

Lecture 17: Parametric ML (logistic regression) & Non-parametric (trees)

COSC 480 Data Science, Spring 2017
Michael Hay


Today

- Wrap up parametric methods: logistic regression
- Start on non-parametric: decision trees
- Quick note:
 - We're not spending a lot of class time on logistic regression (b/c overlap with perceptron)
 - But it's probably the most useful tool in your ML toolkit!


Noisy targets

- Assumption so far, Y is deterministic function of X
 - $Y = f(X)$
- But sometimes the "target function" is not always a *function*

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



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



- Example: two identical credit applications receive different outcomes (approve/deny)

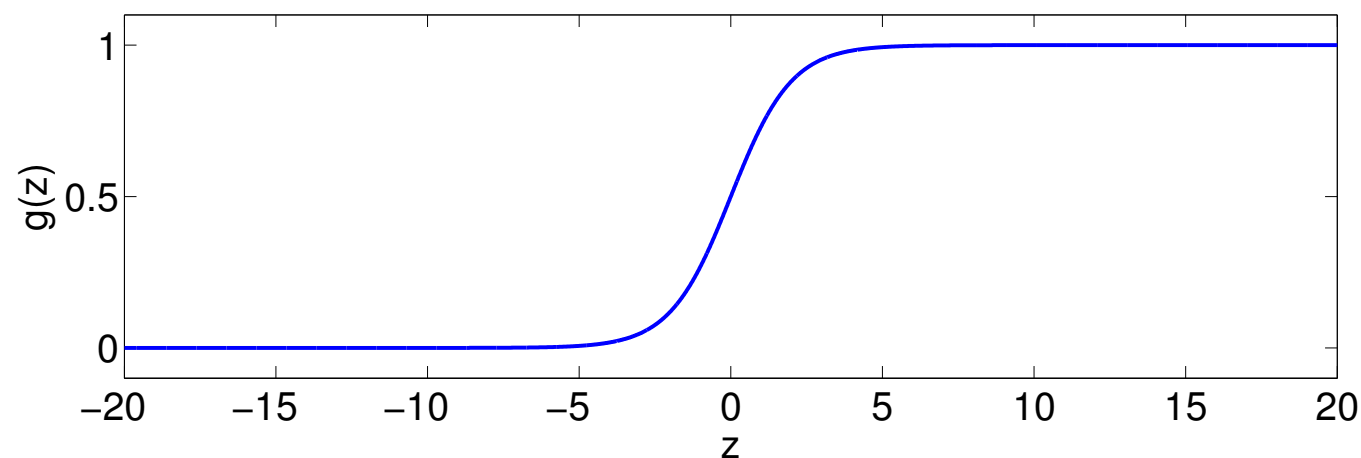
Target *distribution*

- Instead of $Y = f(X)$, treat unknown target function $f(X)$ as a *probability*
- $f(X)$ outputs a number in $[0, 1]$ rather than $\{-1, +1\}$.
 - $P(Y=+1 \mid X) = f(x)$
 - $P(Y=-1 \mid X) = 1-f(x)$
- Machine learning problem: find $h(x) \approx f(x)$
 - Thus, $h(X)$ also outputs a number in $[0, 1]$

Logistic function

- The logistic function maps real numbers to $[0,1]$

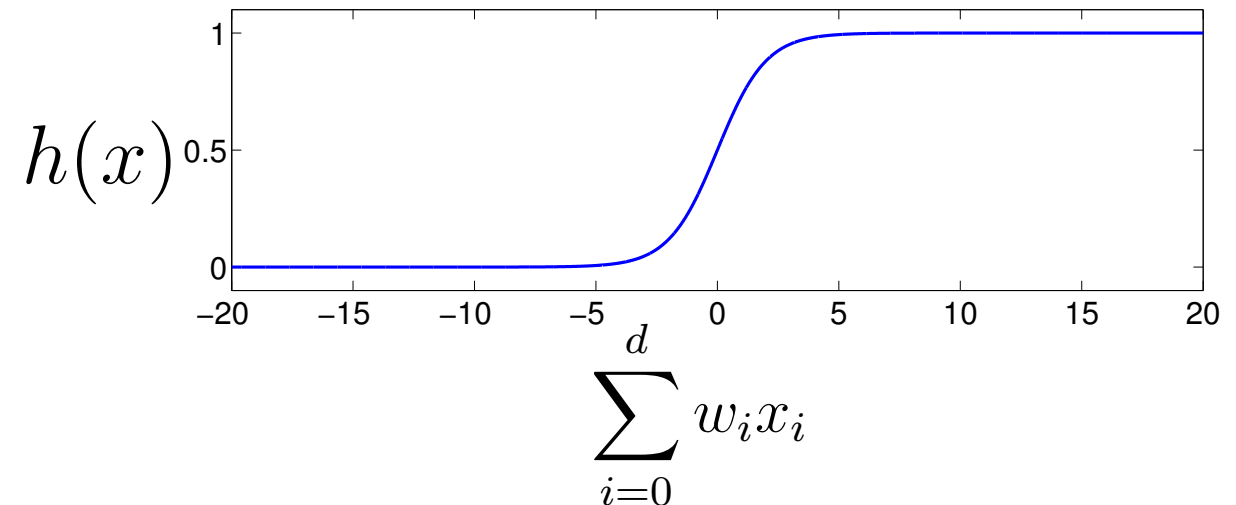
$$g(s) = \frac{e^s}{e^s + 1} = \frac{1}{1 + e^{-s}}$$



