CS 383: Machine Learning

Prof Adam Poliak

Fall 2024

09/11/2024

Lecture 03

Outline

- **Featurization & K-NNs**
- Decision Trees
- Entropy

Predicting Graduation on Time

Features:

- Major:
  - Computer Science, Art History, English
- Dorm:
  - Rhoads, Pembroke, Merion

Major:	{0,1,2}
Dorm:	{0,1,2}

Given a new student, how can we compute distance between her and the training examples if features are categorical?

Distance metric - are all features equal?

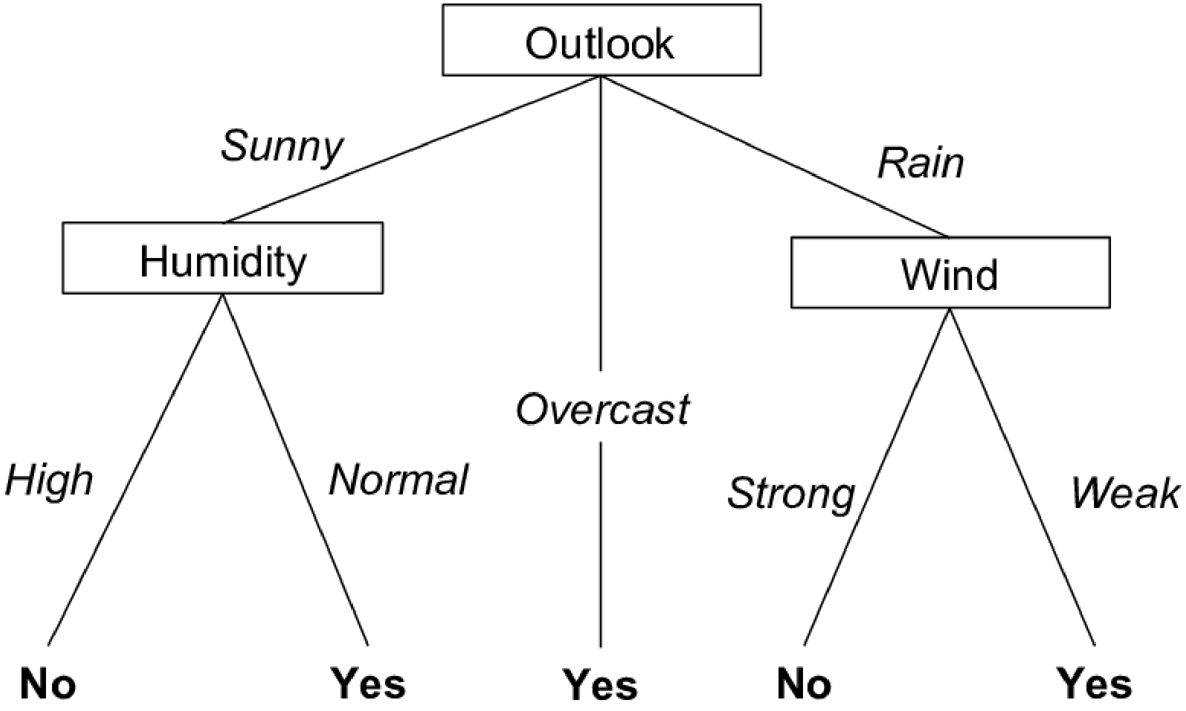
Day	Outlook	Temperature	Humidity	Wind	PlayTennis (y)
$\boldsymbol{x}_1$	Sunny	Hot	High	Weak	No
$\boldsymbol{x}_2$	Sunny	Hot	High	Strong	No
$\boldsymbol{x}_3$	Overcast	Hot	High	Weak	Yes
$\boldsymbol{x}_4$	Rain	Mild	High	Weak	Yes
$\boldsymbol{x}_5$	Rain	Cool	Normal	Weak	Yes
$\boldsymbol{x}_6$	Rain	Cool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

Outline

- Featurization & K-NNs
- **Decision Trees - Overview**
- ID3 Decision Tree Algorithm

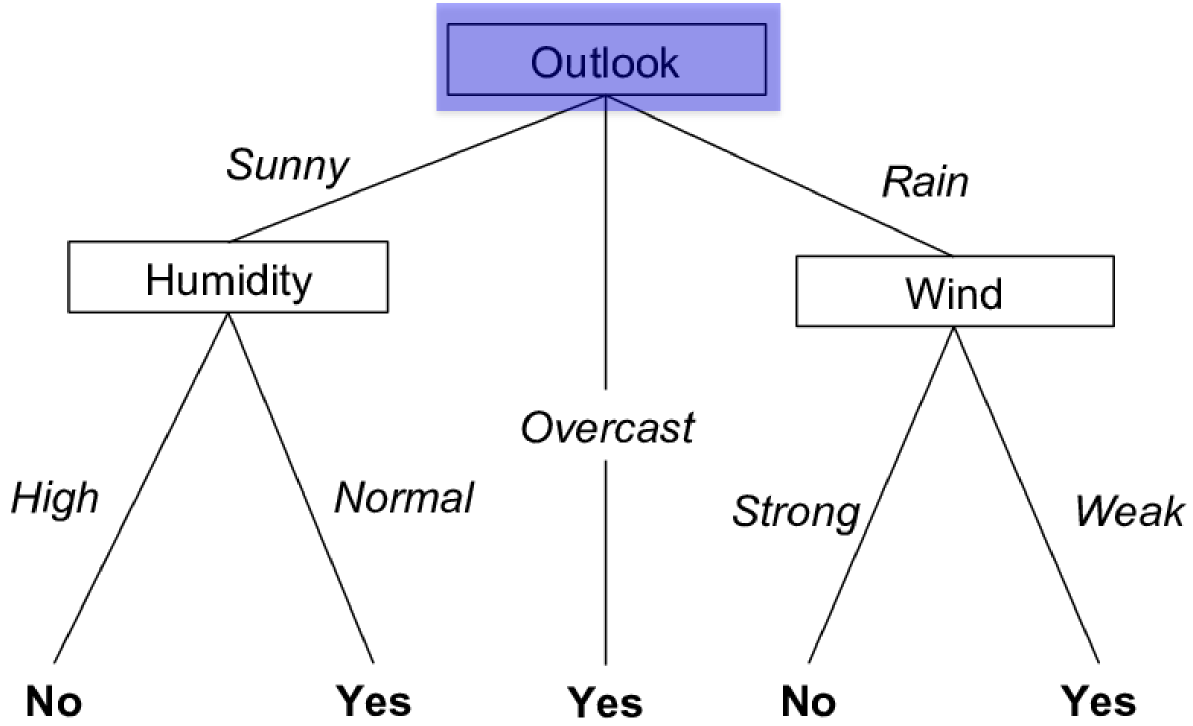
Decision Tree example (tennis)



