Decision Tree Learning (Part 2)

Mark Craven and David Page
Computer Sciences 760
Spring 2019

Stopping criteria

We should form a leaf when
- all of the given subset of instances are of the same class
- we've exhausted all of the candidate splits

Is there a reason to stop earlier, or to prune back the tree?

How can we assess the accuracy of a tree?

- Can we just calculate the fraction of training examples that are correctly classified?

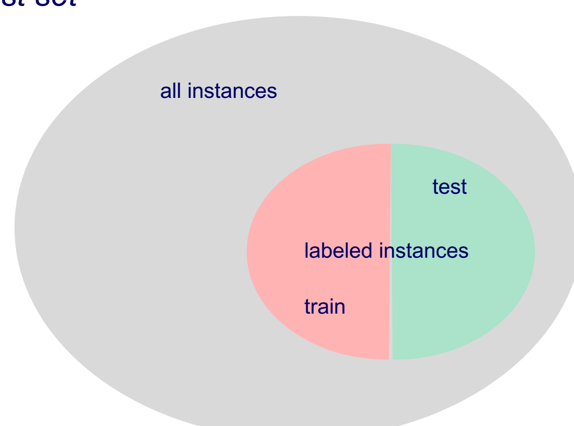
- Consider a problem domain in which instances are assigned labels at random with $P(Y = \mathrm{T}) = 0.5$

    - How accurate would a learned decision tree be on previously unseen instances?

    - How accurate would it be on its training set?

How can we assess the accuracy of a tree?

- to get an unbiased estimate of a learned model's accuracy, we must use a set of instances that are held-aside during learning
- this is called a *test set*



Overfitting

- consider error of model h over
  - training data: $error_D(h)$
  - entire distribution of data: $error(h)$

- model $h \in H$ *overfits* the training data if there is an alternative model $h' \in H$ such that

$$error(h) > error(h')$$

$$error_D(h) < error_D(h')$$

Overfitting

- the concept of overfitting is not specific to decision trees

- overfitting avoidance is one of the principal challenges in machine learning!

Overfitting with noisy data

suppose

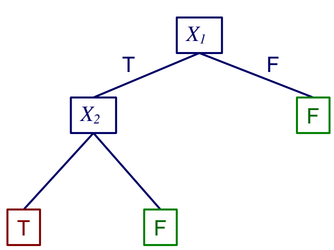
- the target concept is $Y = X_1 \wedge X_2$
- there is noise in some feature values
- we're given the following training set

X_1	X_2	X_3	X_4	X_5	…	Y
T	T	T	T	T	…	T
T	T	F	F	T	…	T
T	F	T	T	F	…	T
T	F	F	T	F	…	F
T	F	T	F	F	…	F
F	T	T	F	T	…	F

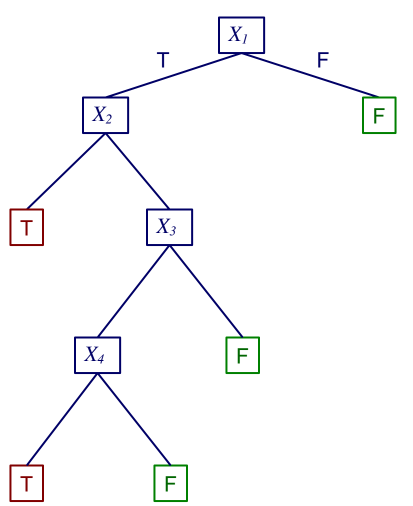
noisy value

Overfitting with noisy data

correct tree

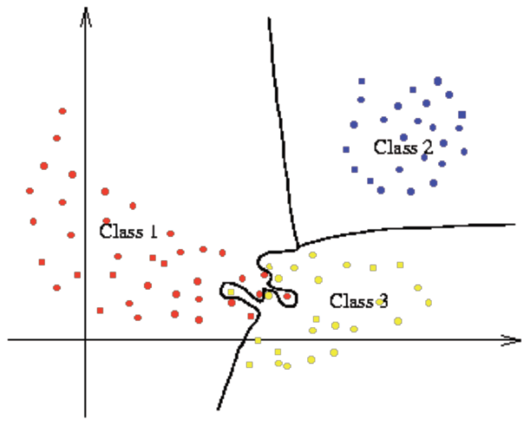


tree that fits noisy training data



Overfitting visualized

consider a problem with
- 2 continuous features
- 3 classes
- some noisy training instances