Logistic regression

- Take our favorite approach (linear model)

$$s = \sum_{i=0}^d w_i x_i$$



but map onto $[0, 1]$ using logistic function

$$h(x) = g \left(\sum_{i=0}^d w_i x_i \right) = \frac{1}{1 + e^{-\sum_{i=0}^d w_i x_i}}$$

How to set the weights?

- Training data, pairs of (x, y) where y is -1 or $+1$
 - Note: we never see $f(x)$ only the "noisy" output y
 - Think of each y as outcome of coin flip: $+1$ with probability $f(x)$, -1 with probability $1 - f(x)$
- Given data, how set weights w_0, \dots, w_d ?

Maximum likelihood principle: Choose w to maximize probability of producing the training data (assuming $f = h$)

Equation for likelihood

$$P(Y = y|X = x) = \begin{cases} h(x) & \text{if } y = +1 \\ 1 - h(x) & \text{if } y = -1 \end{cases}$$

Substitute $h(x) = g\left(\sum_{j=0}^d w_j x_j\right)$ and fact that $g(-z) = 1 - g(z)$

$$P(Y = y|X = x) = g\left(y \sum_{j=0}^d w_j x_j\right)$$

Note: notation differs from book because book uses $y=0$ or $y=1$

- Likelihood of the data $(x_1, y_1), \dots, (x_n, y_n)$ is

$$\prod_{i=1}^n P(Y_i = y_i|X_i = x_i) = \prod_{i=1}^n g\left(y_i \sum_{j=0}^d w_j x_{ij}\right)$$

Maximize likelihood

- The likelihood is a function of the weights w . Want to choose w to *maximize* likelihood.

$$L(w) = \prod_{i=1}^n P(Y_i = y_i | X_i = x_i) = \prod_{i=1}^n g \left(y_i \sum_{j=0}^d w_j x_{ij} \right)$$

- This is equivalent to maximizing the *log* likelihood.

$$\ell(w) = \log L(w) = \sum_{i=1}^n \log \left(g \left(y_i \sum_{j=0}^d w_j x_{ij} \right) \right)$$

- To turn into *minimization* problem, take negative.

$$J(w) = -\ell(w)$$

We can find minimizing w using gradient descent (details not shown).

Interpretation

- How can we interpret the weights?

- "Log odds"
$$\log \left(\frac{h(x)}{1 - h(x)} \right) = \sum_{j=0}^d w_j x_j$$

- A unit increase in x_i increases the odds by e^{w_i}
- Credit example:
 - Suppose x_i is salary in thousands of dollars and w_i is 0.67
 - Increasing salary by \$1000 increases odds of getting credit by $e^{0.69} \approx 2.0$
 - In other words, \$1000 increase in salary doubles chances of getting credit

Summary of logistic regression

- Widely used
- Compared to perceptron
 - Similarity: linear model
 - Difference: outputs a "probability" rather than decision
- Learning algorithm uses cost function that is...
 - mathematically nice
 - motivated by maximum likelihood principle

break after quick preview



English

Play

Story of Akinator

Akinator on your smartphone

Question N°1

Is your character a female?

