Massively Parallel Algorithms
Classification & Prediction Using Random Forests

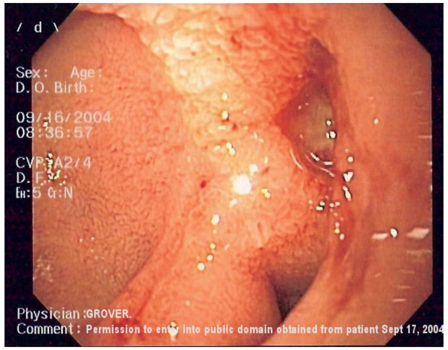
G. Zachmann
University of Bremen, Germany
cgvr.cs.uni-bremen.de

Classification Problem Statement

- Given a set of points $\mathcal{L} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \in \mathbb{R}^d$ and for each such point a label $y_i \in \{l_1, l_2, \ldots, l_n\}$
  - Each label represents a class, all points with the same label are in the same class
- Wanted: a method to decide for a *not-yet-seen* point $\mathbf{x}$ which label it most probably has, i.e., a method to *predict class labels*
  - We say that we learn a classifier C from the training set $\mathcal{L}$:

$$C : \mathbb{R}^d \to \{l_1, l_2, \ldots, l_n\}$$

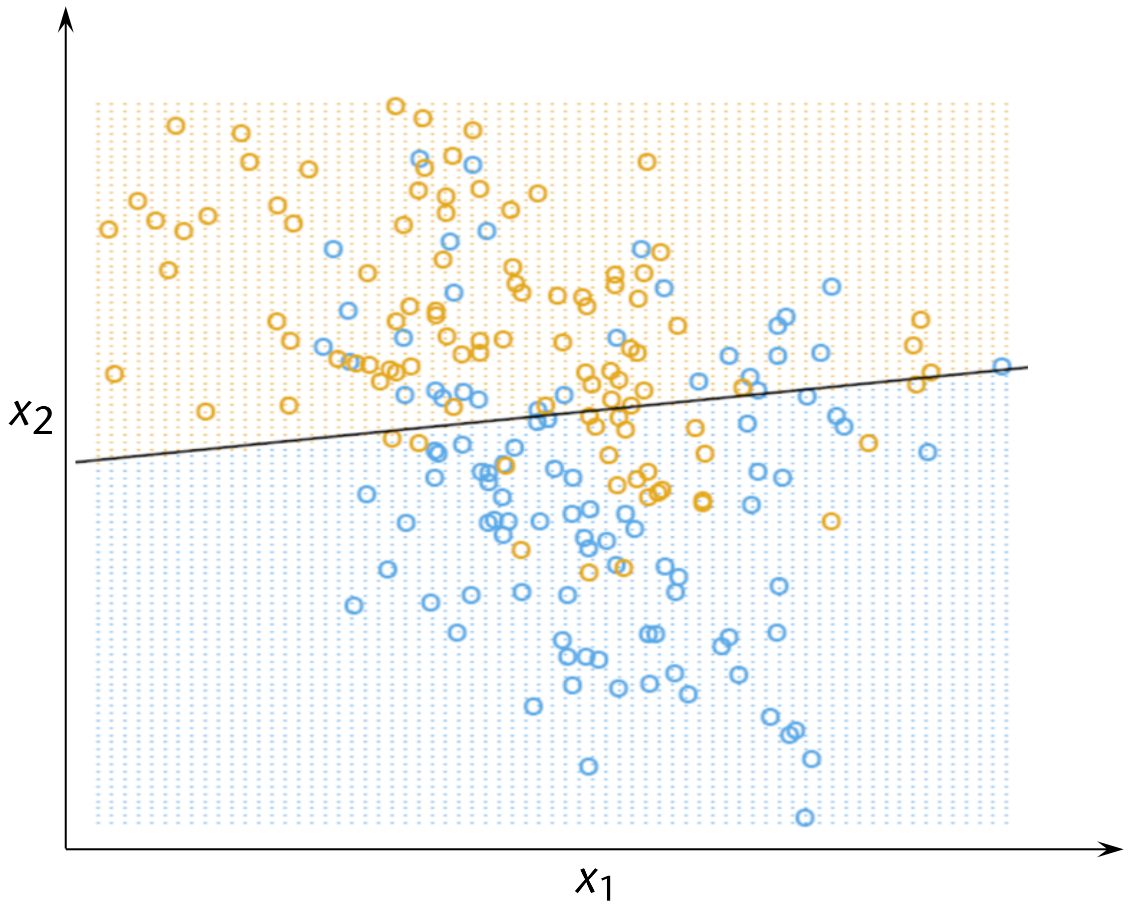
- Typical applications:
  - Computer vision (object recognition, ...)
  - Credit approval
  - Medical diagnosis
  - Treatment effectiveness analysis



Ulcer/tumor or not?

