Lecture 15: Decision Tree Learning

Core 109S IDWT?, Spring 2017

Michael Hay

Example prediction task

- Task: credit approval
- Input: applicant information
- Output: approve credit?
- Data: applications of past customers

attribute	value
age	23 years
gender	male
salary	\$45,000
years in residence	1
years in job	1
current debt	\$15,000

Credit: Yaser Abu Mostafa https://work.caltech.edu/

Components of learning

Input:	$X = X_1, X_2, \ldots, X_d$	(customer application)
Output:	Y	(approve/deny credit?)
Target function f:	$Y = f(X)$	(ideal credit approval function)
Data:	$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$	(historical records)
	$\downarrow \quad \downarrow \quad \downarrow$	
	$\downarrow \quad \downarrow \quad \downarrow$	
Hypothesis	$g(X)$	(function learned from data)

(ideal credit approval function)

Unknown Target Function

$f(X) \rightarrow Y$

(historical records of customers)

Training Examples

$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$

Learning Algorithm

Final Hypothesis

$g \approx f$

(credit approval function, learned from data)

Hypothesis Set

$\mathcal{H}$

(set of candidate functions)

Credit: Yaser Abu Mostafa https://work.caltech.edu/

Example: perceptron

- Given input $x = (x_1, \ldots, x_d)$ *(the d attributes of customer app.)*

- Assign a **weight** w_i for each attribute value x_i. and choose some **threshold** value

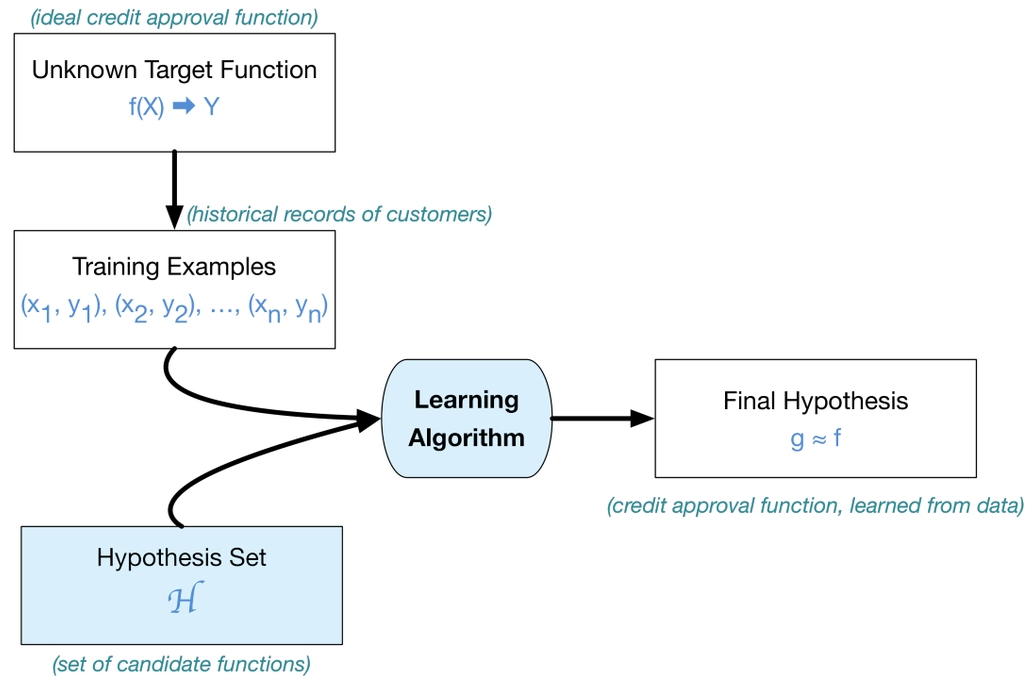
$$\text{Approve credit if} \quad \sum_{i=1}^{d} w_i x_i \geq \text{threshold}$$

$$\text{Deny credit if} \quad \sum_{i=1}^{d} w_i x_i < \text{threshold}$$

Credit: Yaser Abu Mostafa https://work.caltech.edu/

Perceptron = hypothesis set

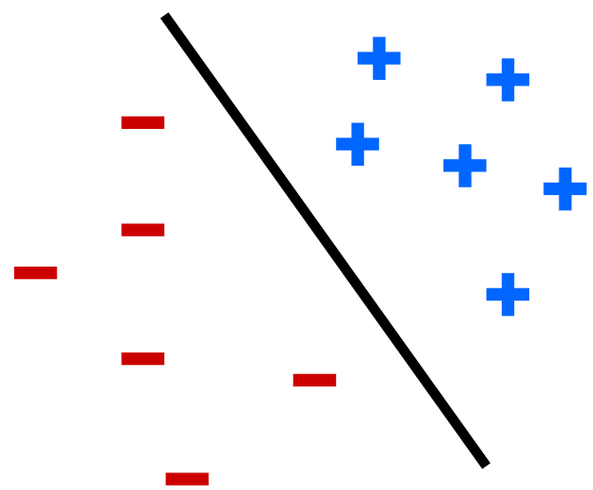
- Solution components:
  - Hypothesis set
  - Learning algorithm