Yes

No

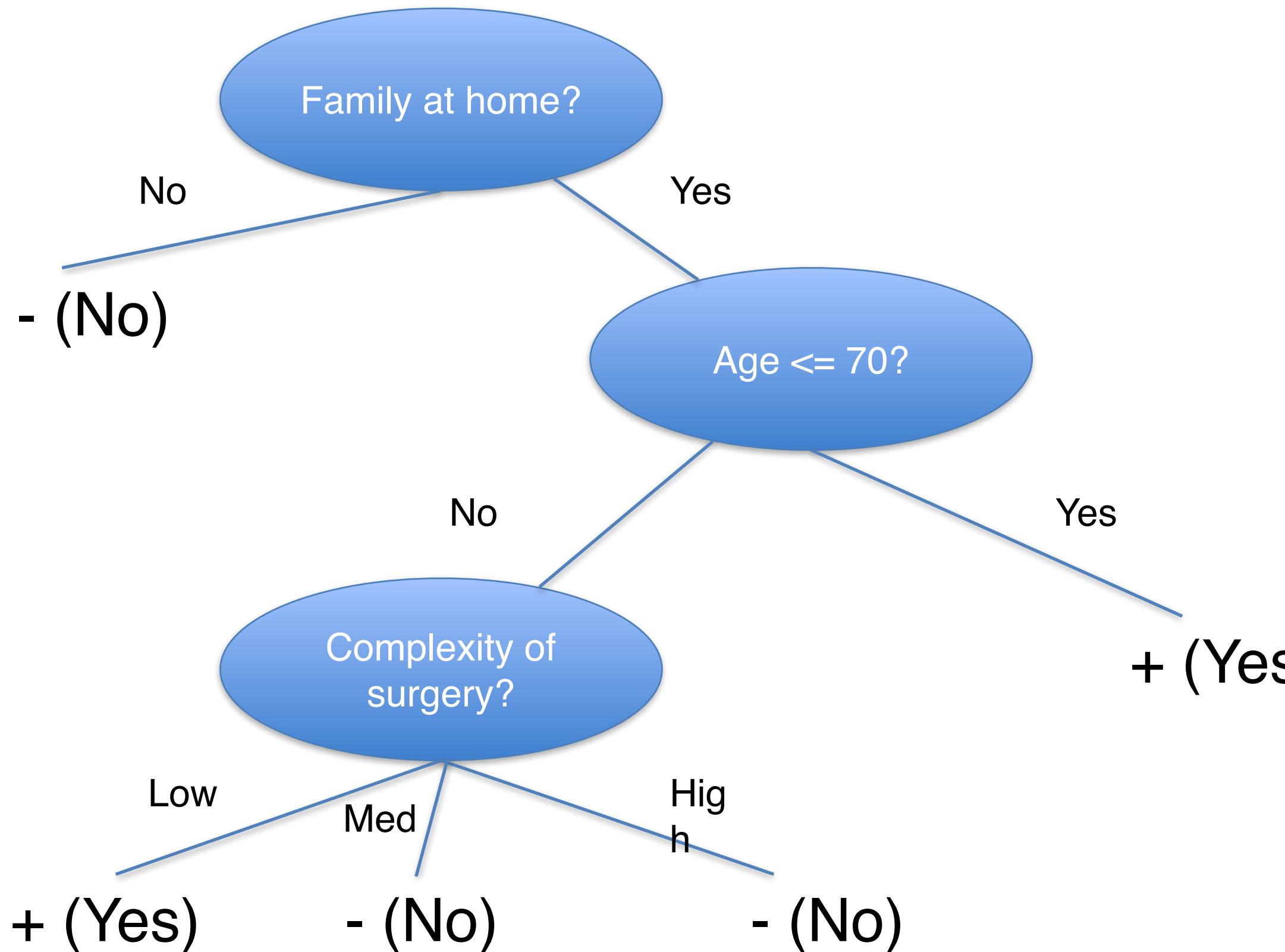
Don't know

Probably

Probably not

Decision Trees

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, recursively build tree on subset of examples assigned to that branch

* and/or other
stopping criteria

Attribute selection

- Intuition: pick an attribute that splits the training data into groups that are *homogenous* in terms of the target attribute
- Example:
 - Suppose we segment on "surgery complexity"
 - Creates three groups (low, med, hi)
 - Check: is low group homogenous in terms of "Send home?" What about the med group? High group?

X₁: Family at home?	X₂: Complexity of surgery	X₃: Age	Y: Send home?
y	low	55	+
y	low	25	+
n	med	72	-
y	med	53	+
n	hi	79	-
n	hi	81	-
n	hi	56	+

Notation

- D is dataset
- $D_{X=x}$ means the subset of data in D where $X=x$
- X = attribute with k values x_1, x_2, \dots, x_k
- 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

- $\text{entropy}(X, D)$ the entropy of attribute X in dataset D
- $\text{segmentEntropy}(X, Y, D)$ the remaining entropy of Y after we segment the data in D on attribute X
- $\text{InfoGain}(X, Y, D) =$
 $\text{entropy}(Y, D) - \text{segmentEntropy}(X, Y, D)$
- 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 $\text{entropy}(Y, D)$ where Y is the attribute "send home?"