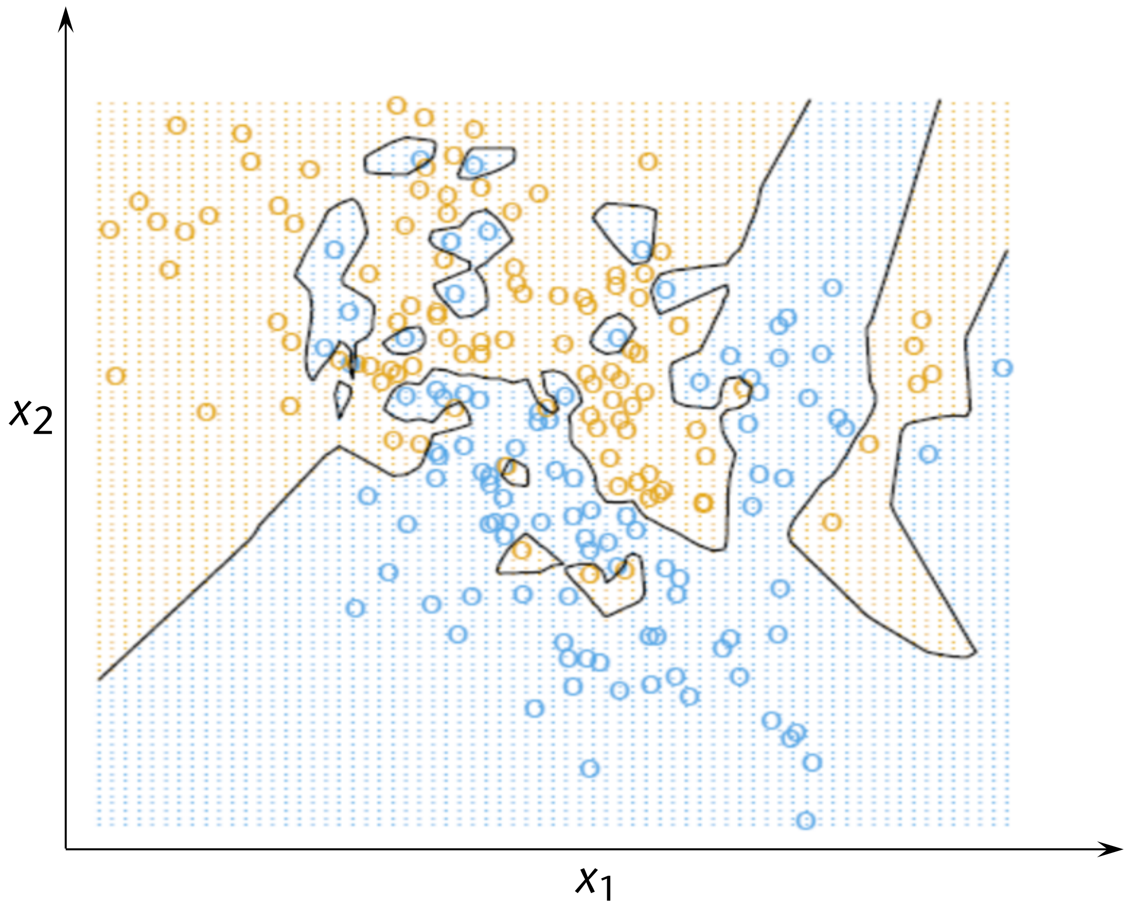
One Possible Solution: Linear Regression

- Assume we have only two classes (e.g., "blue" and "yellow")
- Fit a plane through the data