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

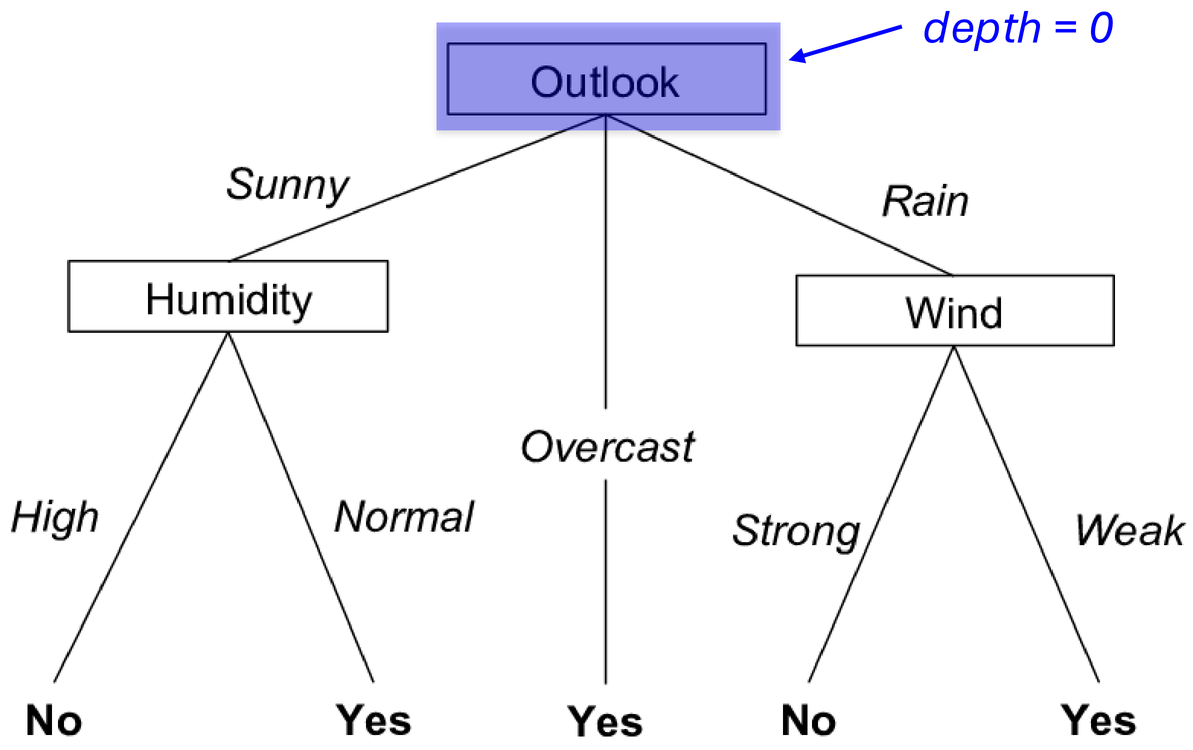
Based on slides by Jessica Wu
and Eric Eaton
[originally by Tom Mitchell]

Decision Tree example (tennis)