It is sufficient to write a math expression.

X_1 : Family at home?	X_2 : Complexity of surgery	X_3 : Age	Y : Send home?
y	low	55	+
y	low	25	+
n	med	72	-
y	med	53	+
n	hi	79	-
n	hi	81	-
n	hi	56	+

Question

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

What is $\text{segmentEntropy}(X_1, Y, D)$ where X_1 is the attribute "family at home?" and Y is "send home?"

It is sufficient to write a math expression.

X_1 : Family at home?	X_2 : Complexity of surgery	X_3 : Age	Y : Send home?
y	low	55	+
y	low	25	+
n	med	72	-
y	med	53	+
n	hi	79	-
n	hi	81	-
n	hi	56	+

Entropy-related measures

- $\text{InfoGain}(X, Y, D) = \text{entropy}(Y, D) - \text{segmentEntropy}(X, Y, D)$
- InfoGain measures how much information about Y is gained when we segment data based on X
- We should pick the attribute X that maximizes InfoGain
 - Note: $\text{entropy}(Y, D)$ is the same for all X , so equivalent to minimizing segmentEntropy

Question

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

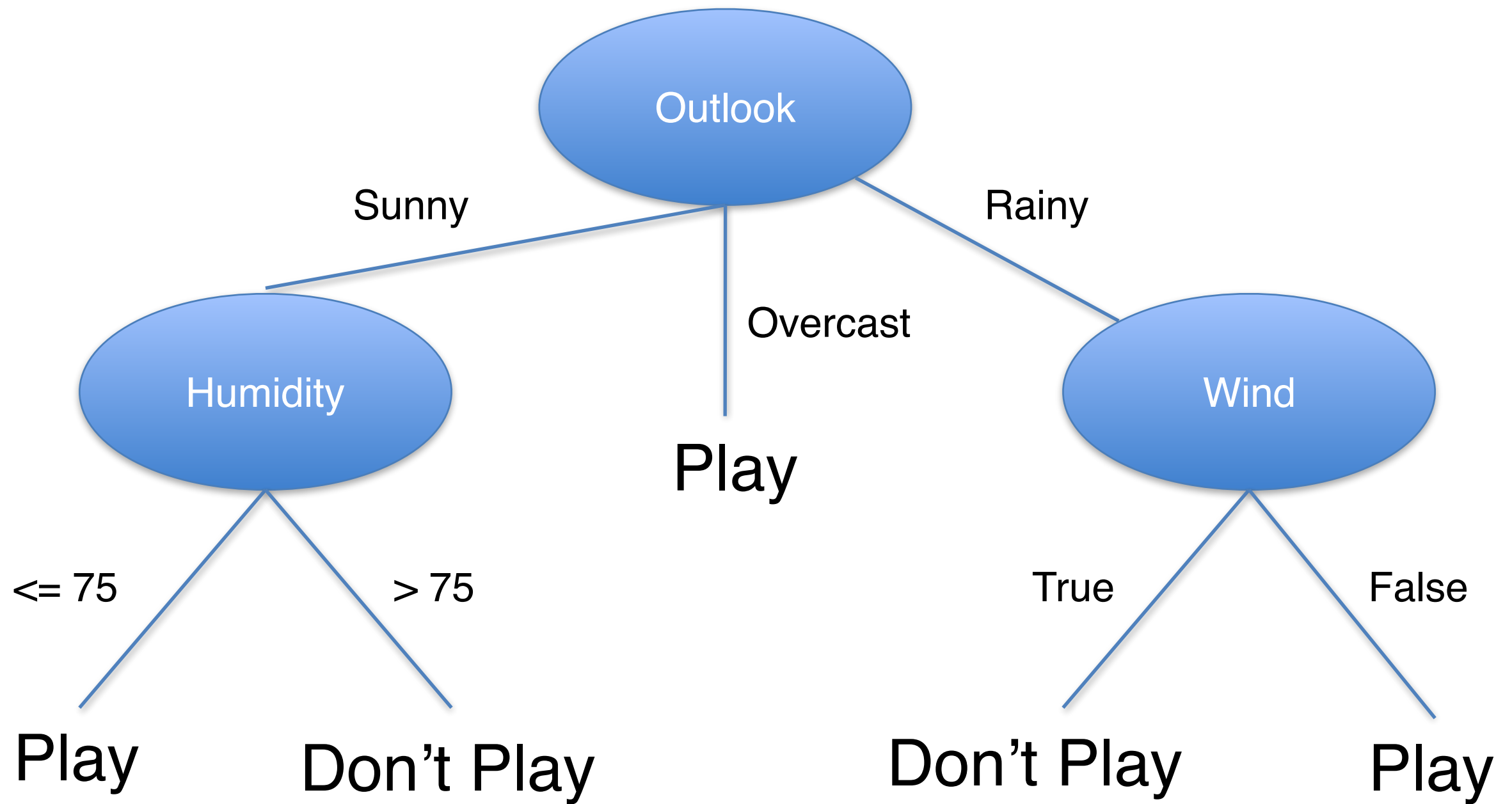
What is $\text{InfoGain}(X_3, Y, D)$ where X_3 is the attribute "age" and Y is "send home?"