- For **perception**:

$\mathcal{H}$ = all possible settings of weights $w_0, w_1, \ldots w_d$. (infinite set)

Learning algorithm: pick misclassified + update weights



Essence of machine learning

- A pattern exists
- We cannot pin it down mathematically
- We have data on it

Credit: Yaser Abu Mostafa https://work.caltech.edu/

Exercise

Instructions: ~1 minute to think/ answer on your own; then discuss with neighbors; then I will call on one of you

Which of the following problems are best suited for Machine Learning?

A. Classifying numbers into primes and non-primes.

B. Detecting potential fraud in credit card charges.

C. Determining the time it would take a falling object to hit the ground.

D. Determining the optimal cycle for traffic lights in a busy intersection.

E. More than one above

F. None of the above

Parametric vs. Non-parametric

- **Parametric**: means structure of h is fixed except for some parameters. Hypothesis set = { settings of parameters }

- Perceptron
- Linear regression
- Logistic regression
- Naive Bayes

- **Non-parametric**: form of *h* is not fixed in advanced but adapts with data (typically grows more complex with more data)

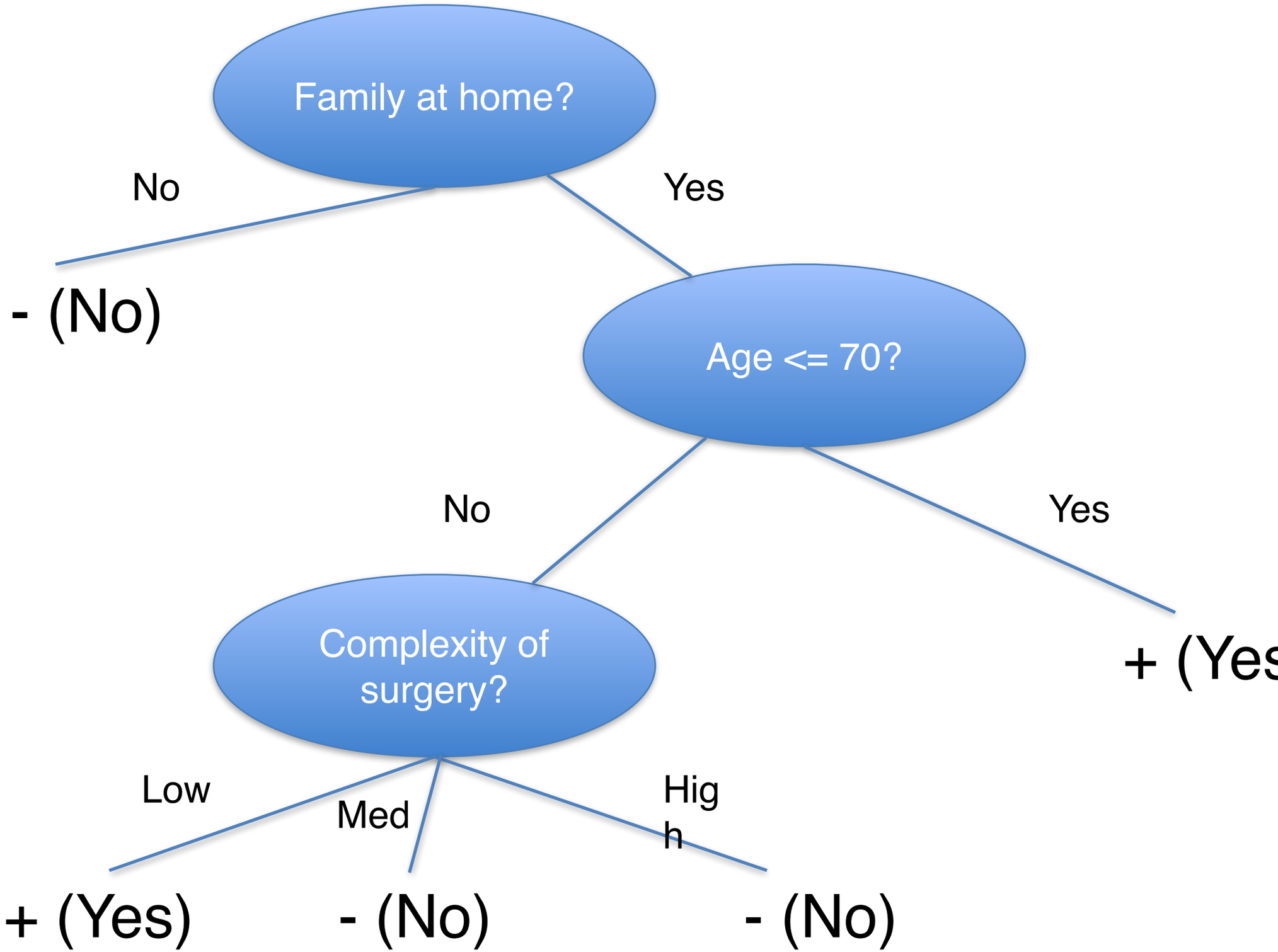
- Decision trees
- K-nearest neighbors

Generally speaking, parametric are more "interpretable" and non-parametric are more "flexible"

Today

- Last time: perceptrons
  - Parametric
  - Expects numerical features
    - Tries find a separating line in feature space
- Today: **Decision trees**
  - Non-parametric
  - Can handle any kind of feature: binary, categorical, numeric
  - Partitions feature space into regions (this will become clear later!)

Decision tree for “Send patient home post-op?”



Learning a decision tree

Input: a collection of *labeled* examples