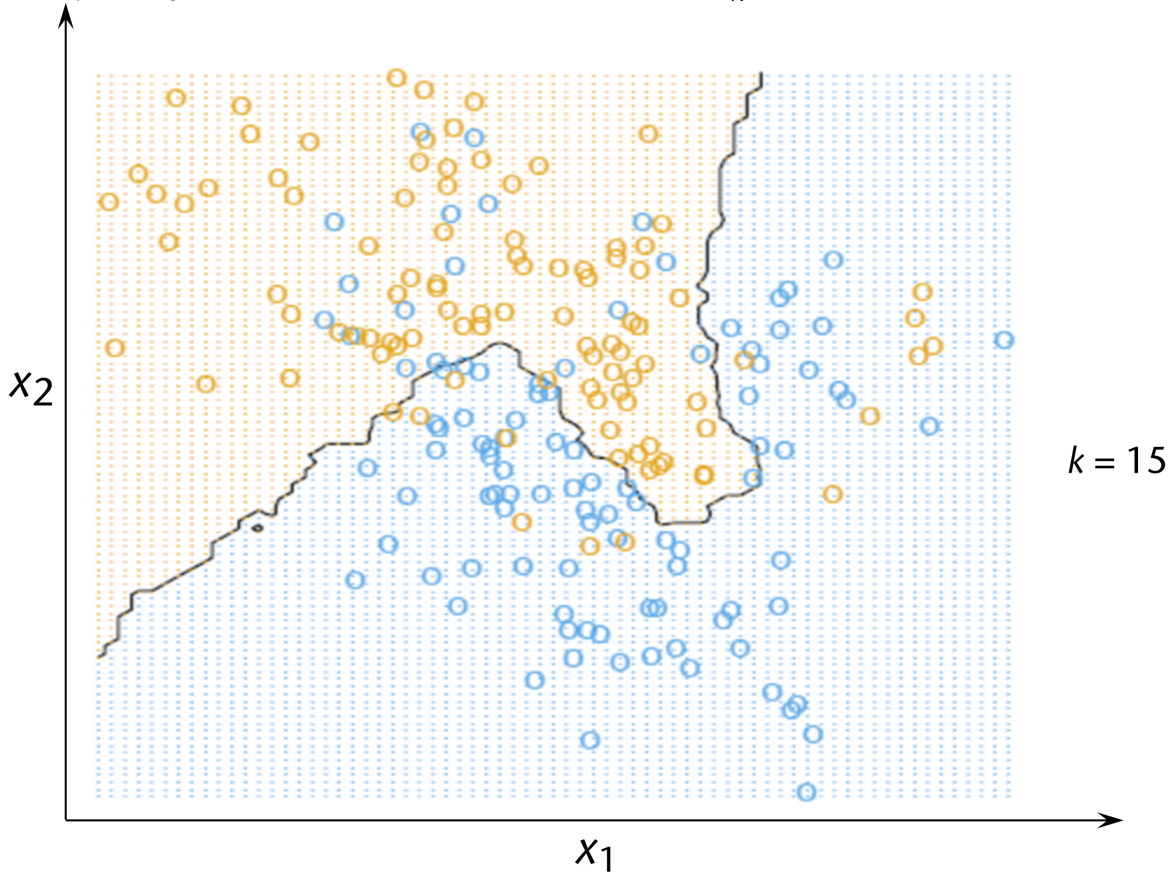
Another Solution: Nearest Neighbor (NN) Classification

- For the query point **x**, find the nearest neighbor $\mathbf{x}^* \in \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \in \mathbb{R}^d$
- Assign the class l^* to **x**



Improvement: k-NN Classification

- Instead of the 1 nearest neighbor, find the k nearest neighbors of $\mathbf{x}$, $\{\mathbf{x}_{i_1}, \ldots, \mathbf{x}_{i_k}\} \subset \mathcal{L}$
- Assign the majority of the labels $\{l_{i_1}, \ldots, l_{i_k}\}$ to $\mathbf{x}$

