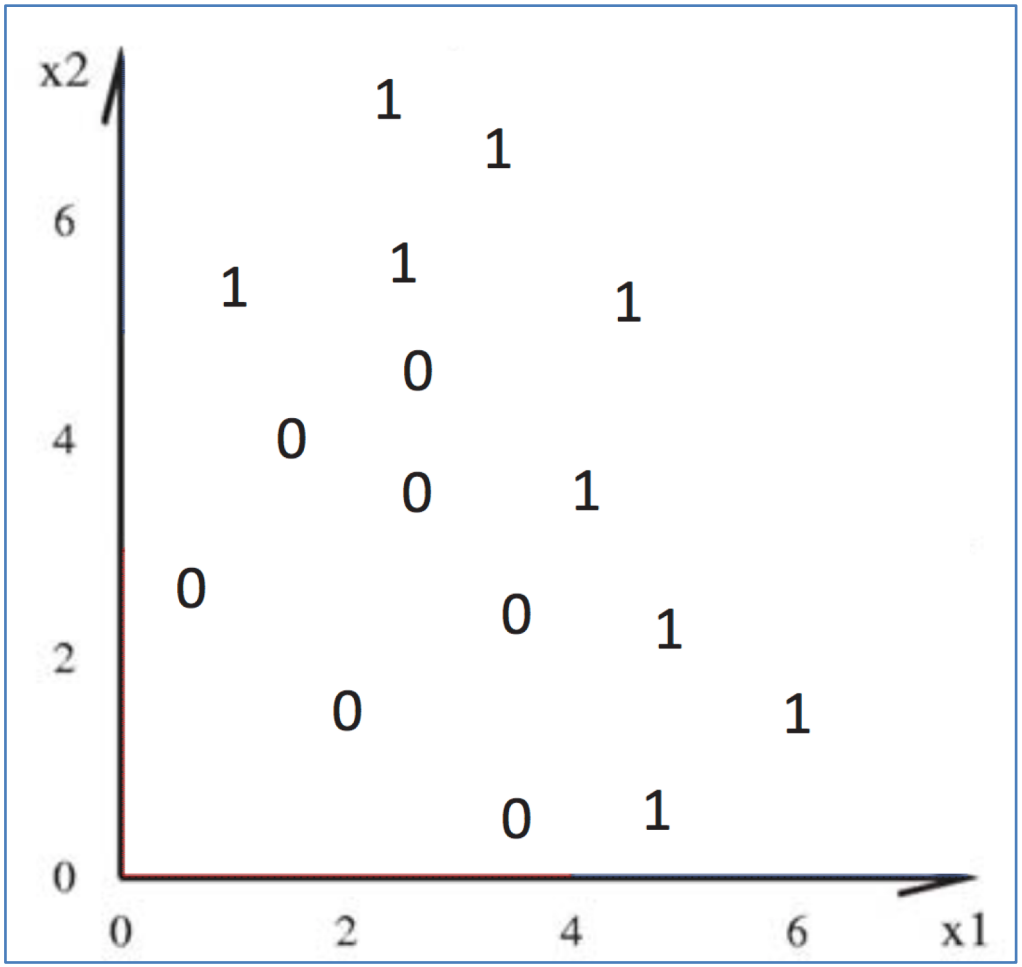


(test example) ***x*** =

Outlook	Temp	Humidity	Wind
Rain	Hot	High	Strong

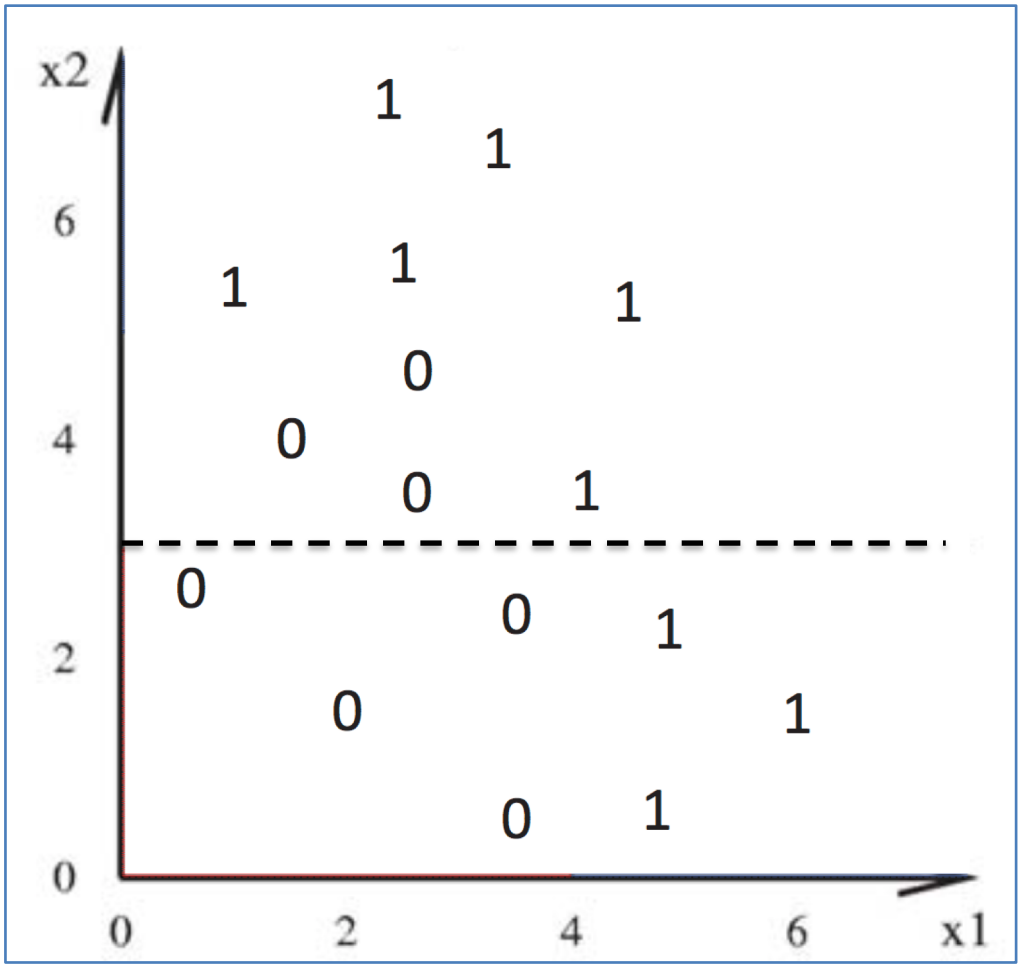