Output: a decision tree

Key problem: which attribute should be chosen?

If all examples have same value for target attribute*
    The tree is a leaf that stores value of the target attribute
Else
    Pick an attribute for the decision node
    Construct one branch for each possible value of that attribute
    Split examples: each branch gets subset of examples that agree with attribute value associated with that branch
    For each branch, repeat this process on subset of examples assigned to that branch

* we may revise this condition later

Notation

- D is dataset
- $D_{X=x}$ means the subset of data in D where X=x
- X = attribute with k values x1, x2, …, xk
- Y = target attribute
- P(X = x, D) is the fraction of records in D where X=x
- p(x) is short for P(X = x, D) when X and D are clear from context

Entropy-related measures

- entropy(X, D) the entropy of attribute X in dataset D
- segmentEntropy(X, Y, D) the remaining entropy of Y after we segment the data in D on attribute X
- InfoGain(X, Y, D) =
  entropy(Y,D) - segmentEntropy(X, Y, D)
- InfoGain measures how much information about Y is gained when we segment data based on X

Question

Instructions: ~1 minute to think/ answer on your own; then discuss with neighbors; then I will call on one of you

What is entropy(Y, D) where Y is the attribute "send home?"

Question

Instructions: ~1 minute to think/ answer on your own; then discuss with neighbors; then I will call on one of you

What is segmentEntropy(X_1, Y, D) where X_1 is the attribute "family at home?" and Y is "send home?"

Question

Instructions: ~1 minute to think/ answer on your own; then discuss with neighbors; then I will call on one of you

What is InfoGain(X_1, Y, D) where X_1 is the attribute "family at home?" and Y is "send home?"