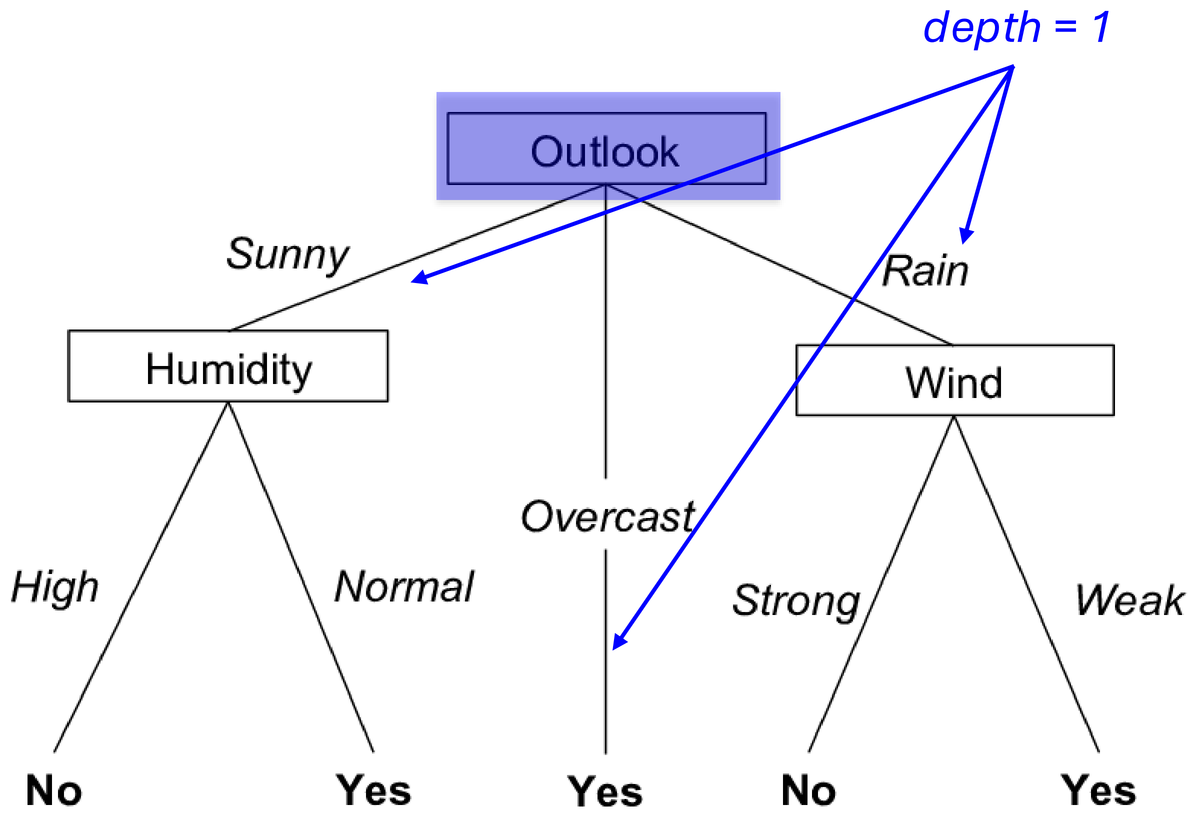
Key term: *depth*

Decision Tree example (tennis)



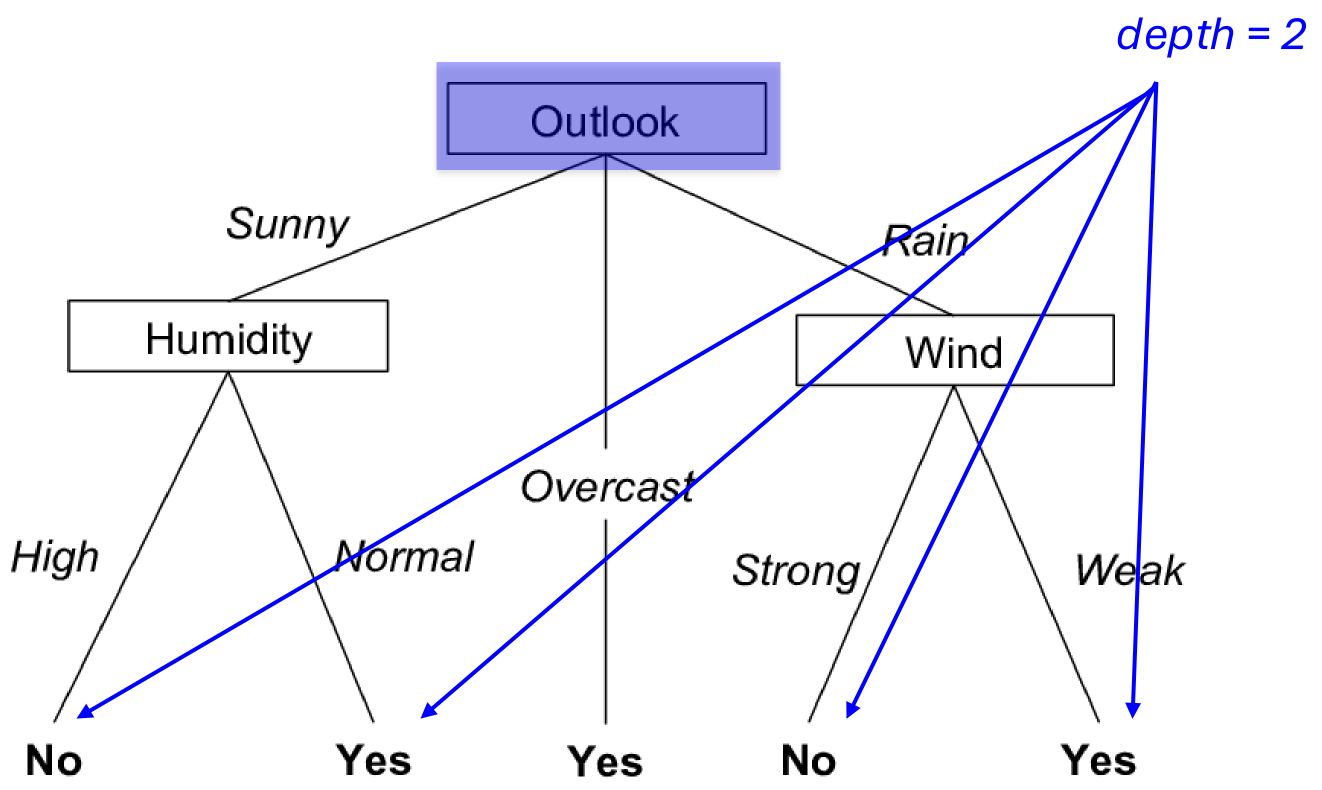
Key term: *depth*

Decision Tree example (tennis)