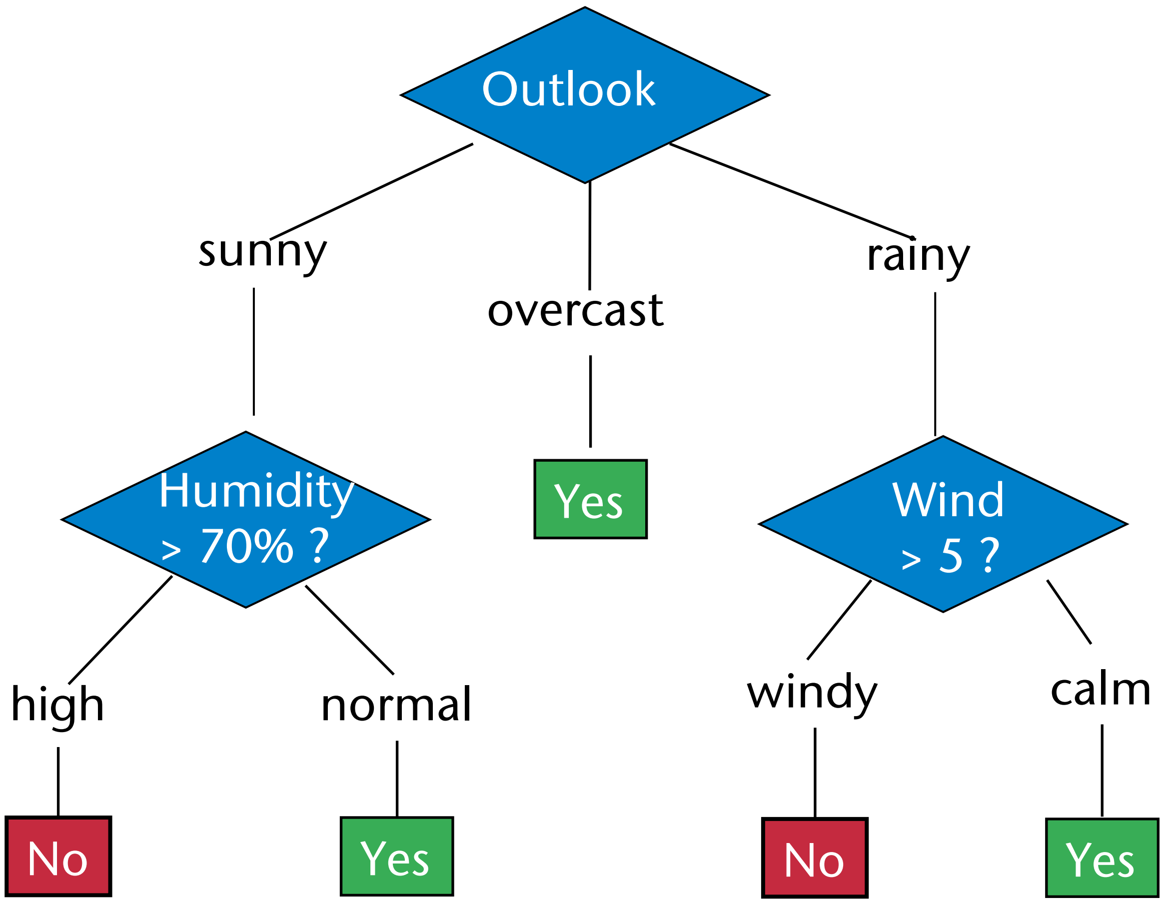


More Terminology

- The coordinates/components $x_{i,j}$ of the points $\mathbf{x}_i$ have special names: independent variables, predictor variables, features, …
  - Specific name of the $x_{i,j}$ depends on the domain / community
- The space where the $\mathbf{x}_i$ live (i.e., $\mathbb{R}^d$) is called feature space
- The labels y_i are also called target, dependent variable, response variable, …
- The set $\mathcal{L}$ is called the training set / learning set (will become clear later)

Decision Trees

- Simple example: decide whether to play tennis or not



A new sample
(= observation)
could be
(Outlook=rainy,
Wind=calm,
Humidity=high)

Pass it down the tree ⟶
decision is yes.

- The *feature space* = "all" weather conditions
  - Based on the attributes

    outlook ∈ { sunny, overcast, rainy },

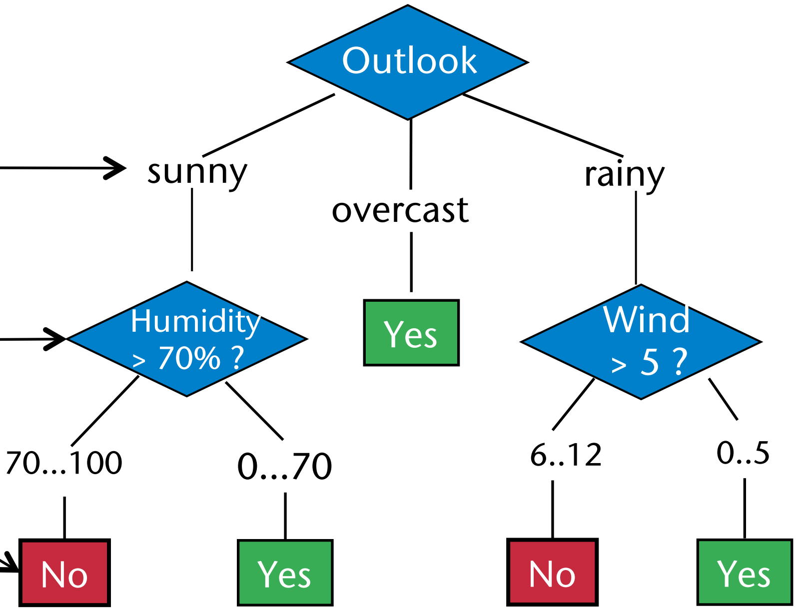
    humidity ∈ [0,100] percent ,

    wind ∈ {0, 1, ..., 12} Beaufort
  - Here, our feature space is mixed continuous/discrete
- Anatomy of a decision tree:

Branches represent different values or ranges of the attribute(s)

Each *node* tests one or more attribute(s). This is sometimes called a weak classifier

Leaves are the eventual decisions (= classes/labels)



