Lecture 18: Overfitting

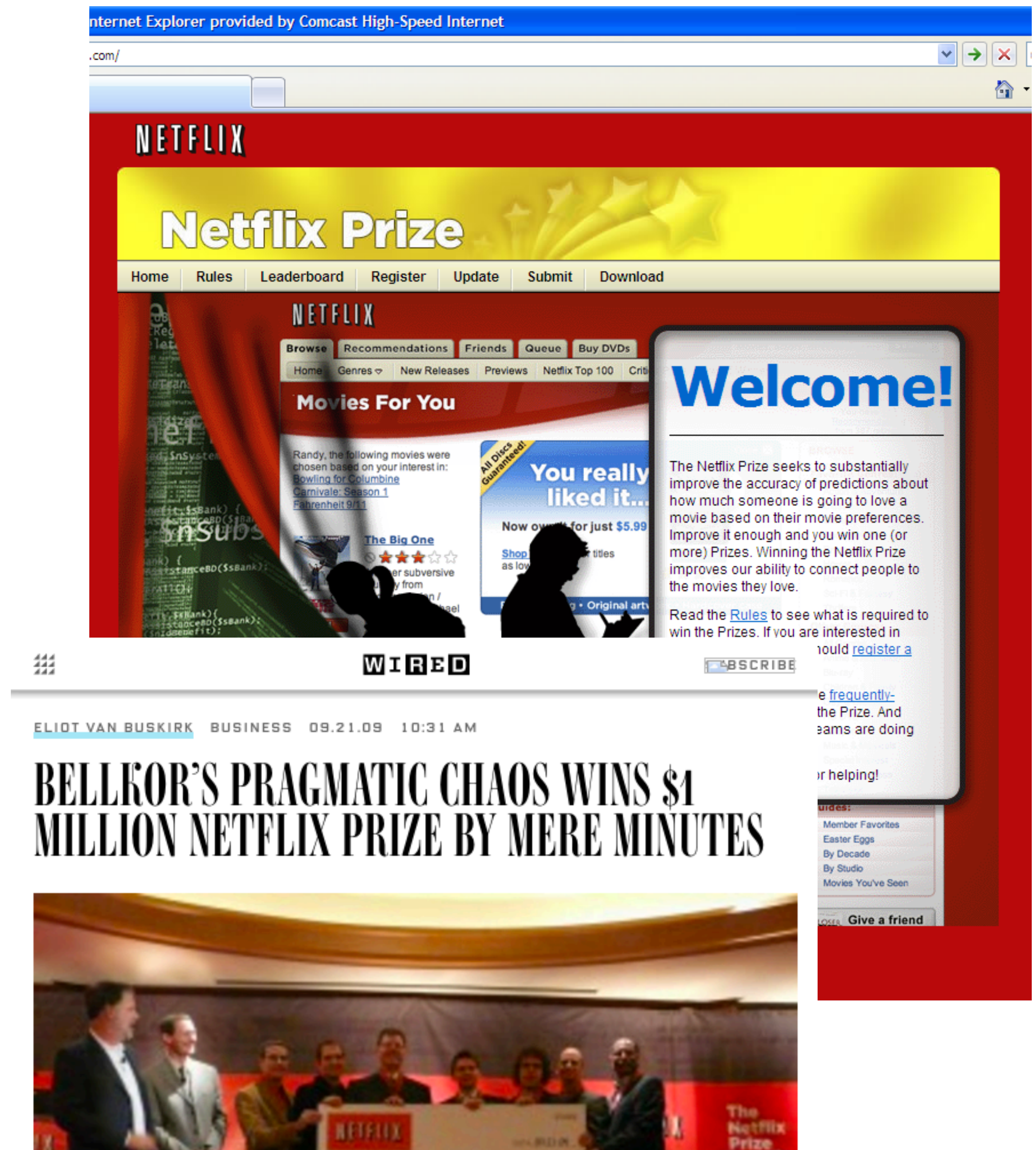
Core 109S IDWT?, Spring 2017
Michael Hay

Logistics

- Homework due tonight!
  - You must *scan* and upload
- Exam next W in class
- Let's take questions at the end of class today…

Evaluating Netflix prize contestants

- Available data: movie-user pairs with ratings
- Leaderboard:
  - Movie-user pairs (without ratings)
  - Contestant submits ratings for each pair
  - Get back total score
- Final winner: top 5 leaders were evaluated on *secret* dataset hidden from contestants. Why?