y_{pred} = No

Can also consider continuous features



Example by: Eric Eaton

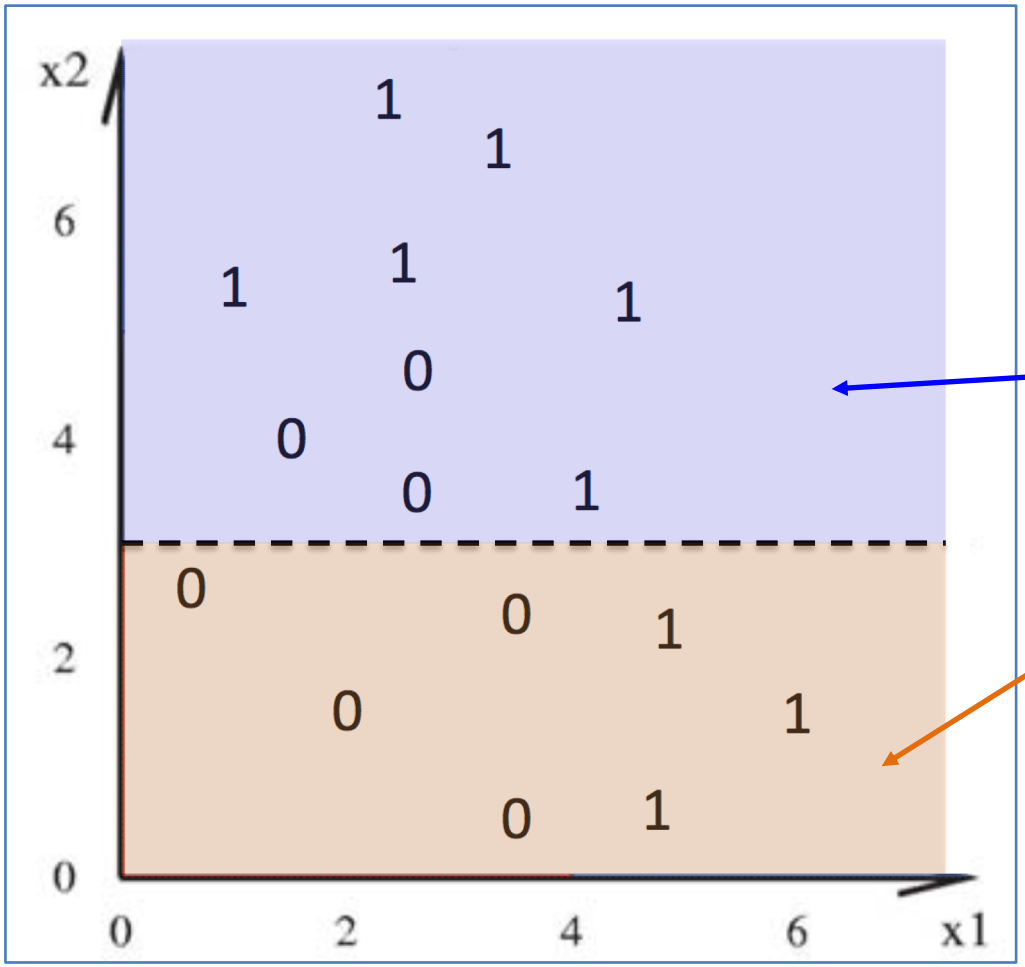
Can also consider continuous features