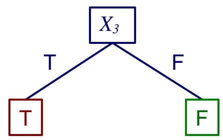


Overfitting with noise-free data

suppose
- the target concept is $Y = X_1 \wedge X_2$
- P(X_3 = T) = 0.5 for both classes
- P(Y = T) = 0.67
- we're given the following training set

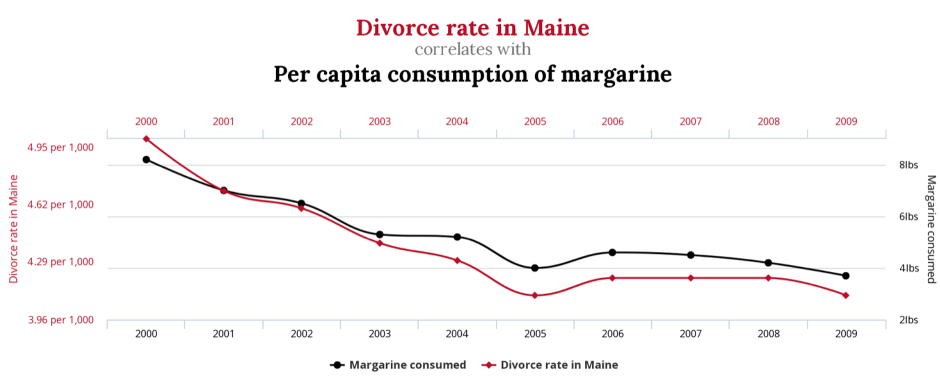
X_1	X_2	X_3	X_4	X_5	…	Y
T	T	T	T	T	…	T
T	T	T	F	T	…	T
T	T	T	T	F	…	T
T	F	F	T	F	…	F
F	T	F	F	T	…	F

Overfitting with noise-free data

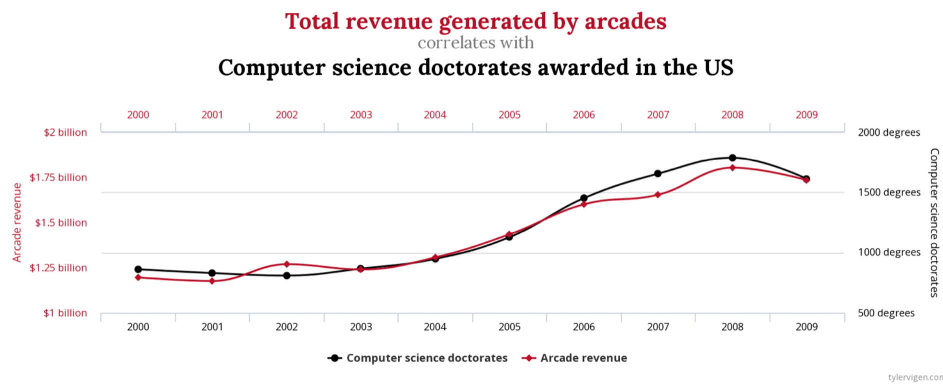
	training set accuracy	test set accuracy
	100%	50%
	60%	67%

- because the training set is a limited sample, there might be (combinations of) features that are correlated with the target concept by chance