Today

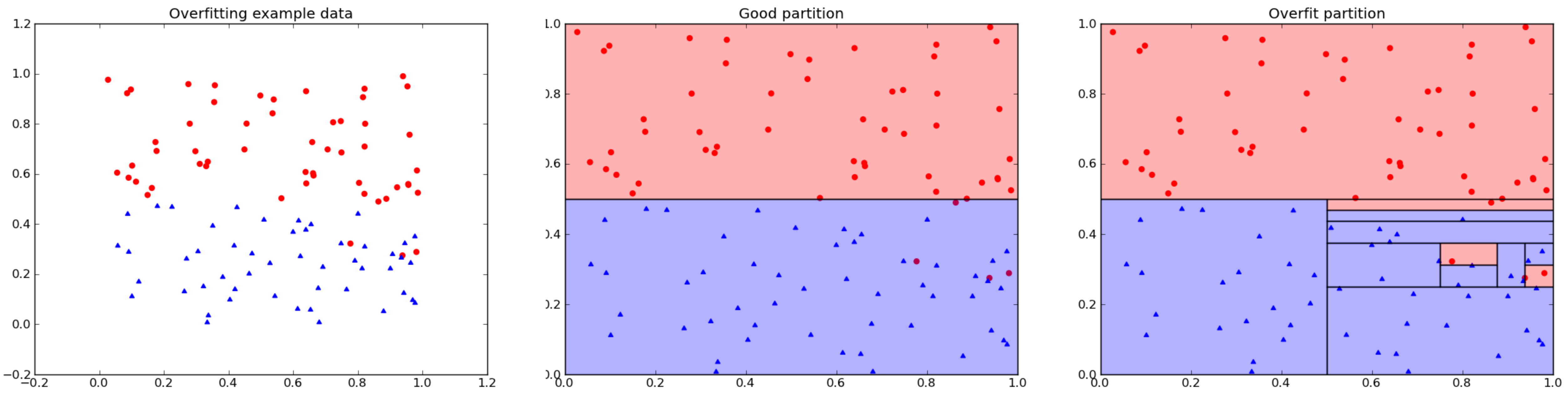
- **Overfitting**
  - What it is
  - Why it's bad
  - How to avoid it

Overfitting

- "Fitting the data *more than is warranted.*"
  — Abu-Mostafa, *Learning From Data*

- "Finding chance occurrences in data that look like interesting patterns but which do not generalize [to previously unseen data points]."
  — Provost & Fawcett, *Data Science for Business*

Overfitting illustration: decision trees



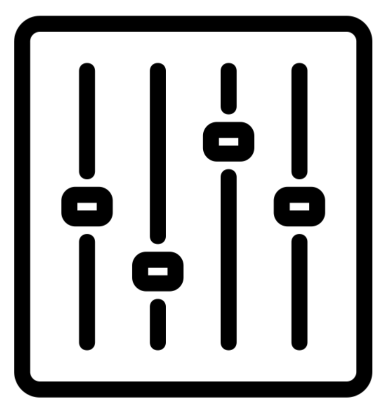
Credit: http://people.cs.uchicago.edu/~amr/122-w12/assignments/hw1a/index.html

Overfitting illustration: feature engineering

- Suppose we include features such as: "Name of sender"
- What could go wrong…
  - … with decision tree?
  - … with perceptron? (assume m binary features, for m names in training data)
- Alternative feature that capture same idea but less likely to overfit?

Hypothesis complexity

- Informally, "complexity" is how finely hypothesis can be fit to the data
- Most machine learning techniques provide one or more "knobs" or "sliders" to adjust complexity



The best algorithms have knobs that go to eleven!

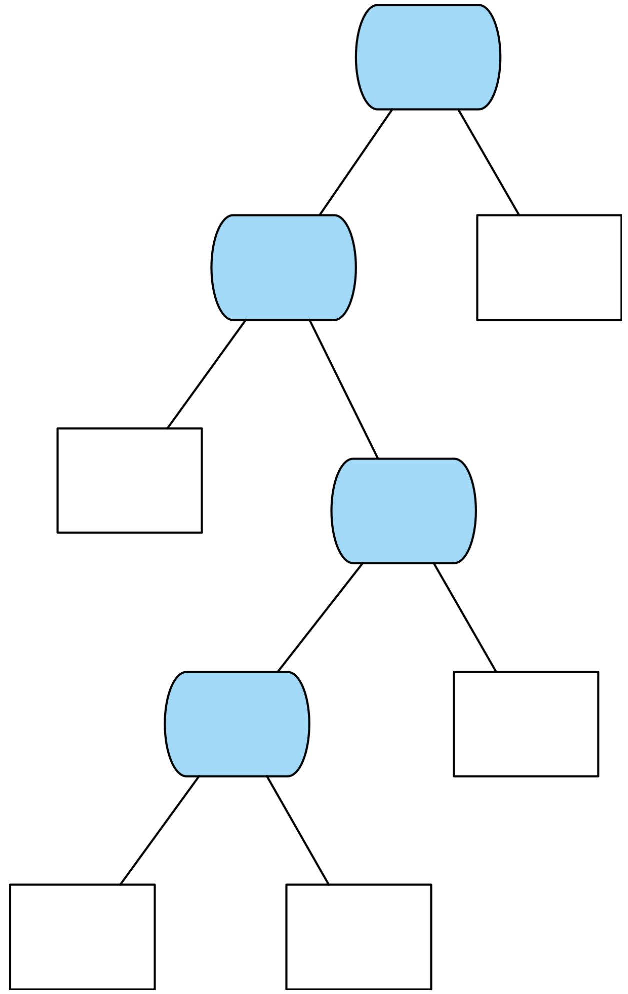
https://en.wikipedia.org/wiki/Up_to_eleven