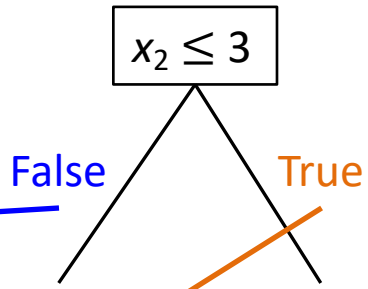


$x_2 \leq 3$

Example by: Eric Eaton

Can also consider continuous features





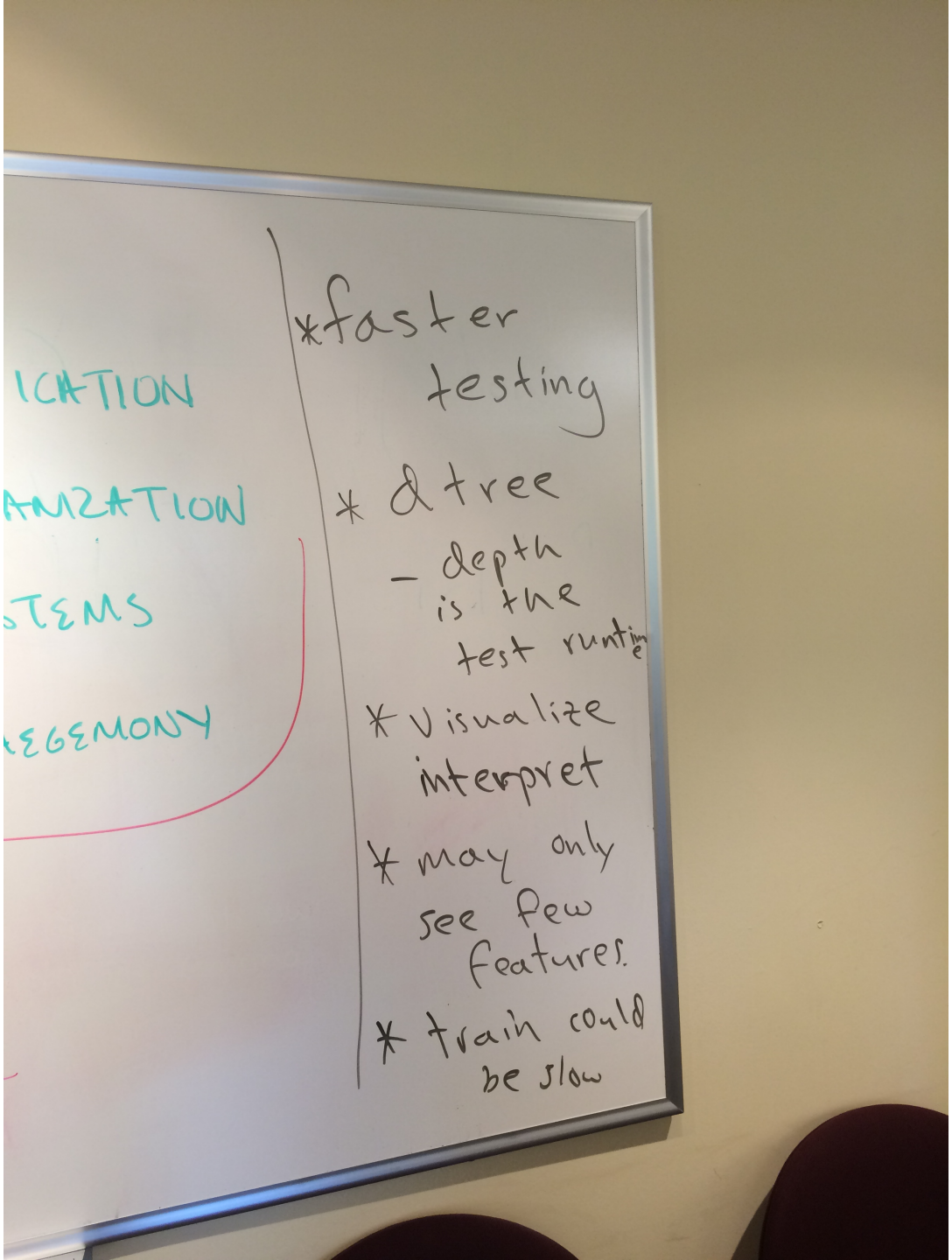
Correction!