Another Example

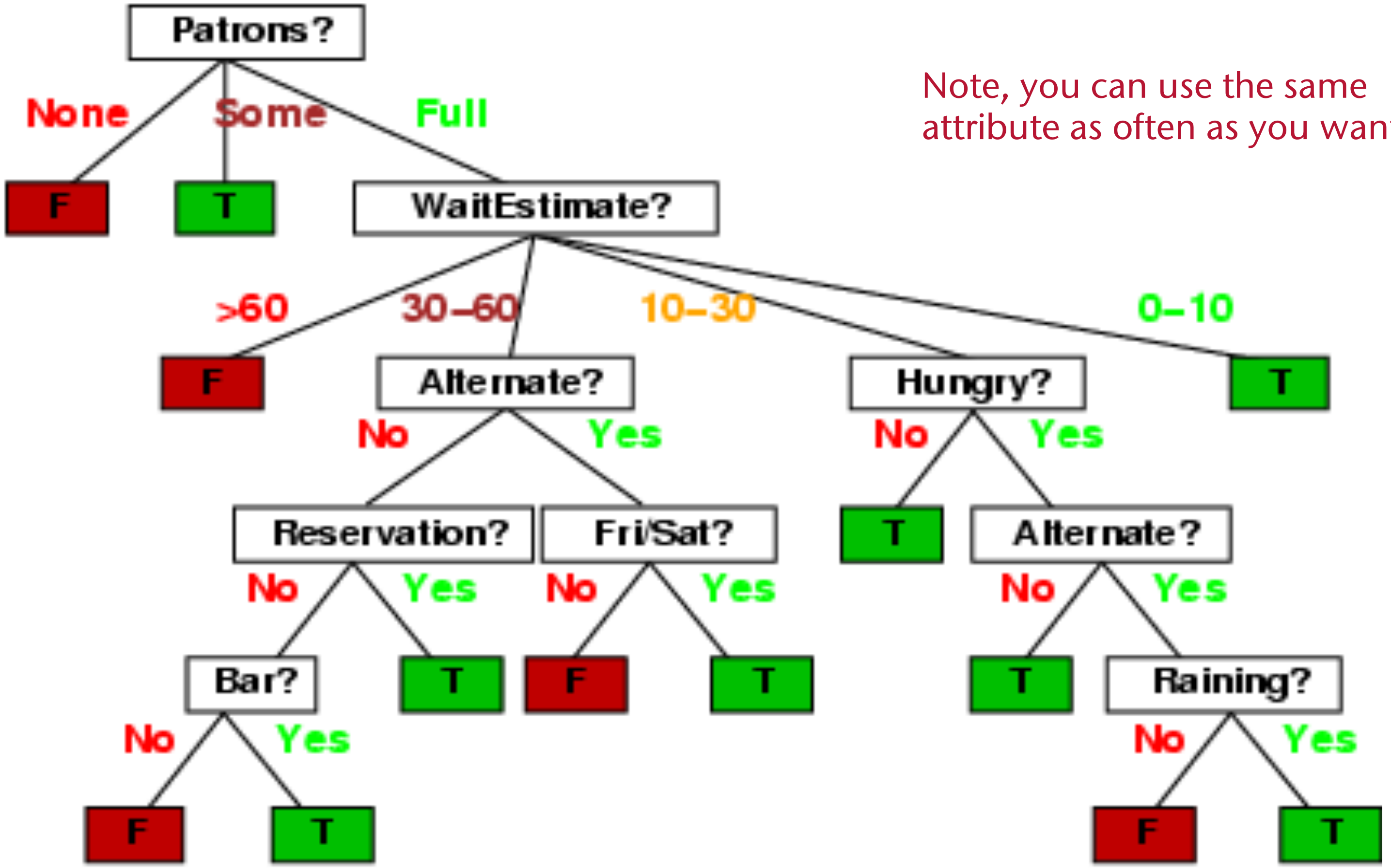
- "Please wait to be seated" ...
- Decide: *wait* or *go* some place else?
- Variables that could influence your decision:
  - Alternate: is there an alternative restaurant nearby?
  - Bar: is there a comfortable bar area to wait in?
  - Fri/Sat: is today Friday or Saturday?
  - Hungry: are we hungry?
  - Patrons: number of people in the restaurant (None, Some, Full)
  - Price: price range ($, $$, $$$)
  - Raining: is it raining outside?
  - Reservation: have we made a reservation?
  - Type: kind of restaurant (French, Italian, Thai, Burger)
  - WaitEstimate: estimated waiting time (0-10, 10-30, 30-60, >60)

- You collect data to base your decisions on:

Example	Attributes										Target
	Alt	Bar	Fri	Hun	Pat	$Price$	$Rain$	Res	$Type$	Est	$Wait$
X_1	T	F	F	T	Some	\$\$\$	F	T	French	0–10	T
X_2	T	F	F	T	Full	\$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	\$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	\$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
X_6	F	T	F	T	Some	\$\$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	\$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	\$\$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	\$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	\$\$\$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	\$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	\$	F	F	Burger	30–60	T