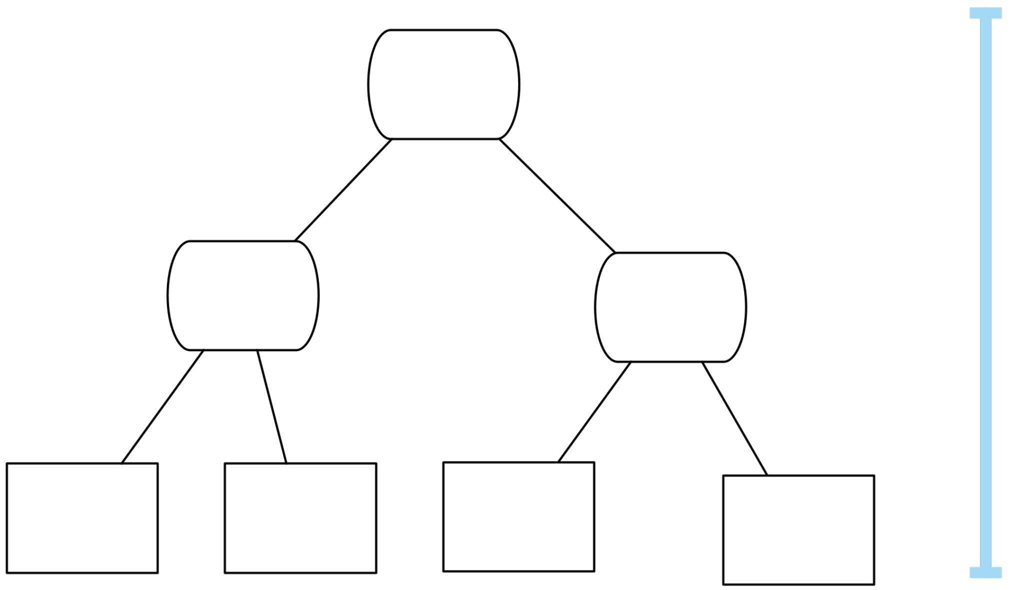
- Let's look at some examples…

Mixer by Yorlmar Campos from the Noun Project

Decision tree

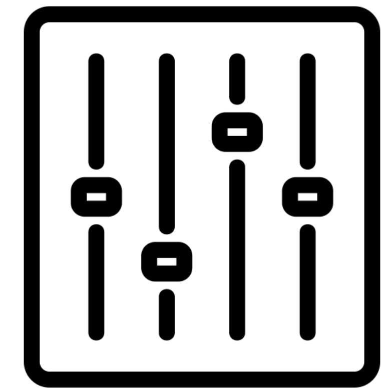
- Number of decision nodes, or maximum tree height





Mixer by Yorlmar Campos from the Noun Project

Perceptron

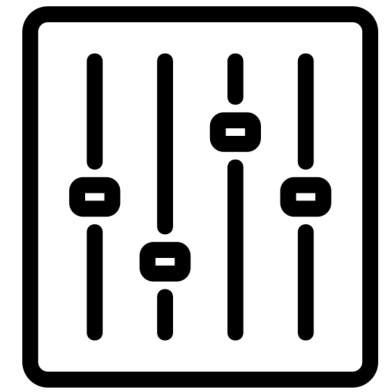


- Number of weights w_i such that $w_i \neq 0$

$$h(x) = \mathrm{sign}\left(\sum_{i=0}^{d} w_i x_i\right)$$

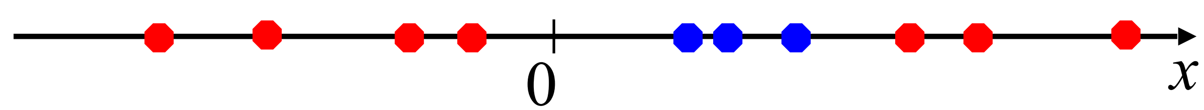
- Alternative "knob": sum of weights (after taking absolute value)
- Note: Same knob applicable to other linear models such as linear regression, logistic regression