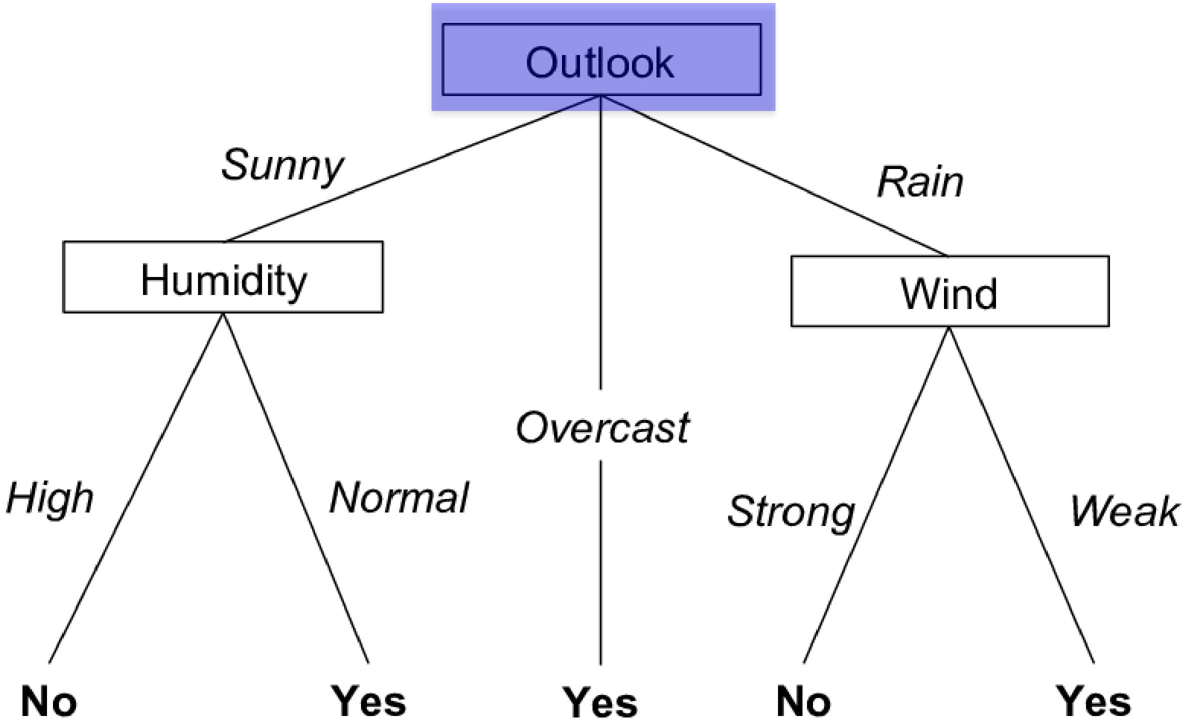
Key term: *depth*

Decision Tree example (tennis)



Key term: *depth*

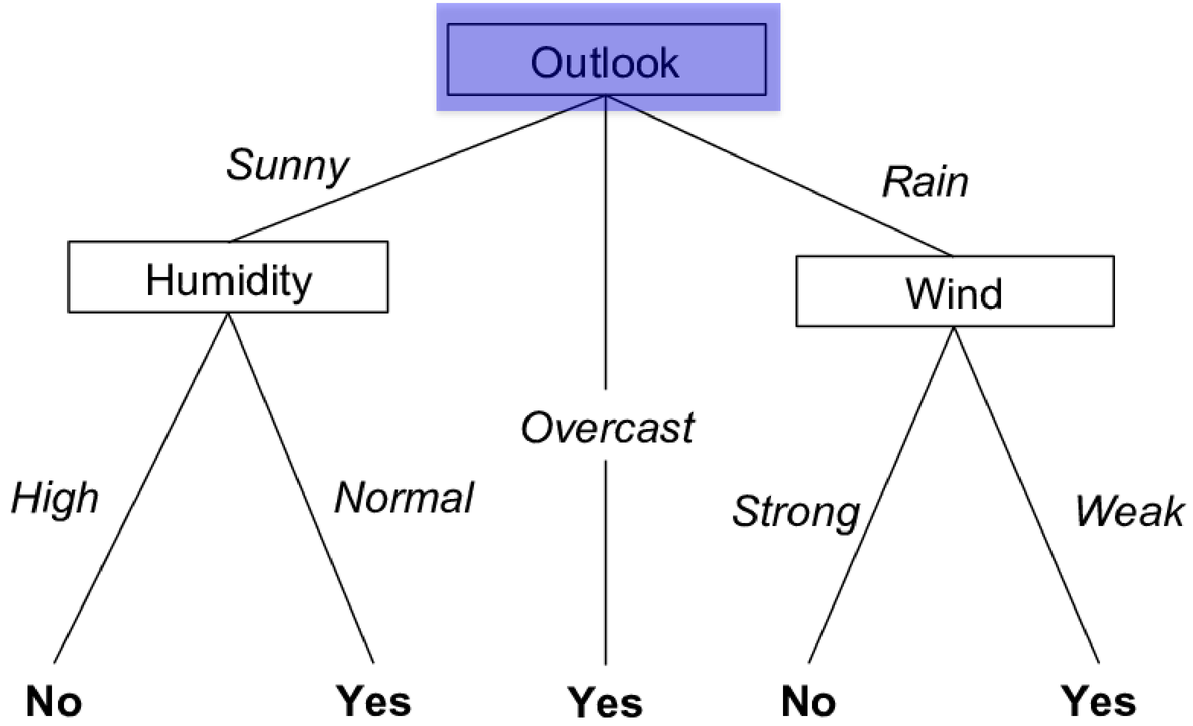
Decision Tree example (tennis)



(test example) $\boldsymbol{x}$ =

Outlook	Temp	Humidity	Wind
Rain	Hot	High	Strong

Decision Tree example (tennis)