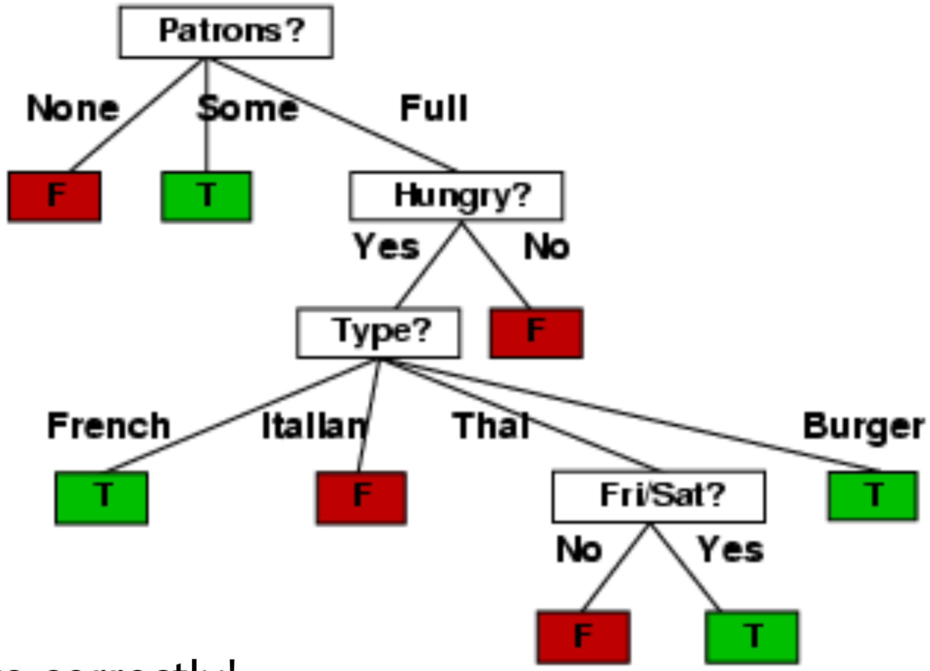
- Feature space: 10-dimensional, 6 Boolean attributes, 3 discrete attributes, one continuous attribute

- A decision tree that classifies all "training data" correctly:



Note, you can use the same attribute as often as you want

- A better decision tree:

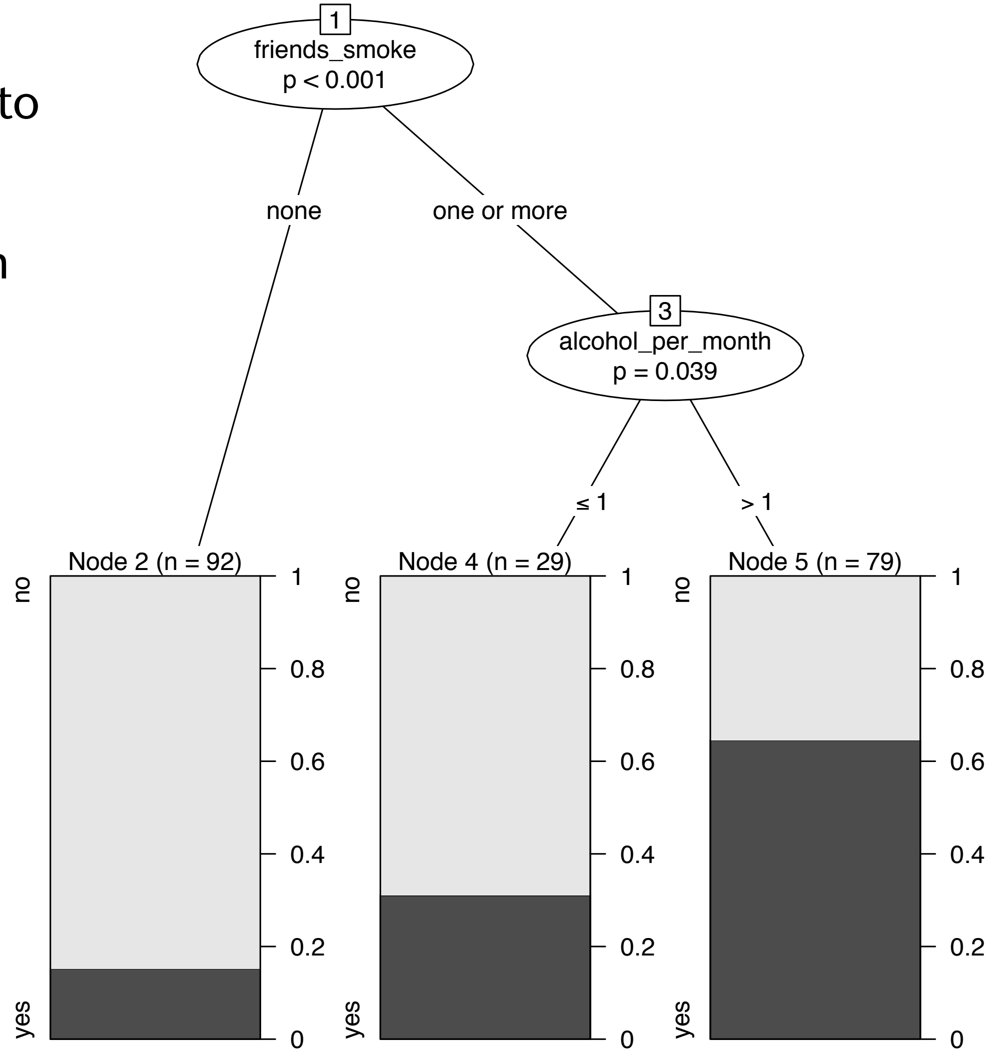


  - Also classifies all training data correctly!
  - Decisions can be made faster
- Questions:
  - How to construct (optimal) decision trees methodically?
  - How well does it generalize? (what is its generalization error?)

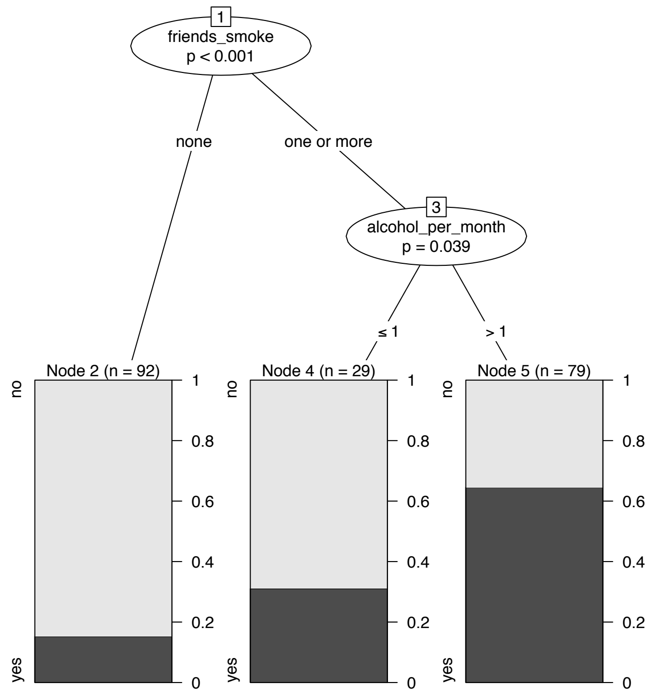
Construction (= Learning) of Decision Trees

- By way of the following example
- Goal: predict adolescents' intention to smoke within next year
  - Binary response variable *IntentionToSmoke*
- Four predictor variables (= attributes):
  - *LiedToParents* (bool) = subject has ever lied to parents about doing something they would not approve of
  - *FriendsSmoke* (bool) = one or more of the 4 best friends smoke
  - *Age* (int) = subject's current age
  - *AlcoholPerMonth* (int) = # times subject drank alcohol during past month
- Training data:
  - Kitsantas et al.: *Using classification trees to profile adolescent smoking behaviors*. 2007
  - 200 adolescents surveyed

- A decision tree:
  - Root node splits all points into *two subsets*
  - Node 2 = all data points with *FriendsSmoke* = False
  - Node 2 contains 92 points, 18% have label "yes", 82% have label "no"
  - Ditto for the other nodes