Question

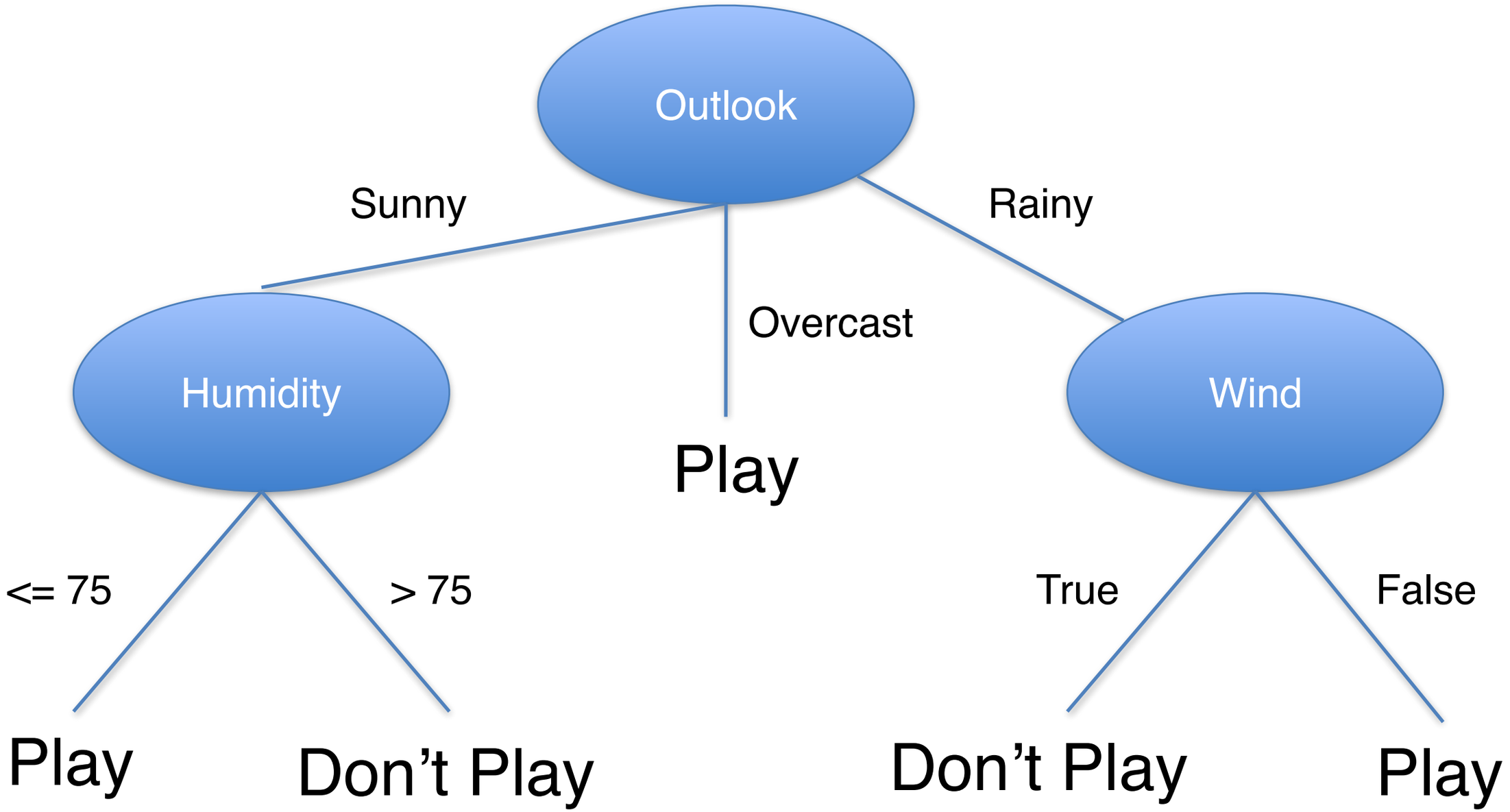
Instructions: ~1 minute to think/ answer on your own; then discuss with neighbors; then I will call on one of you

What is InfoGain(X_3, Y, D) where X_3 is the attribute "age" and Y is "send home?"

Hint: you should be able to answer this without laborious calculation.

Age is a numerical attribute. What does your answer suggest about using InfoGain on numerical attributes?

“Is it a good day to play golf?”



Attributes with Numeric Values: the Golf Tree

	Outlook	Temp	Humidity	Wind	Class
Example1	Sunny	85	85	False	Don't Play
Example2	Sunny	80	90	True	Don't Play
Example3	Overcast	83	88	False	Play
Example4	Rainy	70	96	False	Play
Example5	Rainy	68	80	False	Play
Example6	Rainy	65	70	True	Don't Play
Example7	Overcast	64	65	True	Play
Example8	Sunny	72	95	False	Don't Play
Example9	Sunny	69	70	False	Play
Example10	Rainy	75	80	False	Play
Example11	Sunny	75	70	True	Play
Example12	Overcast	72	90	True	Play
Example13	Overcast	81	75	False	Play
Example14	Rainy	71	96	True	Don't Play

Attributes with Numeric Values

- Split the numeric range into two groups:
  values <= threshold
  values > threshold
- How to select the threshold:
  - Sort the examples by the values of the attribute.
  - Search the examples, noting ***transition points***: places where adjacent examples belong to different classes.
  - The average value at transition points represent ***potential splits***.
  - Evaluate each ***split*** by applying the information gain formula.
  - Choose the best ***split***.
- Compare the gain for the best split against information gain for the remaining attributes.

Attributes with Numeric Values: the Golf Tree

Considering only the examples with Outlook=Sunny

	Humidity	Class
Example9	70	Play
Example11	70	Play
Example1	85	Don’t Play
Example2	90	Don’t Play
Example8	95	Don’t Play

Only one transition point here: 70 to 85, potential split: 77.5, info gain?

Summary

- Key idea behind decision trees is *segmentation:* we are splitting training dataset into subset based on value of a selected attribute

- We can use *information gain* to select the "best" attribute to split on

- For numeric attributes, we can consider "transition points"

- Note: for numeric attributes, it may make sense to split on the same attribute multiple times within the same tree.

- Next time: looking more closely at trees, trees vs. perceptrons