Feature engineering

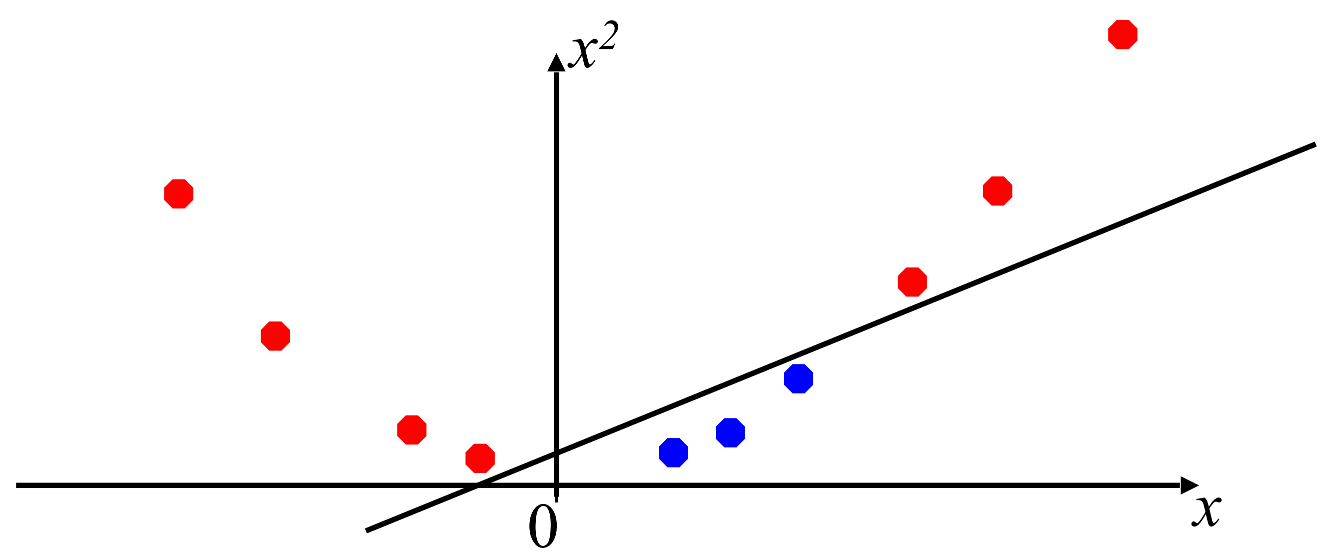


- Feature engineering is a "knob" applicable to any approach
- ***New*** features:
  - 

number of !!, ALL CAPS, SenderInContacts, # misspellings
- ***Transformations*** of existing features:
  - Dataset: inseparable by perceptron in original feature space