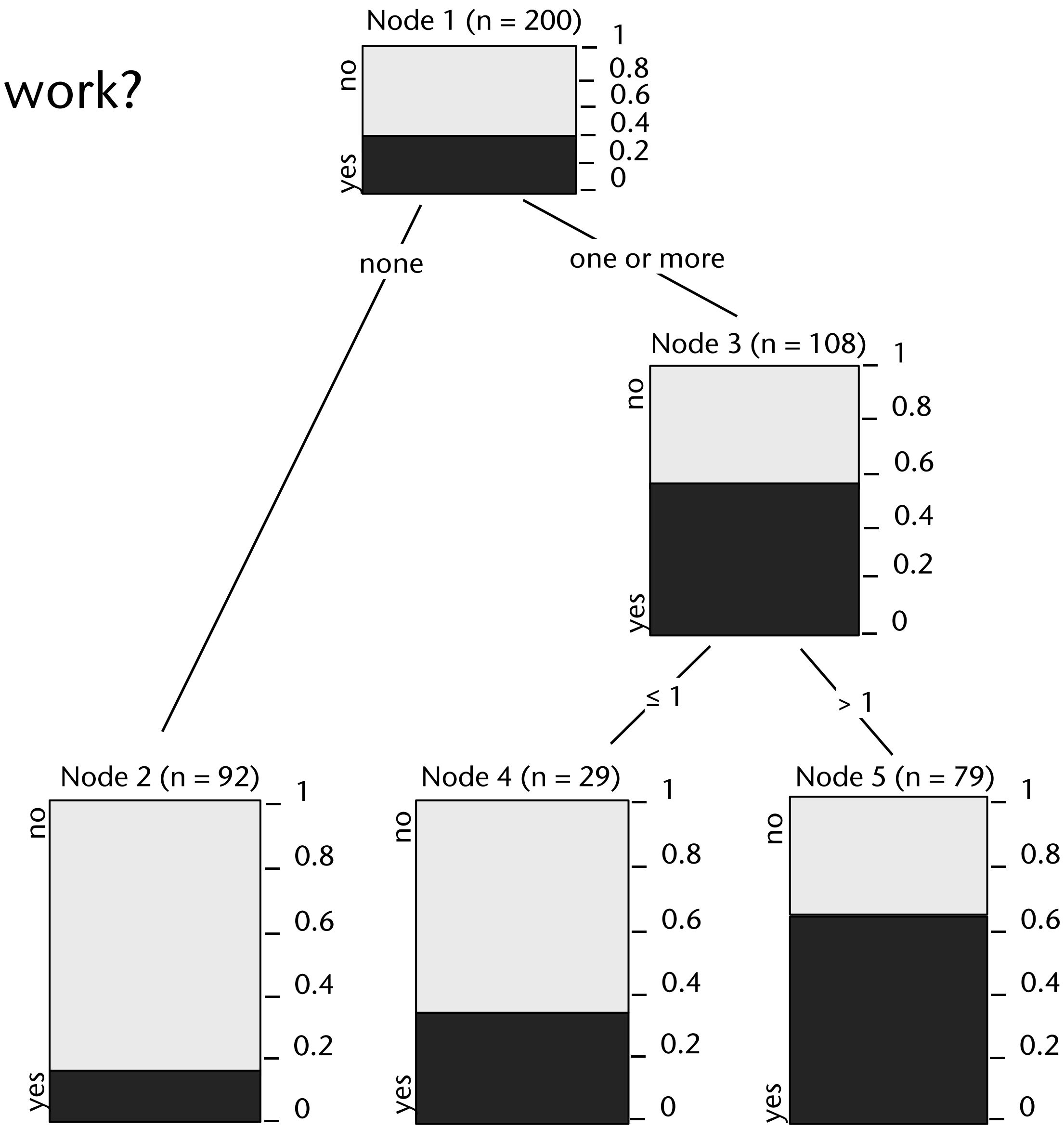


- Observation: a decision tree partitions feature space into rectangular regions:

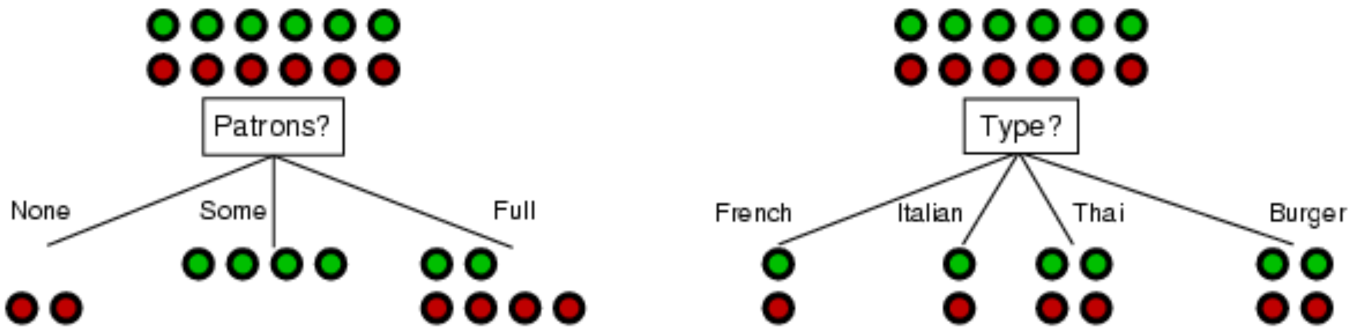


Selection of Splitting Variable and Cutpoint

- Why does our example work?
  - In the root node, *IntentionToSmoke*=yes is 40%
  - In node 2, *IntentionToSmoke*=yes is 18%, while in node 3 *IntentionToSmoke*=yes is 60%
  - So, after first split we can make better predictions



- Ideally, a good attribute (and cutpoint) splits the samples into subsets that are "all positive" or "all negative"
- Example (restaurant):



To wait or not to wait is still at 50%

- Example (abstract):