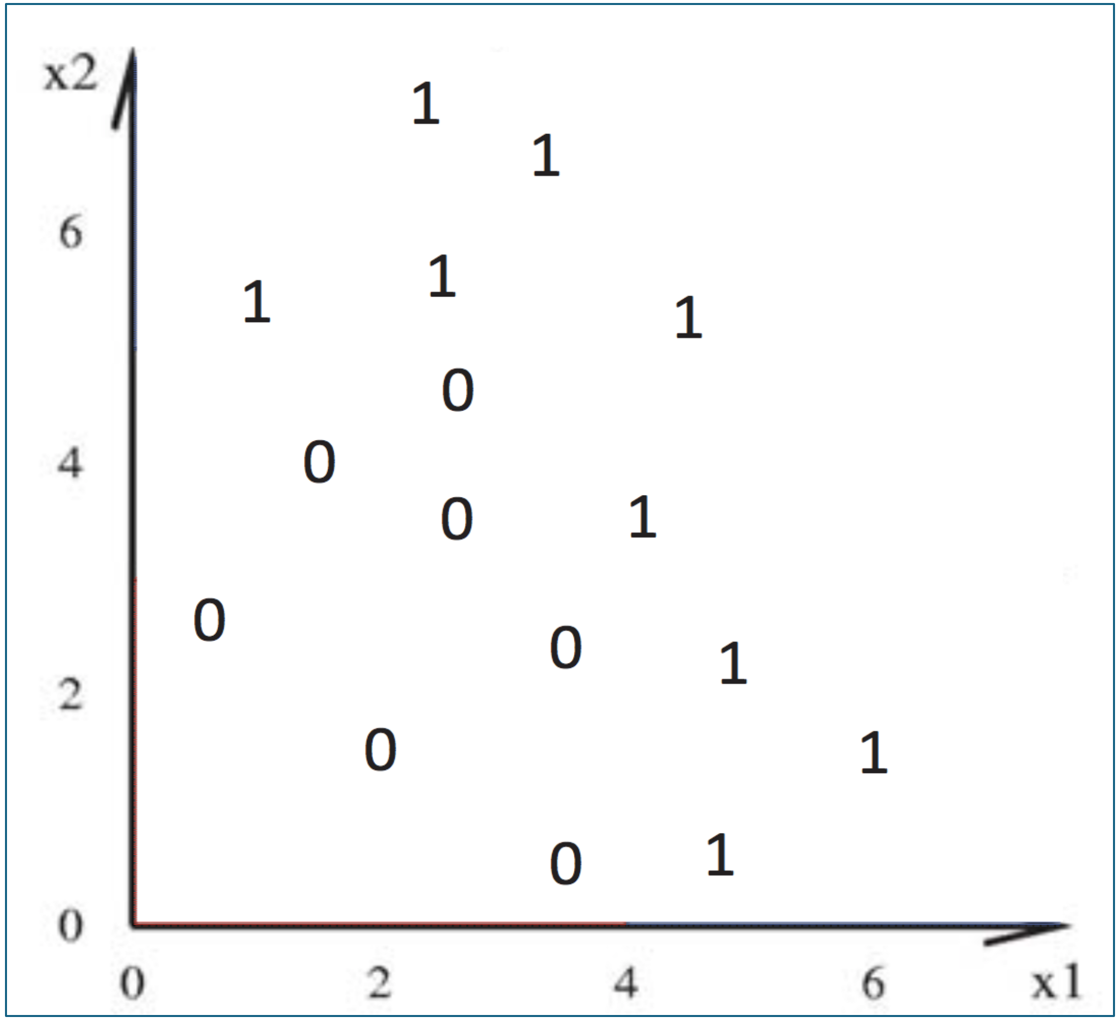


(test example) ***x*** =

Outlook	Temp	Humidity	Wind
Rain	Hot	High	Strong

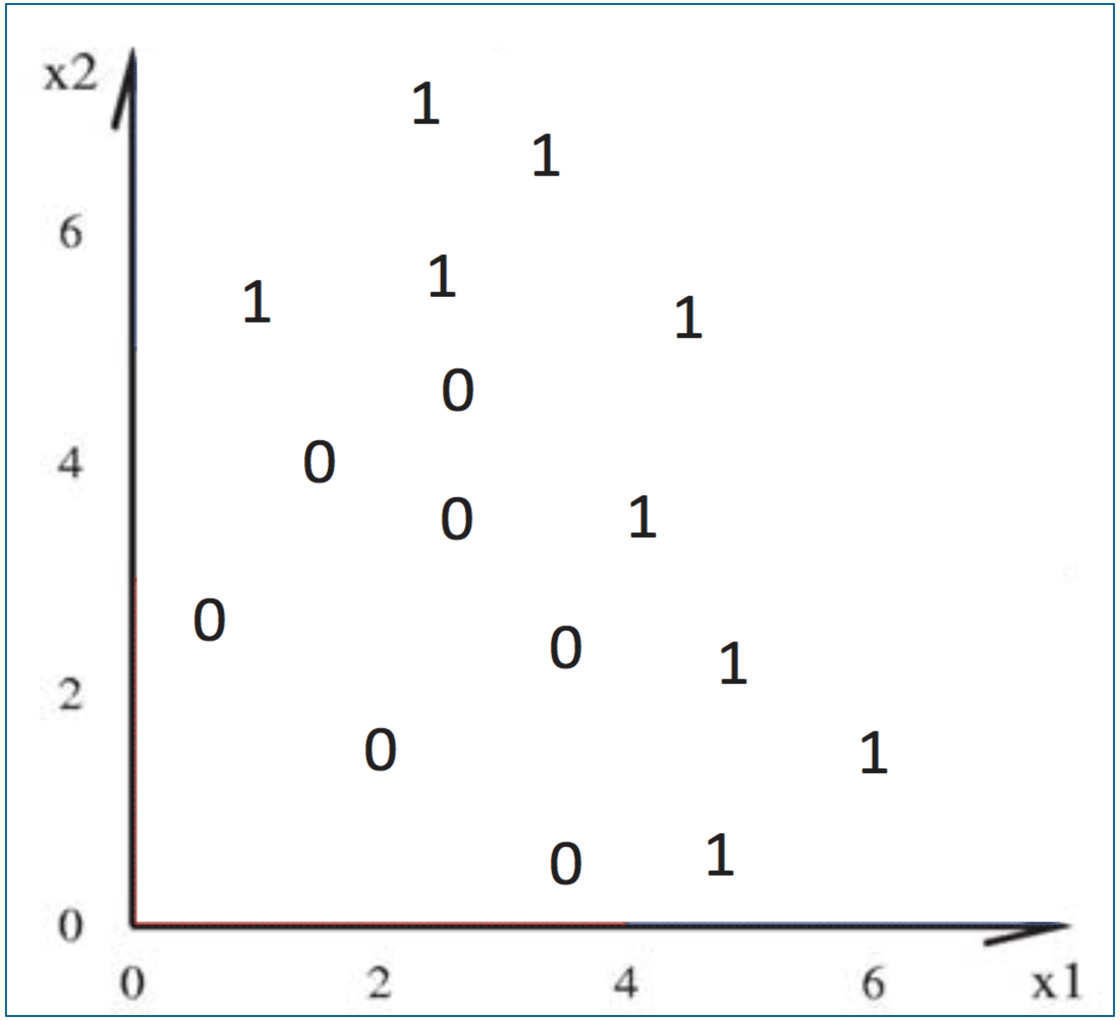
$\hat{y}$ = No

Continuous Features



Example by: Eric Eaton

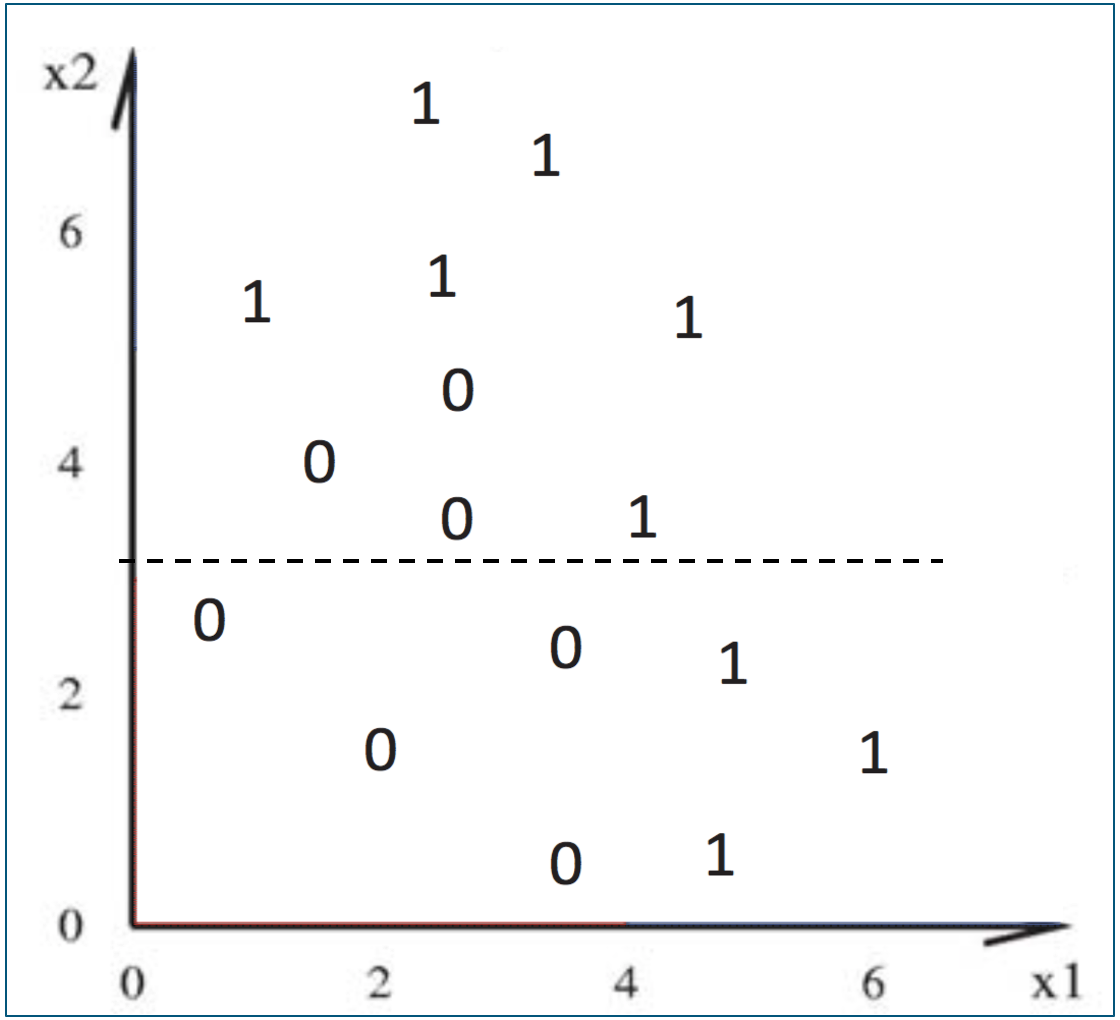
Continuous Features



$\boxed{x_2 \leq 3}$

Example by: Eric Eaton

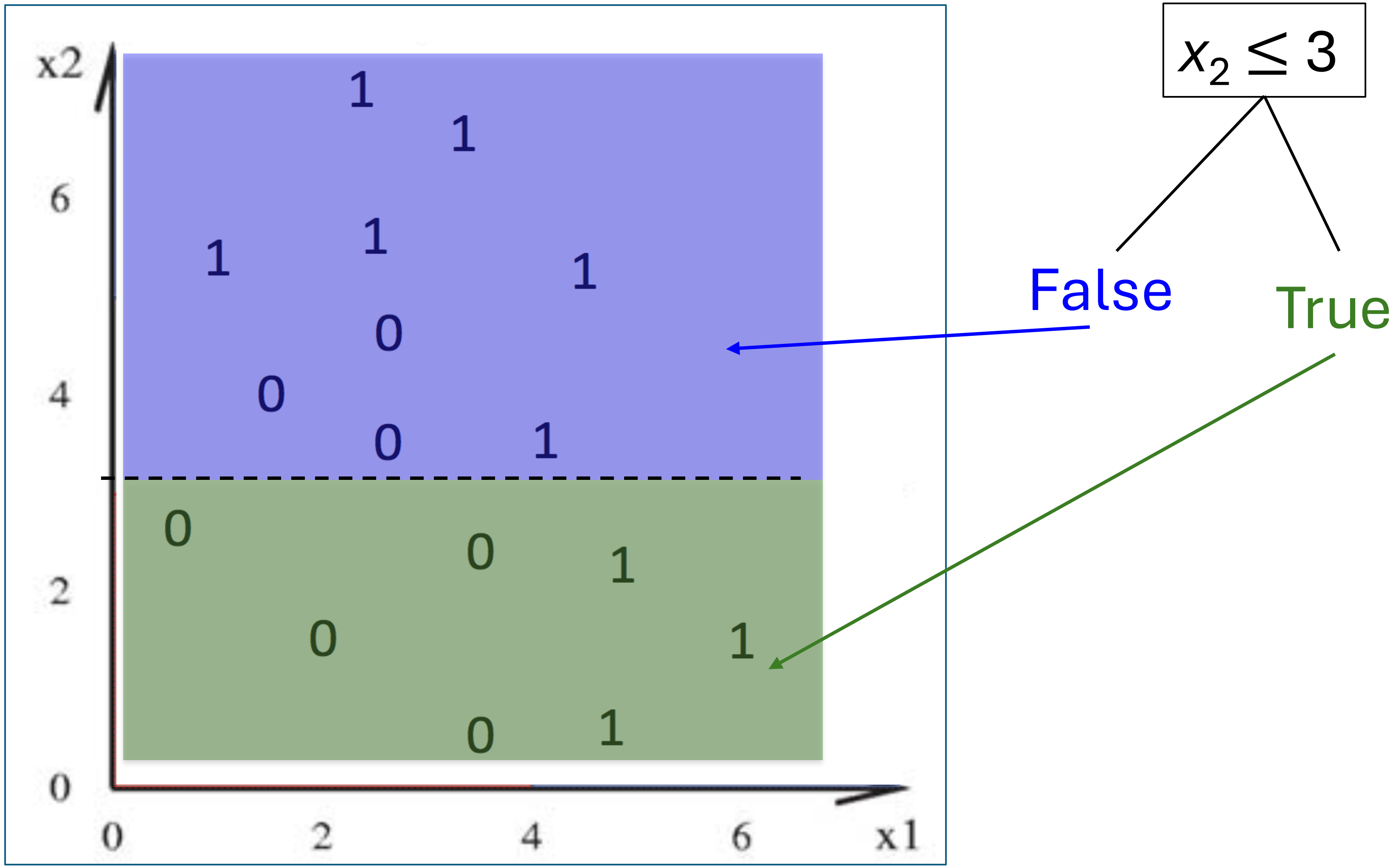
Continuous Features



$x_2 \leq 3$

Example by: Eric Eaton

Continuous Features



Example by: Eric Eaton

Decision Trees: Pros vs Cons

Discuss with a partner! Think about:

* training and testing
* featurization
* runtime
* human factors

Decision Trees: Pros vs 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

- Featurization & K-NNs
- Decision Trees - Overview
- **ID3 Decision Tree Algorithm**

ID3 Decision Tree Algorithm

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
```

Top-Down Decision Tree Algorithm

Dataset (X,y)

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
```

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
          Dk = subset of instances that have outcome k
          N.child[k] = MakeSubtree(Dk,F-S)
  return subtree rooted at 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
  else
      make an internal node N
      S = FindBestFeature(D, F)
      for each outcome k of S
          Dk = subset of instances that have outcome k
          N.child[k] = MakeSubtree(Dk,F-S)
  return subtree rooted at N
```

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

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
```