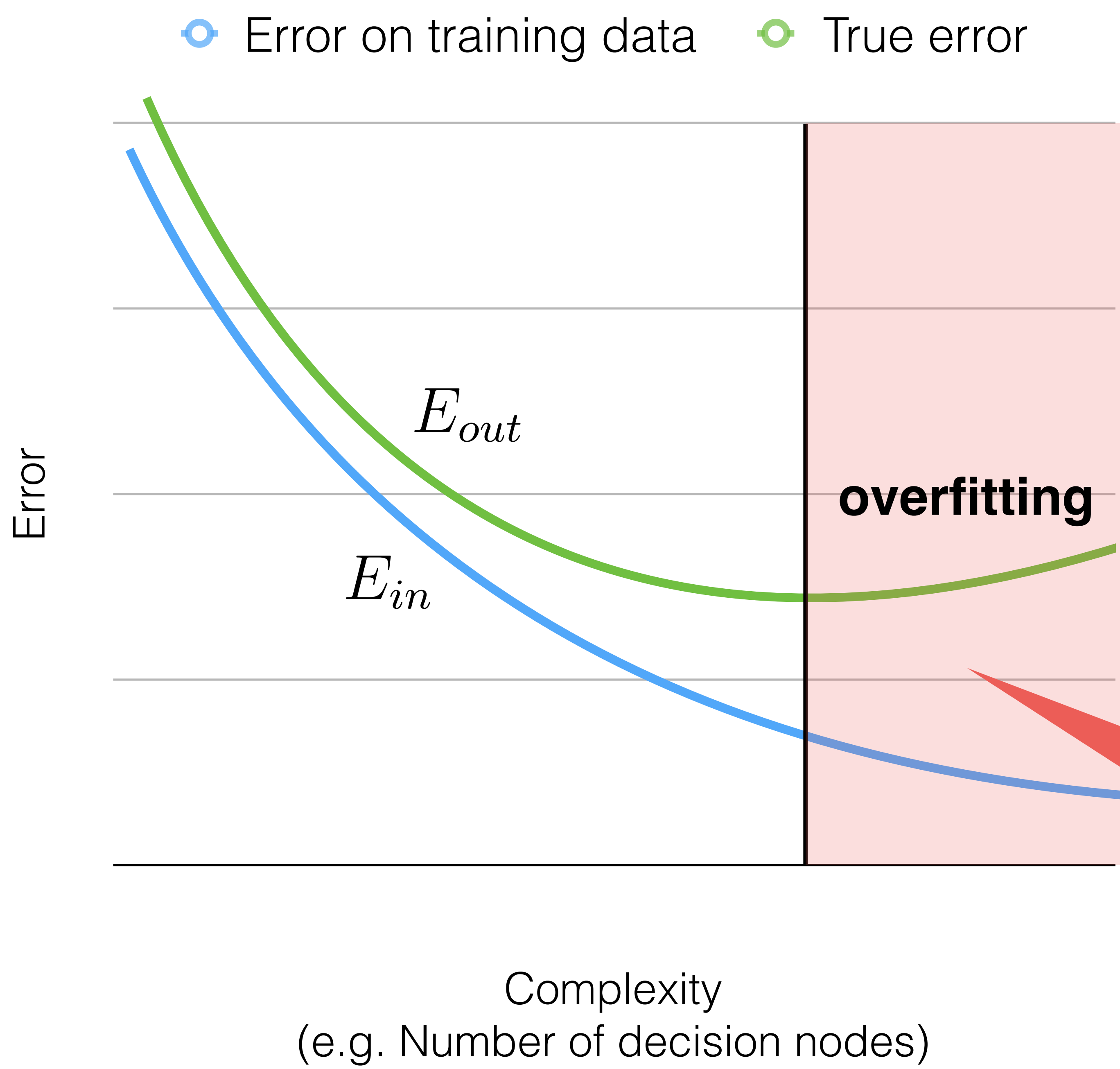


  - Becomes separable when we add second feature $x_{new} = x^2$



Mixer by Yorlmar Campos from the Noun Project

Overfitting revisited



True error starts to *increase*. Overfitting may cause harm! Why does this happen?

Overfitting starts when additional complexity decreases E_{in} but *not* E_{out}

Today

- **Overfitting**
    - What it is
    - **Why it's bad**
    - How to detect it
    - How to avoid it

Example

3/16 ≈ 19% of population has $A=a_1$, $B=b_1$, and T=yes

Sample

A	B	T
a_1	b_1	y
a_1	b_1	y
a_1	b_1	y
a_2	b_2	n
a_1	b_2	n
a_2	b_1	n
a_2	b_2	n
a_2	b_1	n

- Suppose we want to predict T based on two binary attributes A and B
- Suppose we will train a decision tree on a *sample* from the *population*