Example by: Eric Eaton

Decision Tree pros/cons

Discuss with a partner! Think about:

* training and testing
* featurization
* runtime
* human factors


ICATION
ANIZATION
STEMS
EGEMONY
*faster
testing
* d tree
- depth
is the
test runtime
* visualize
interpret
* may only
see few
features.
* train could
be slow

Decision Tree pros/cons

- Very interpretable! Easy to say *why* we made a classification (can point to which features)
- Compact representation and fast predictions

- Can be brittle (not looking at each example holistically)
- Featurization and implementation difficulties

Outline for September 10

- Recap featurization discussion
- Begin: Decision Trees
- ID3 Decision Tree algorithm
- Entropy

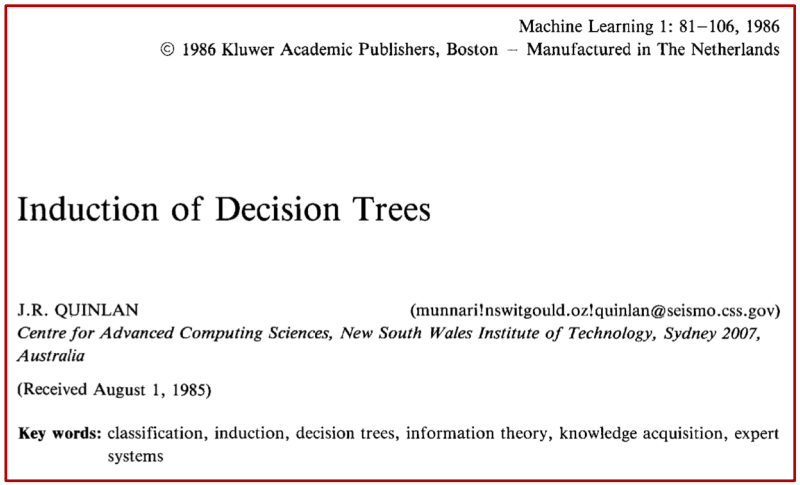
ID3 Decision Tree algorithm (1986)

- Select feature that "best" informs label prediction (i.e. *y*)

- **Divide**: partition data into branches based on their value at this feature

- **Conquer**: recurse on each partition

Posted as optional reading



Machine Learning 1: 81–106, 1986

Induction of Decision Trees

J.R. QUINLAN (munnari!nswitgould.oz!quinlan@seismo.css.gov)
Centre for Advanced Computing Sciences, New South Wales Institute of Technology, Sydney 2007, Australia

(Received August 1, 1985)

Key words: classification, induction, decision trees, information theory, knowledge acquisition, expert systems

Top-Down decision tree algorithm

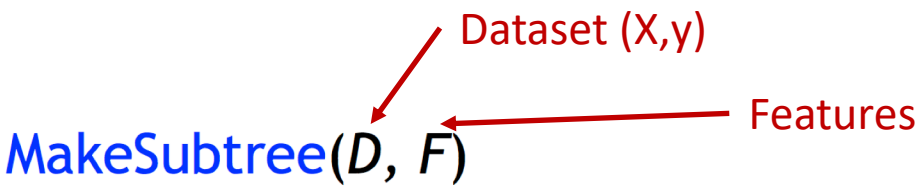
```
MakeSubtree(D, F)
  if stopping criteria met
      make a leaf node N
      determine class label/probabilities for N
```

Slide: modified from Ameet Soni

Top-Down decision tree algorithm

MakeSubtree(*D*, *F*)
  if stopping criteria met
    make a leaf node *N*
    determine class label/probabilities for *N*

Top-Down decision tree algorithm




```
MakeSubtree(D, F)
  if stopping criteria met
      make a leaf node N
      determine class label/probabilities for N
```

Top-Down decision tree algorithm

Dataset (X,y)

Features

```
MakeSubtree(D, F)
  if stopping criteria met
      make a leaf node N
      determine class label/probabilities for N
```

For us: use majority label (break ties arbitrarily)

Top-Down decision tree algorithm

Dataset (X,y)

Features

```
MakeSubtree(D, F)
  if stopping criteria met
      make a leaf node N
      determine class label/probabilities for N
  else
      make an internal node N
      S = FindBestFeature(D, F)
      for each outcome k of S
          D_k = subset of instances that have outcome k
          N.child[k] = MakeSubtree(D_k,F-S)
  return subtree rooted at N
```

For us: use majority label (break ties arbitrarily)

Slide: modified from Ameet Soni

Top-Down decision tree algorithm

Dataset (X,y)

Features

```
MakeSubtree(D, F)
  if stopping criteria met
      make a leaf node N
      determine class label/probabilities for N
  else
      make an internal node N
      S = FindBestFeature(D, F)
      for each outcome k of S
          D_k = subset of instances that have outcome k
          N.child[k] = MakeSubtree(D_k, F-S)
  return subtree rooted at N
```

For us: use majority label (break ties arbitrarily)

Why don't we want to use this feature again?