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

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

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

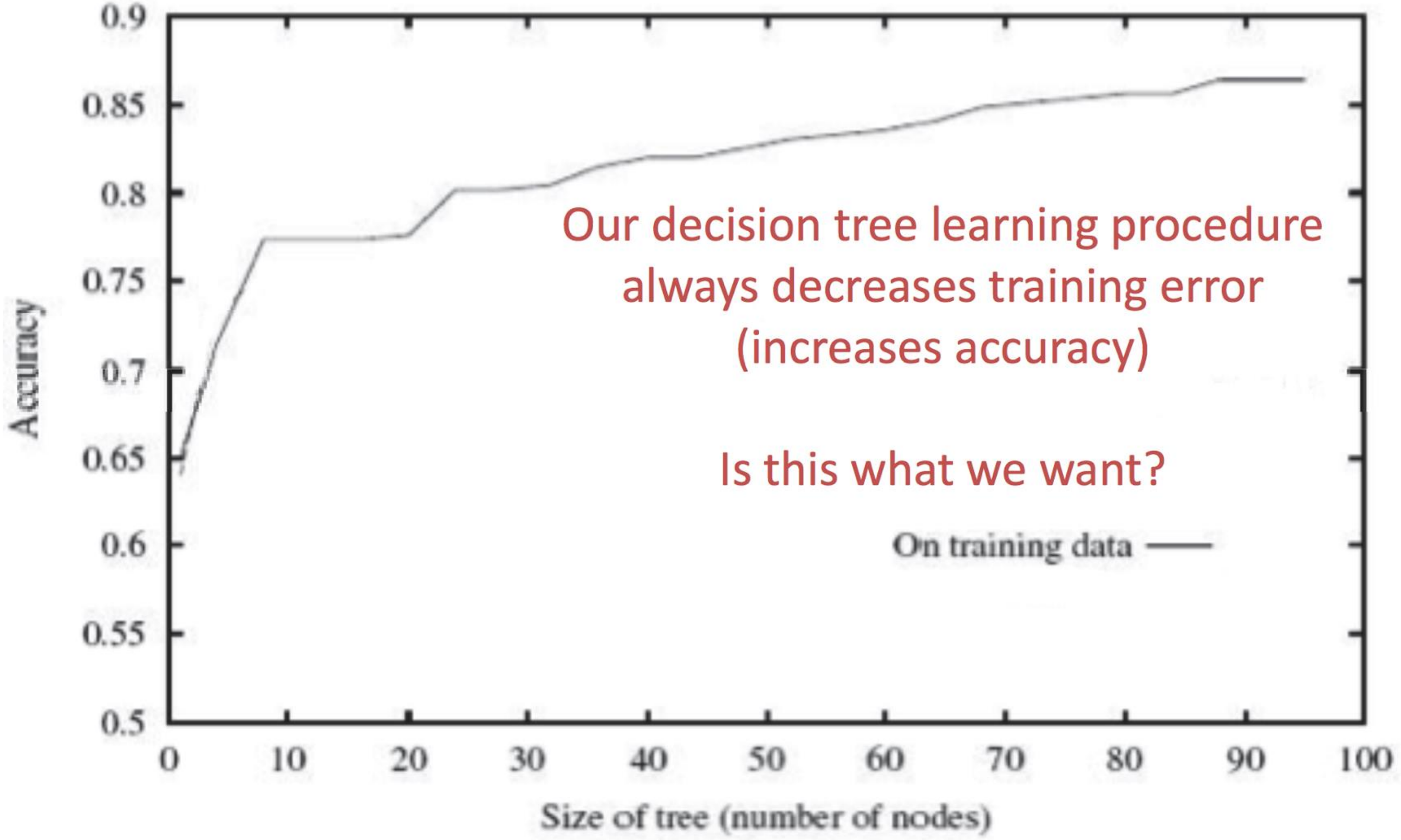
Next class

- “Best feature”

- HW01 due Friday night

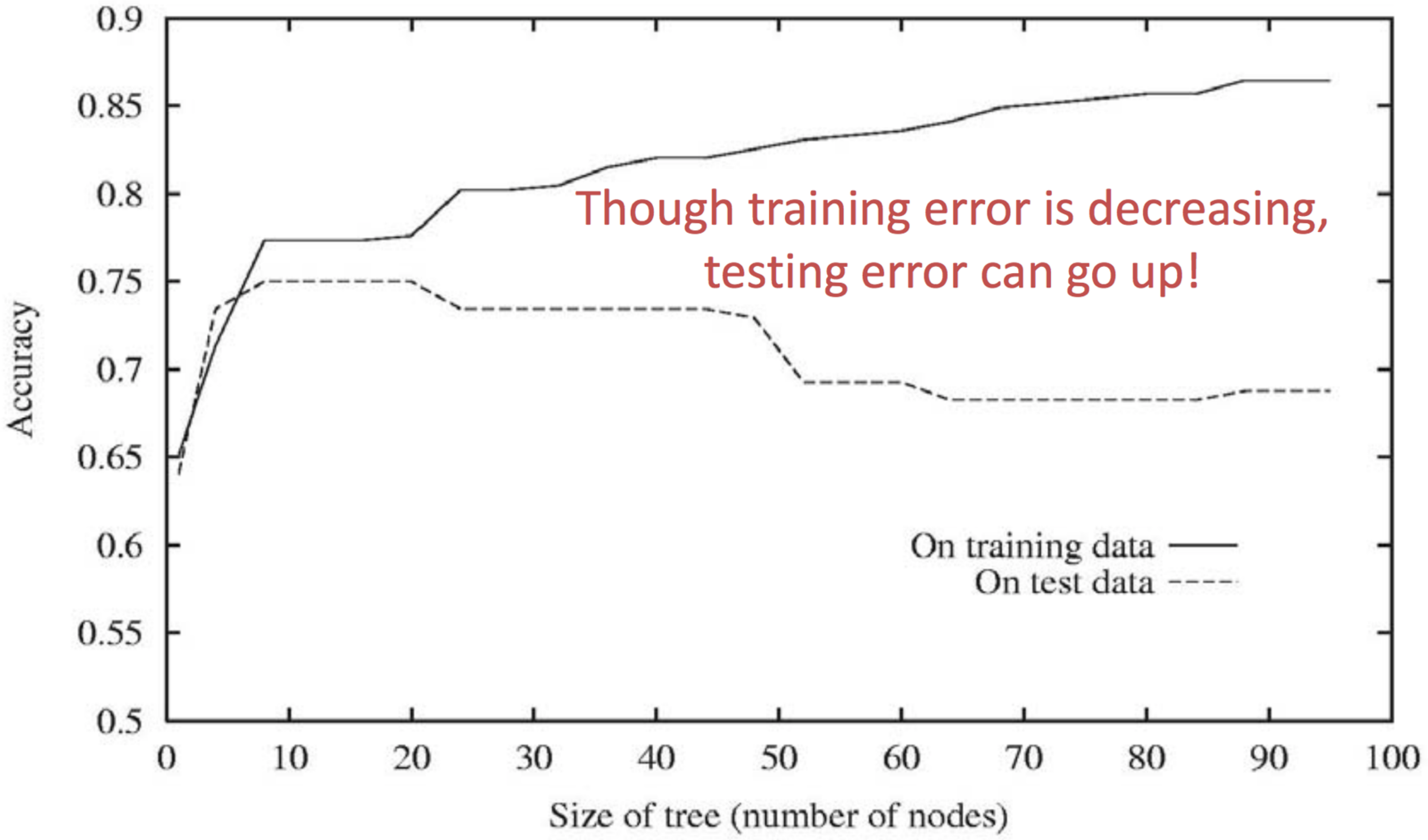
- Thursday lecture back on schedule
    - Reading quiz on Thursday

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: $error_{D}(h)$

Slide: modified from Ameet Soni

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.

$$error_{train}(h) < error_{train}(h') \text{ AND } error_D(h) > error_D(h')$$

Slide: modified from Ameet Soni

Avoiding overfitting in decision trees

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

HW2 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)