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?

X_1 : Family at home?	X_2 : Complexity of surgery	X_3 : Age	Y : Send home?
y	low	55	+
y	low	25	+
n	med	72	-
y	med	53	+
n	hi	79	-
n	hi	81	-
n	hi	56	+

“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 \leq 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.
- May split on same numeric attribute more than once.

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?

Question

	Temp	Class
Example7	64	Play
Example6	65	Don't Play
Example5	68	Play
Example9	69	Play
Example4	70	Play
Example14	71	Don't Play
Example8	72	Don't Play
Example12	75	Play
Example10	75	Play
Example11	75	Play
Example2	80	Don't Play
Example13	81	Play
Example3	83	Play
Example1	85	Play

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

What are the
potential splits for
Temp?

- Search the examples, noting ***transition points***: places where adjacent examples belong to different classes.
- The average value at transition points represent ***potential splits***.

Question

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

The first attribute selected for a decision node is the attribute with the highest information gain.

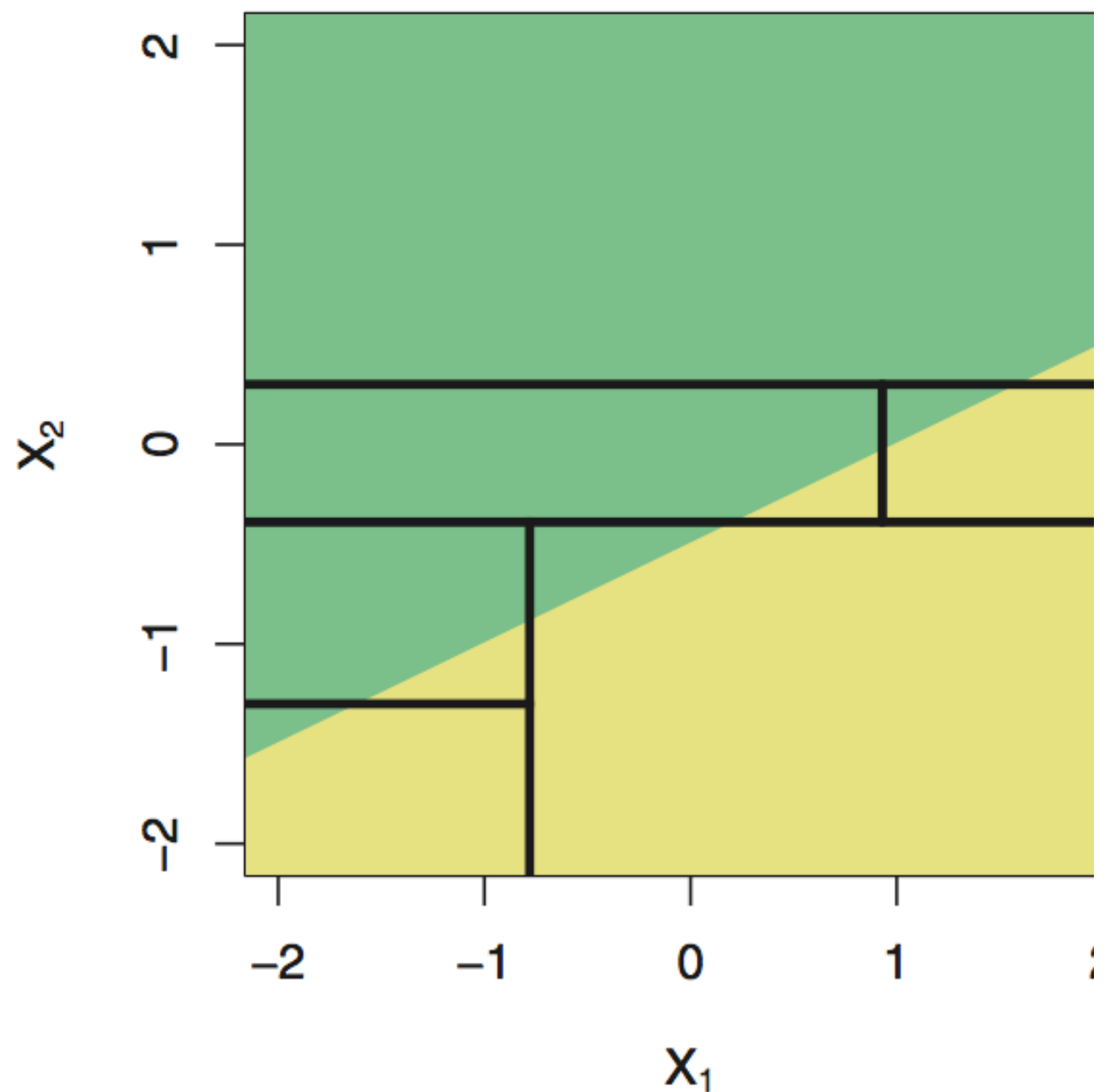
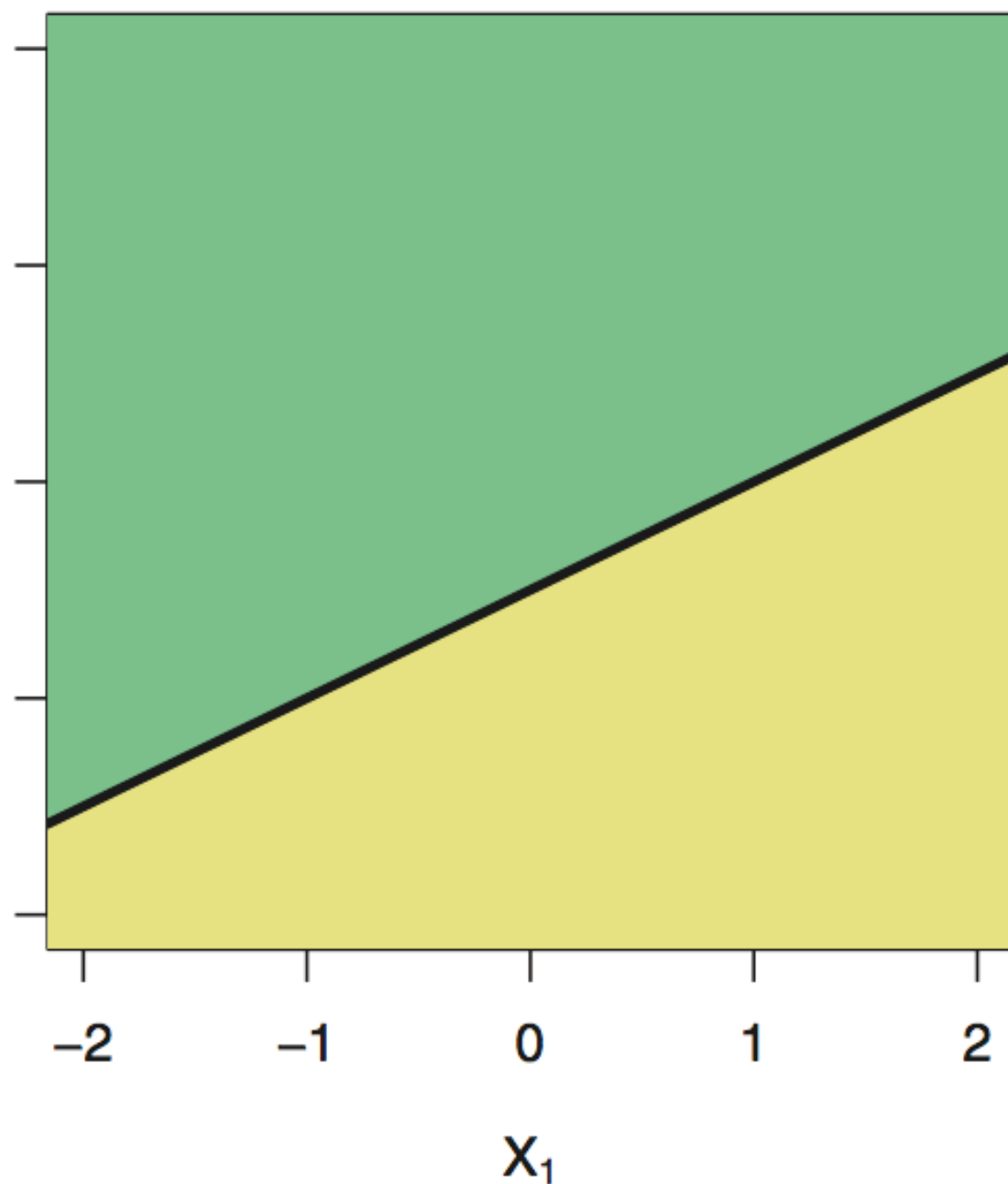
Claim: The second attribute selected for a decision node will be the attribute with the second highest information gain.