Top-Down decision tree algorithm

Dataset (X,y)

Features

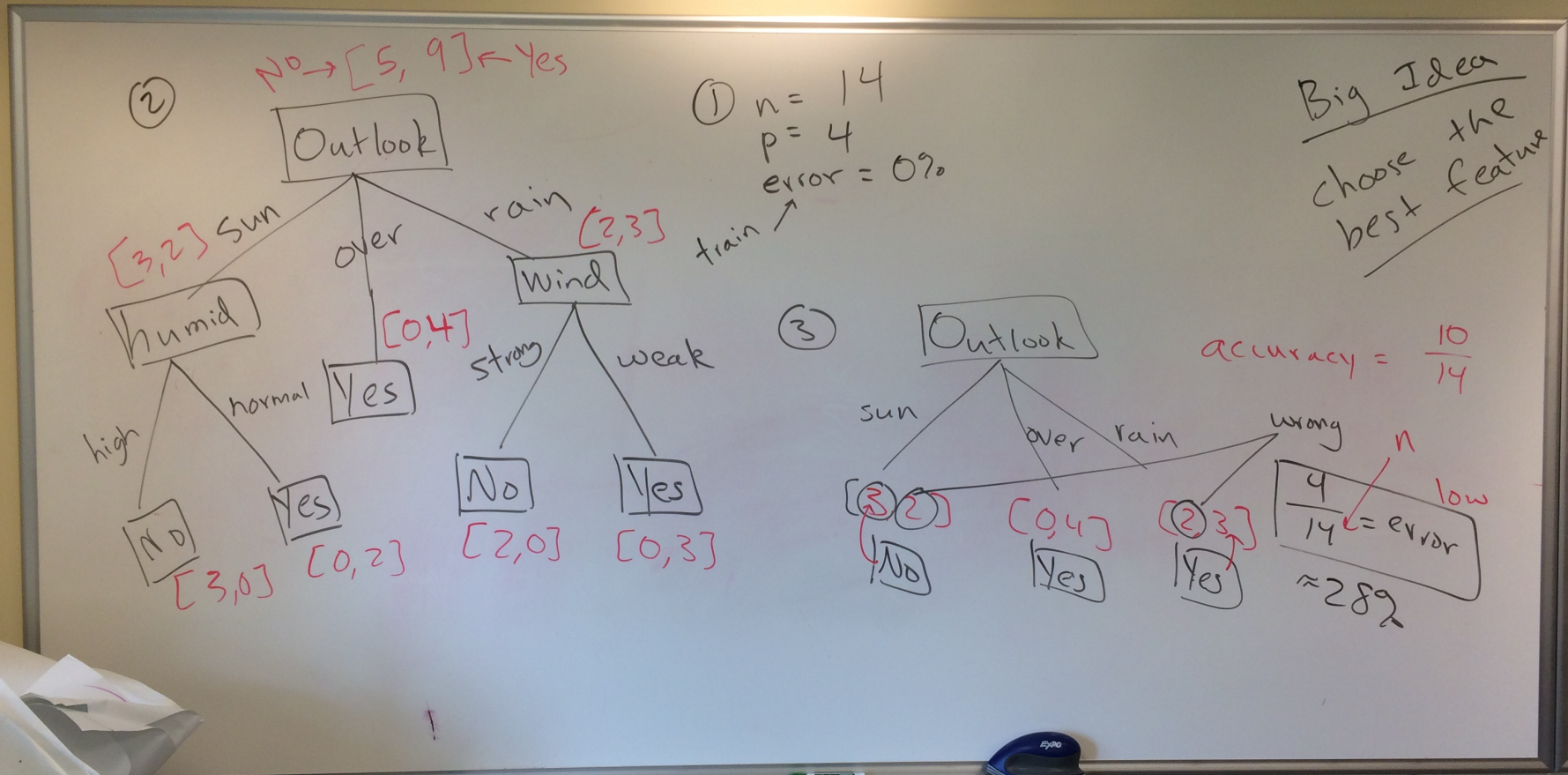
```
MakeSubtree(D, F)
  if stopping criteria met
      make a leaf node N
      determine class label/probabilities for N
  else
      make an internal node N
      S = FindBestFeature(D, F)
      for each outcome k of S
          D_k = subset of instances that have outcome k
          N.child[k] = MakeSubtree(D_k, F-S)
  return subtree rooted at N
```