Population

A	B	T=yes	T=no
a_1	b_1	3/16	1/16
	b_2	3/16	1/16
a_2	b_1	1/16	3/16
	b_2	1/16	3/16

A vs. T

A	T=yes	T=no
a_1	6/16	2/16
a_2	2/16	6/16

When $A=a_1$, T will be yes 6/(6+2)= 75% of the time

When $A=a_2$, T will be yes 2/(6+2)= 25% of the time

B vs. T

B	T=yes	T=no
b_1	4/16	4/16
b_2	4/16	4/16

Knowing B is not helpful for predicting T

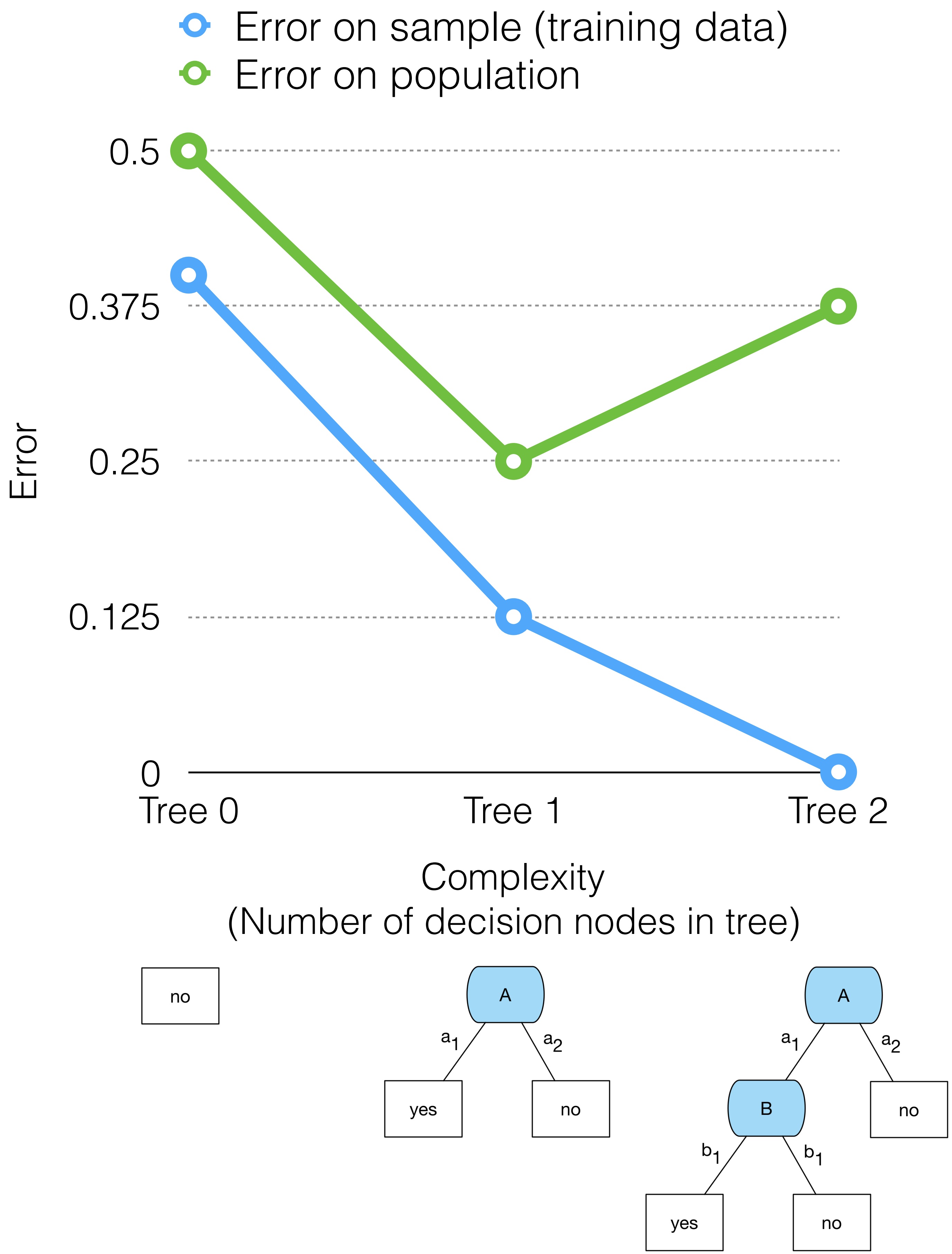
Exercise: Fit Trees 1 and 2 to this data. You should be able to "eyeball" the solution without computing info gain.

Example, continued…

- Given sample of records from population
- Train three decision trees of varying complexity
  - Tree 0: at most *zero* decision nodes
  - Tree 1: at most *one* decision node
  - Tree 2: no constraints
- (Shown on board)

Sample
(Training data)

A	B	T
a_1	b_1	y
a_1	b_1	y
a_1	b_1	y
a_2	b_2	n
a_1	b_2	n
a_2	b_1	n
a_2	b_2	n
a_2	b_1	n