A. True

B. False

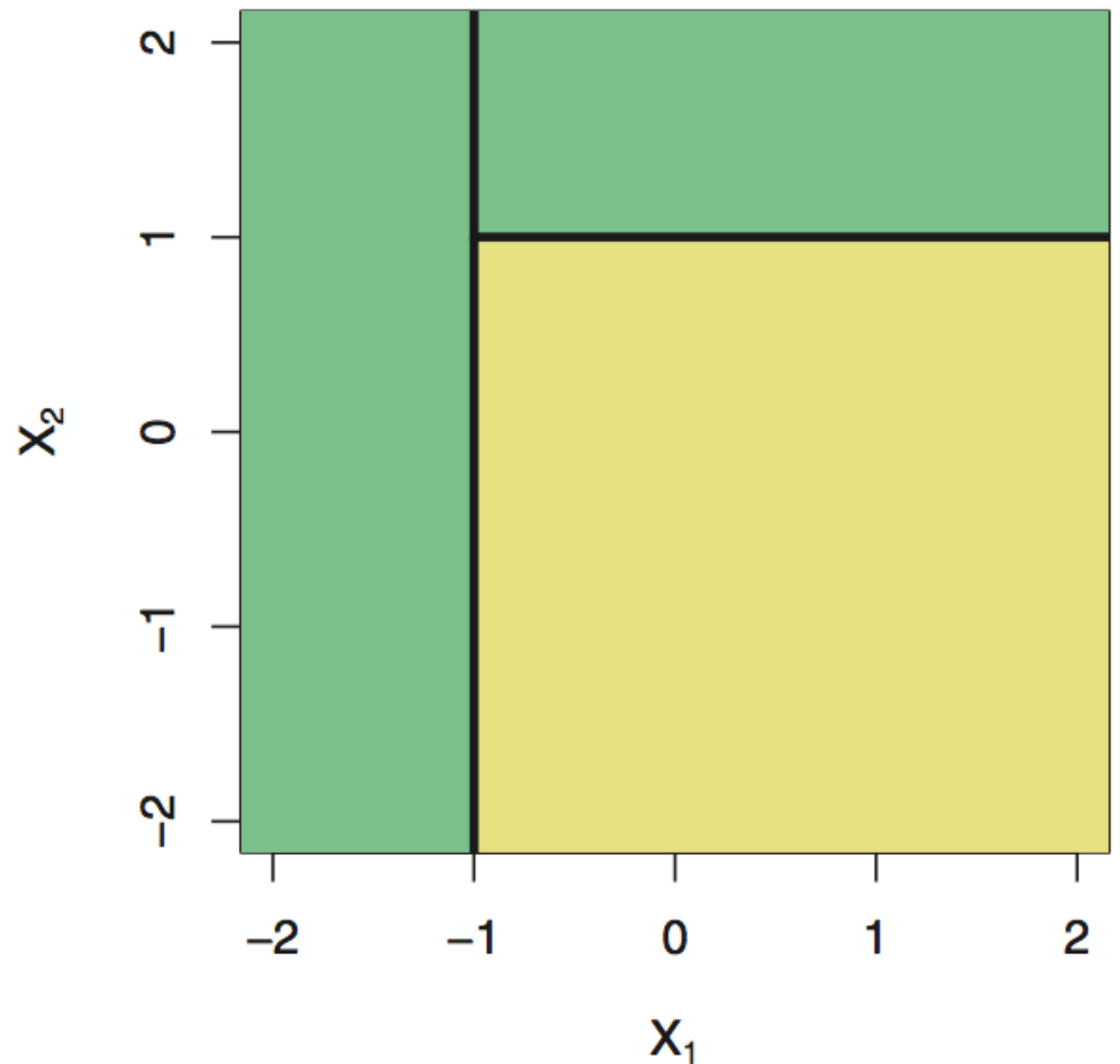
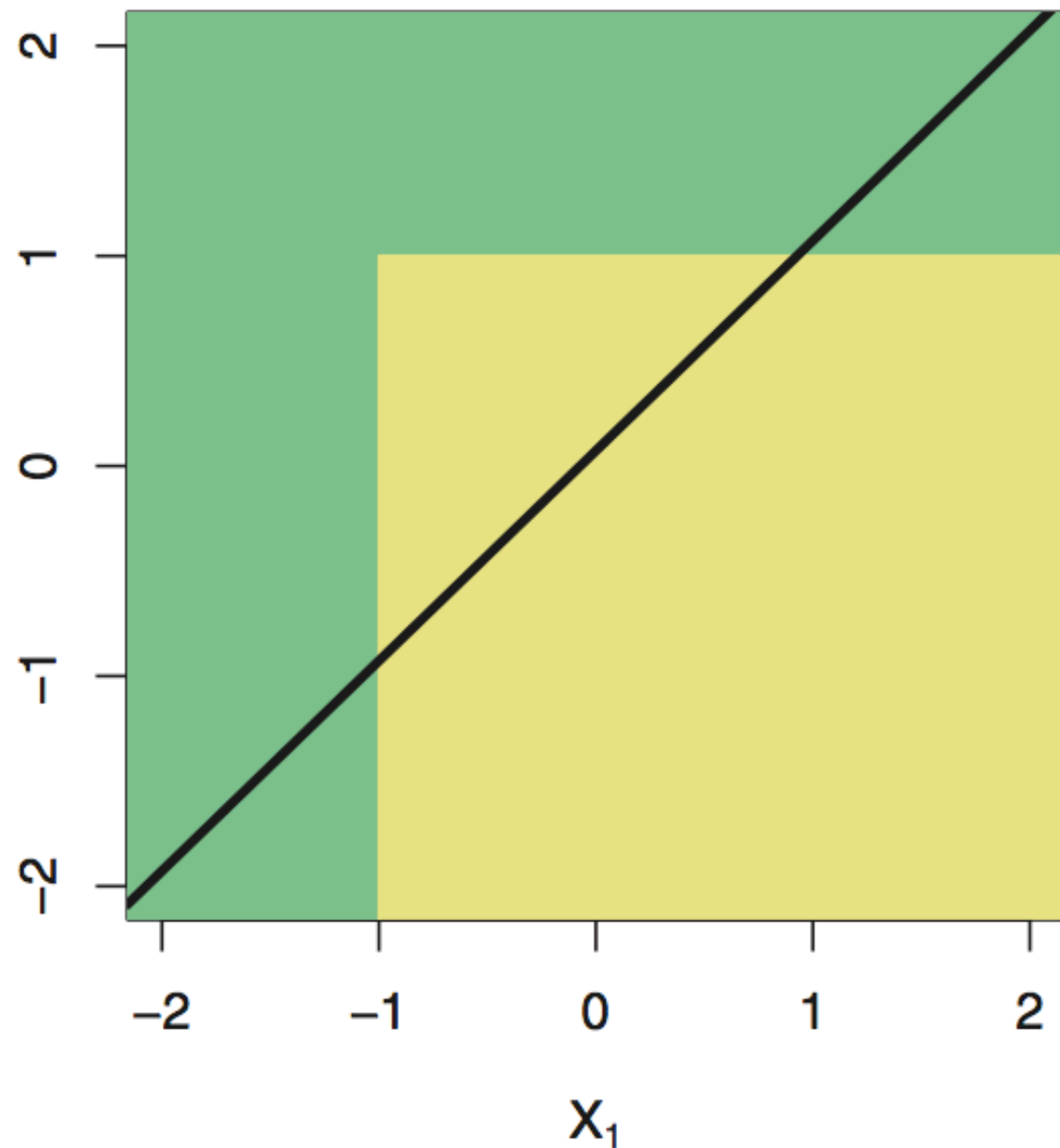
C. Not enough information given in problem

Linear vs. trees



True boundary is linear; tree can only approximate with axis-parallel splits

Linear vs. trees



True boundary is non-linear: linear model (perceptron) gives poor approximation

Comparison

Perceptron

Parameteric

Numerical features

Finds separating line

"Robust": similar line on two different samples of data

Decision Tree

Non-parametric

All feature types

Partitions space into regions (finer with more data)

Non-robust: shape of tree can change a lot on different samples

- See ISL p. 315 for nice summary of pros/cons of trees.

Summary of trees

- 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.