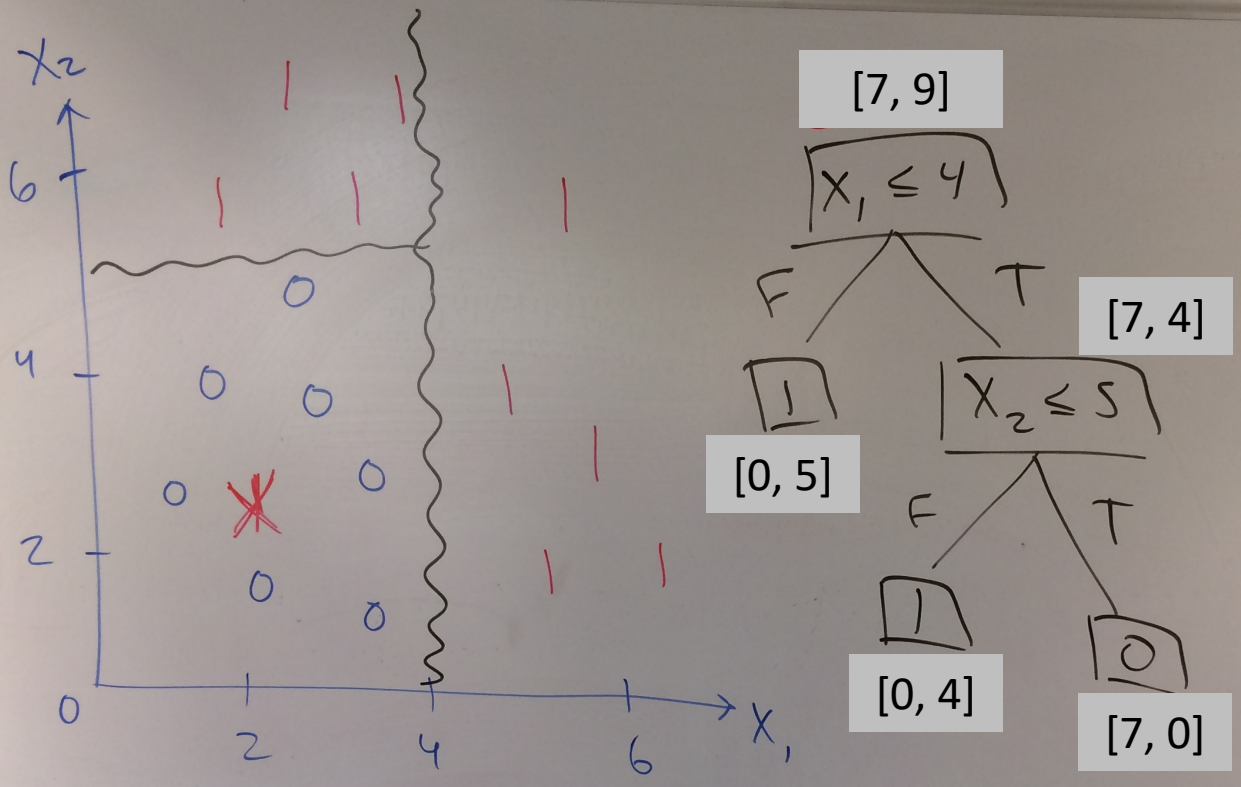
For us: use majority label (break ties arbitrarily)

Why don't we want to use this feature again?

Now: Handout 3 + think about:
what design choices do we need to make?

Slide: modified from Ameet Soni


② No → [5, 9] ← Yes
Outlook
[3,2] Sun
over
rain [2,3]
humid
[0,4]
Wind
high
normal
strong
weak
No
Yes
Yes
No
Yes
[3,0]
[0,2]
[2,0]
[0,3]
① n = 14
p = 4
error = 0%
train
Big Idea
choose the best feature
③
Outlook
accuracy = 10/14
sun
over
rain
wrong
3 2
[0,4]
2 3
No
Yes
Yes
4/14 = error
n
low
≈ 28%


X2
6
4
2
0
2
4
6
X1

[7, 9]
X1 ≤ 4
F
T
1
[0, 5]
[7, 4]
X2 ≤ 5
F
T
1
[0, 4]
0
[7, 0]