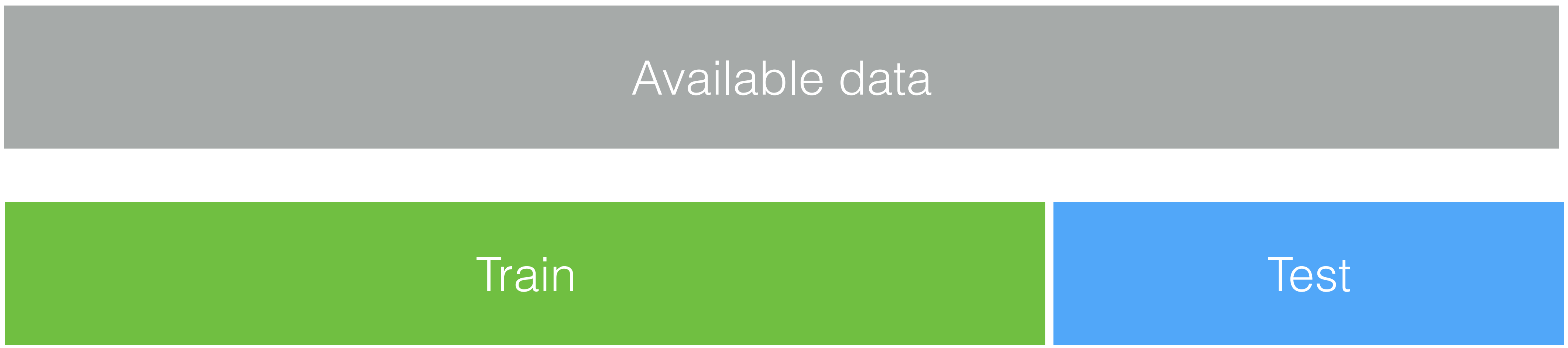
Overfitting…

- …is not due to "pathological" sample: spurious "patterns" are likely to occur by chance
- … isn't unique to trees: all algorithms fit patterns in the data and can be fooled
- … can hurt: spurious patterns can cause hypothesis to make incorrect decisions

Today

- **Overfitting**
  - What it is
  - Why it's bad
  - How to **detect it**
  - How to avoid it

Train/test split



- Lesson learned: measuring error on the data that was used to fit hypothesis may not be indicative of "true" error
- Solution: split data into two parts
  - Train: use this for learning algorithm
  - Test: use this to judge accuracy of *final* learned hypothesis
- If training error is (a lot) lower than test error, overfitting *has likely* occurred.

Today

- **Overfitting**
  - What it is
  - Why it's bad
  - How to detect it
  - How to **avoid it**

How to avoid overfitting

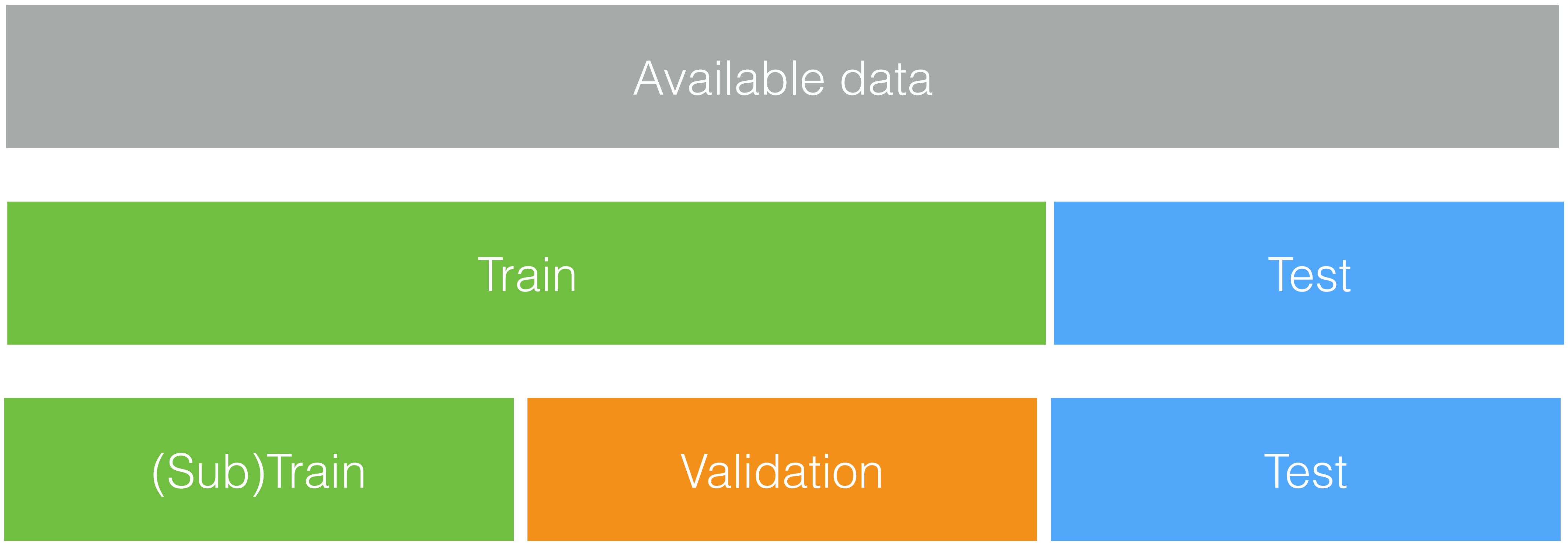
- Two high level approaches

  - **Regularization**: add a "penalty" for hypothesis complexity, adjust learning algorithm to find hypothesis that minimizes:

    training error + penalty

  - **Validation set**: fit hypotheses of varying complexity, set aside a portion of training data to judge the best overall

Validation set



- Break training data into two parts: (Sub)Train and Validation
- Break hypothesis set into subsets of increasing complexity $H_1, H_2, H_3, \ldots, H_C$
- Use (Sub)Train to find best hypothesis in each hypothesis group: $h_{H1}, h_{H2}, \ldots, h_{HC}$
- Use validation to compare these hypotheses and pick overall best