Design choice: stopping criteria

1. All the data points in our partition have the same label

Design choice: stopping criteria

1. All the data points in our partition have the same label

2. No more features remain to split on

Design choice: stopping criteria

1. All the data points in our partition have the same label

2. No more features remain to split on

3. No features are informative about the label

Design choice: stopping criteria

1. All the data points in our partition have the same label

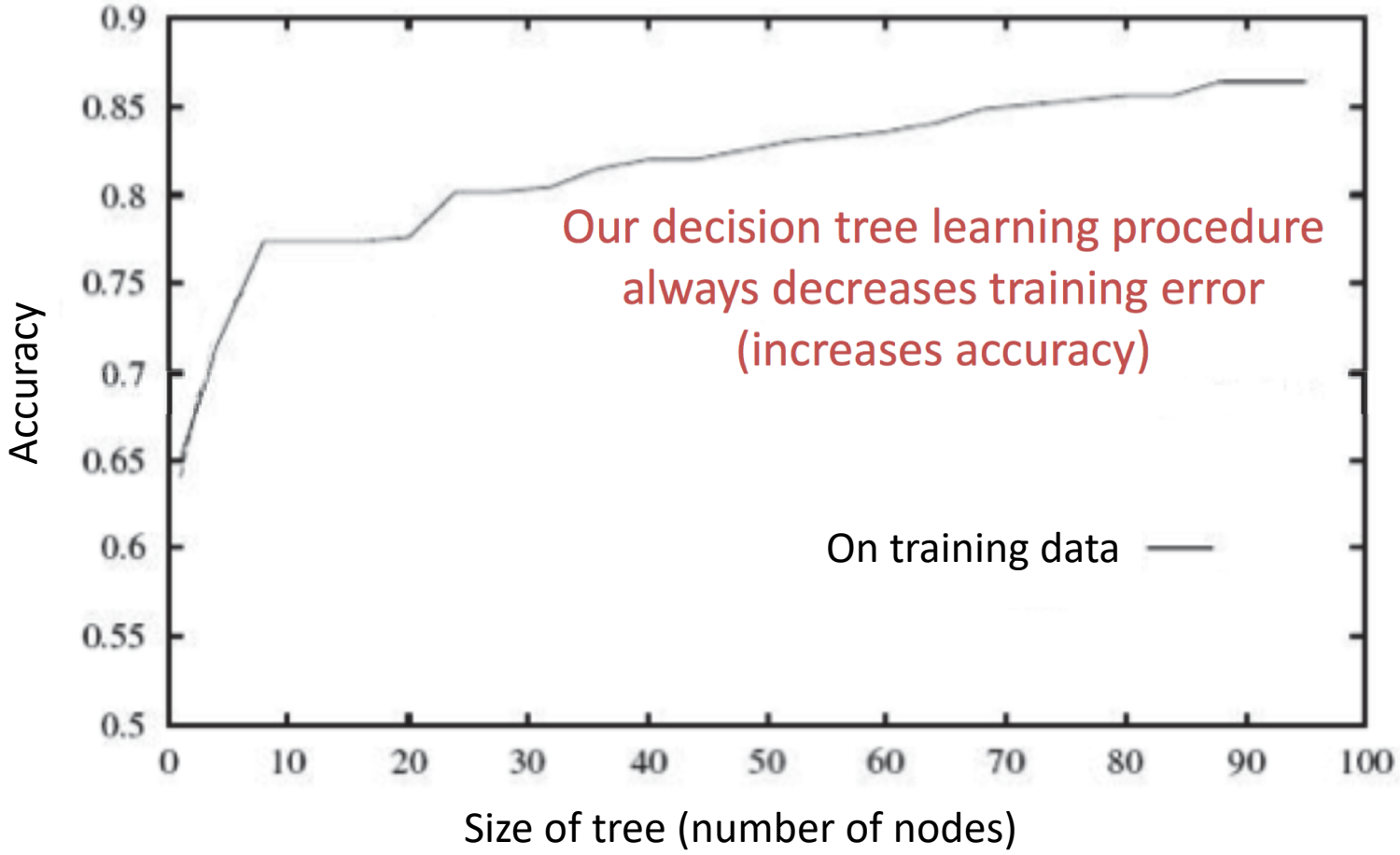
2. No more features remain to split on

3. No features are informative about the label

4. Reached (user specified) max depth in the tree

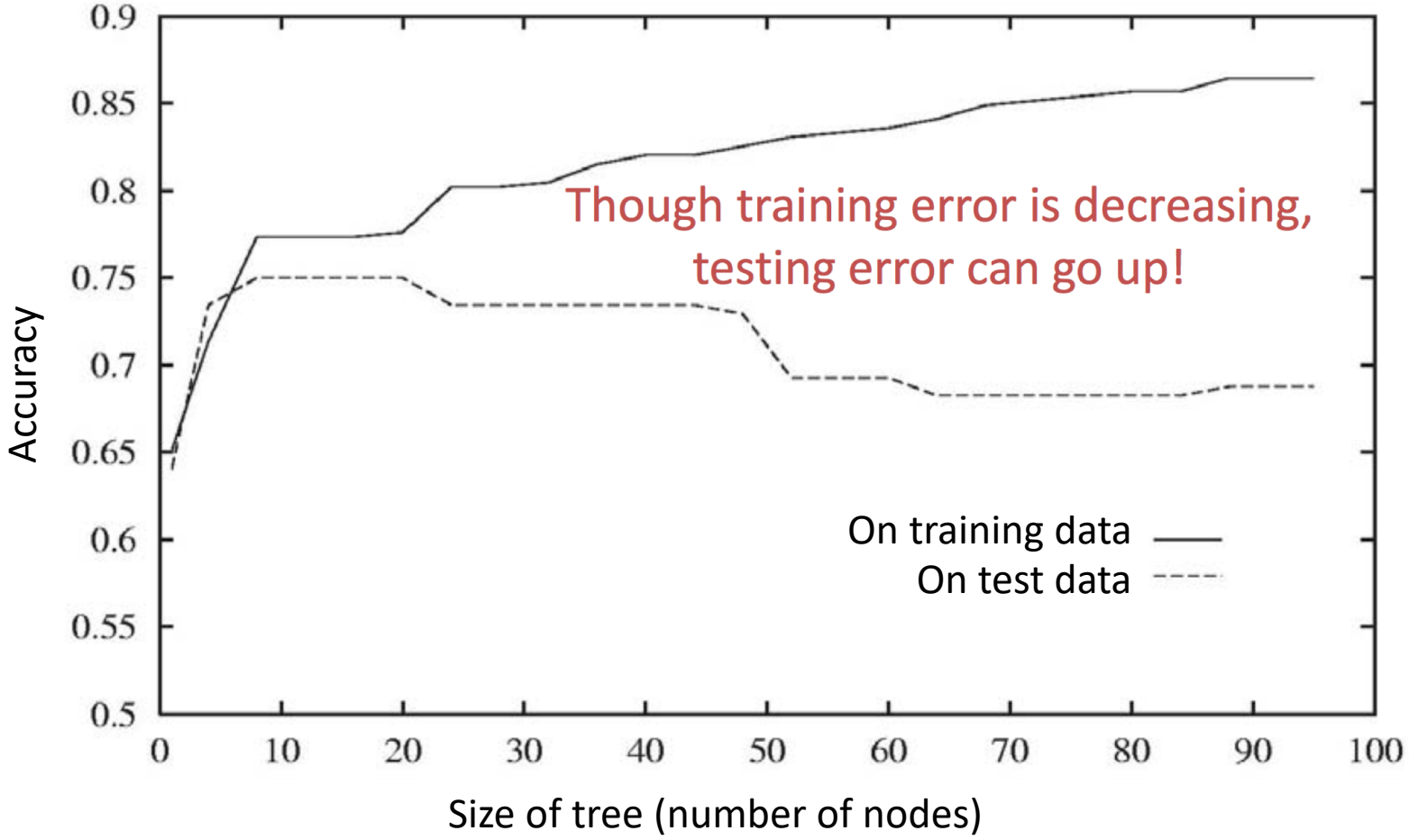
Detour to overfitting

Overfitting



Slide: modified from Jessica Wu
Based on slide by David Kauchak
(originally by Pedro Domingos)

Overfitting



Slide: modified from Jessica Wu
Based on slide by David Kauchak
(originally by Pedro Domingos)

Overfitting definition

- Consider a hypothesis (tree): h
  - Training error: $error_{train}(h)$
  - Error over all possible data: $error_{D}(h)$

Overfitting definition

- Consider a hypothesis (tree): *h*
  - Training error: $error_{train}(h)$
  - Error over all possible data: $error_{D}(h)$

- A hypothesis *h* **overfits** training data if there exists another hypothesis *h'* s.t.
  - $error_{train}(h) < error_{train}(h')$ AND
  - $error_{D}(h) > error_{D}(h')$

Avoiding overfitting for us

- Stop when leaf label reaches a certain fraction (i.e. 95% “yes”, 5% “no”)

For our Lab 2 implementation

- Set a maximum depth for the tree

- Set a minimum number of examples in leaf (i.e. if we have a 2-1 split, stop)

Outline for September 10

- Recap featurization discussion
- Begin: Decision Trees
- ID3 Decision Tree algorithm
- Entropy

How to select “best” features?

X			Y
Color	**Shape**	**Size**	**Likes toy?**
red	square	big	+
blue	square	big	+
red	circle	small	-
blue	square	small	-
red	circle	big	+

Example by: Ameet Soni

How to select “best” features?

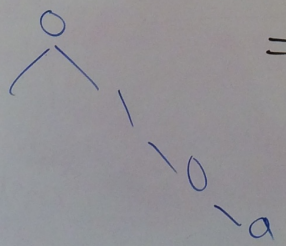
X			Y
Color	**Shape**	**Size**	**Likes toy?**
red	square	big	+
blue	square	big	+
red	circle	**big**	-
blue	square	**big**	-
red	circle	big	+

Example by: Ameet Soni

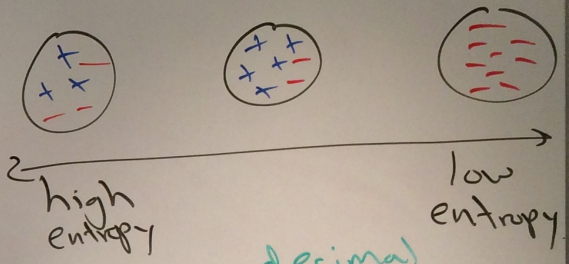
Entropy: avg # bits needed to transmit info

year	prob (p)	idea	(sort) cumulative prob	binary
senior	0.5	0	0	0.0000...
junior	0.25	1	0.5	0.1000...
soph	0.125	01	0.75	0.1100...
first	0.125	10	0.875	0.1110...

110 1101000 1110



intuition



$\square \cdot 2^2 + \square \cdot 2^1 + \square \cdot 2^0 + \square \cdot 2^{-1} + \square \cdot 2^{-2} \ldots$

decimal point

$5 = 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1$

$\Rightarrow 101$

$\frac{1}{2}$ $\frac{1}{4}$