Drawbacks of splitting into train/validation/test

- A fixed amount of labeled examples are now divided into three subsets

- We want each subset to be *as big as possible*

    - Bigger (sub)train: more *information* for training

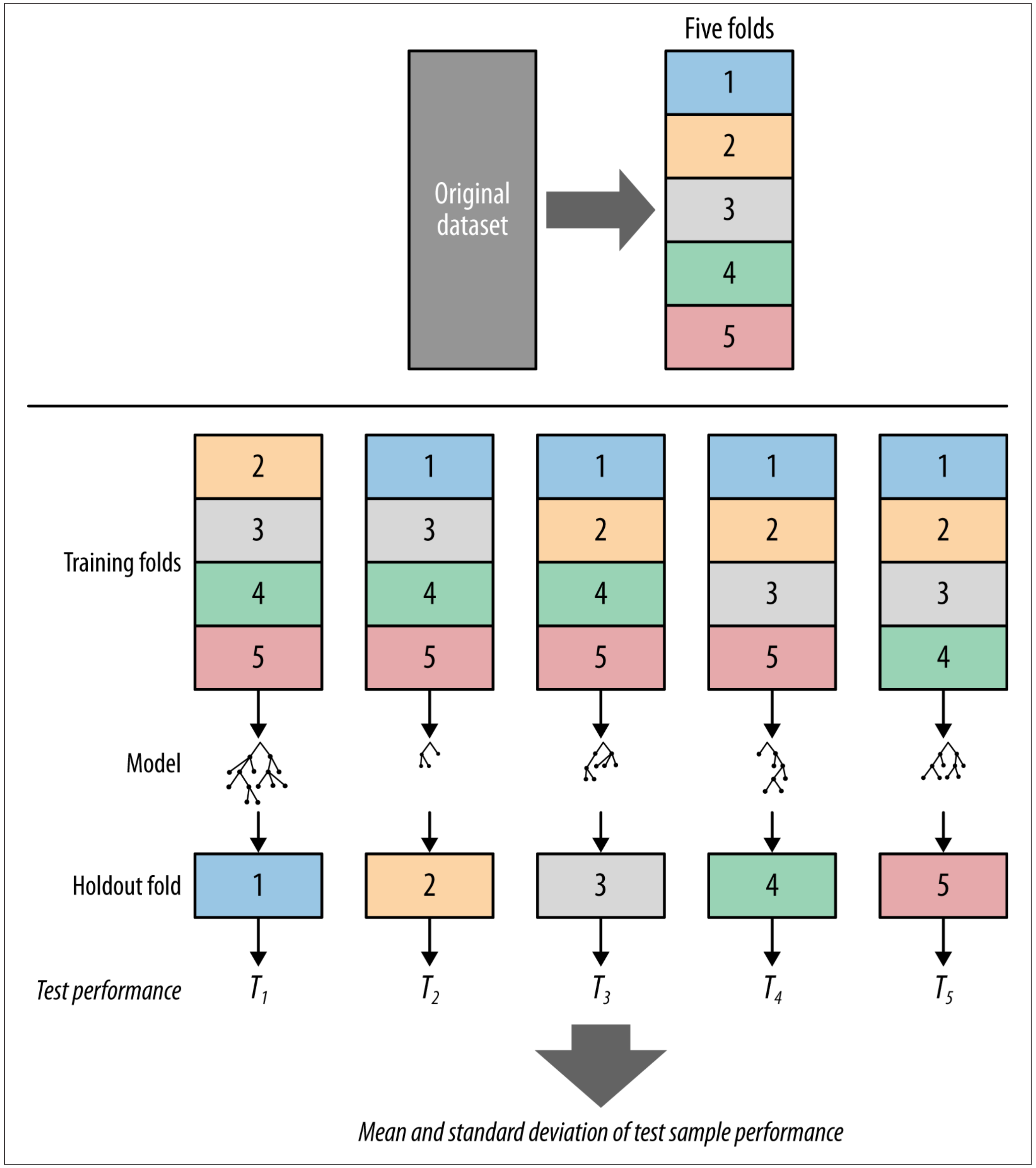
    - Bigger validation and test: more reliable estimates of "true" error

Cross validation

- Applicable for splitting a dataset into two parts:
  - Example: **train/test**
  - Example: (sub)train/validate
- Idea: use every example as a test example
- Extreme version: ***Leave one out cross validation***
  - Learn on n-1 examples, test on nth example, record result
  - Repeat this n times! Average the results.

Credit: Data Science for Business, Provost and Fawcett

Cross validation



Credit: Data Science for Business, Provost and Fawcett