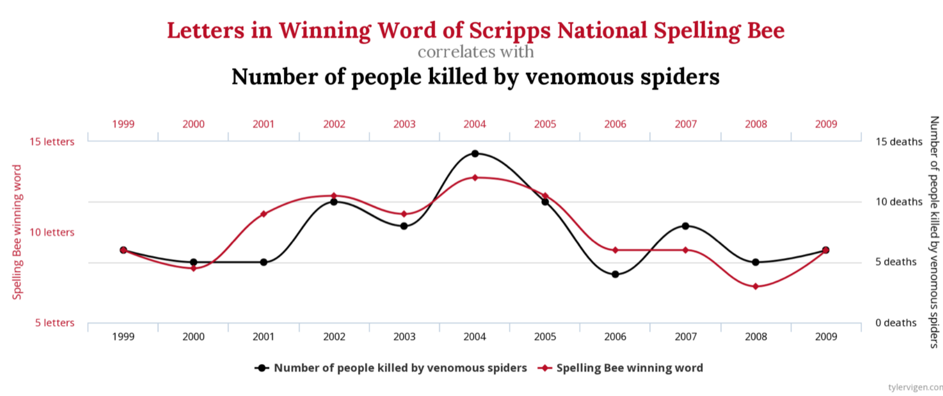
tylervigen.com



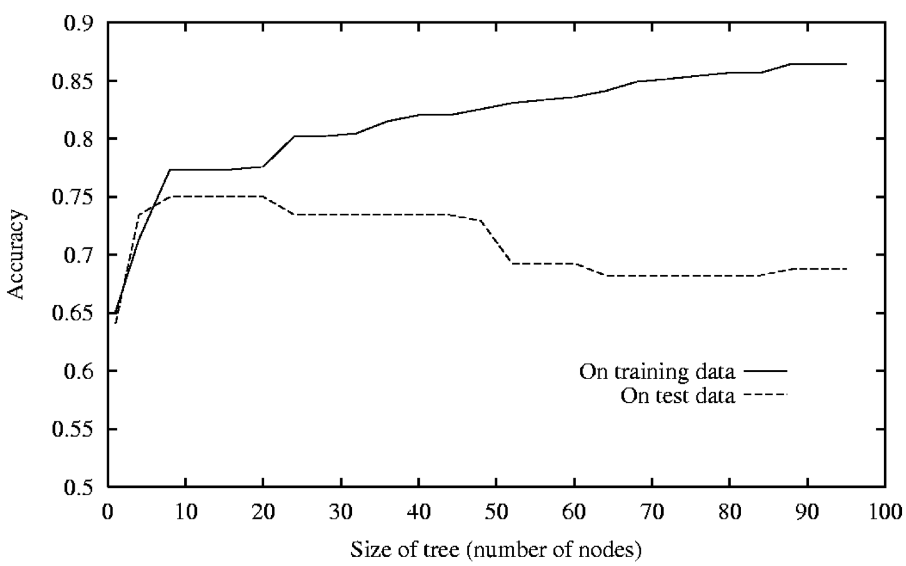
tylervigen.com



tylervigen.com



Overfitting in decision trees



Avoiding overfitting in DT learning

two general strategies to avoid overfitting

1. *early stopping*: stop if further splitting not justified by a statistical test
   - Quinlan's original approach in ID3

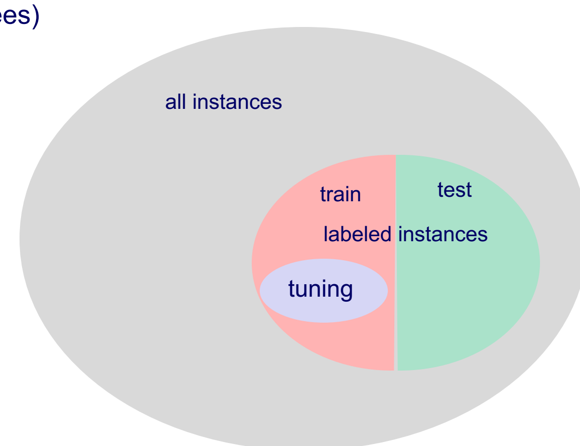
2. *post-pruning*: grow a large tree, then prune back some nodes
   - more robust to myopia of greedy tree learning

Pruning in C4.5

1. split given data into training and *tuning* (*validation*) sets
2. grow a complete tree
3. do until further pruning is harmful
   - evaluate impact on tuning-set accuracy of pruning each node
   - greedily remove the one that most improves tuning-set accuracy

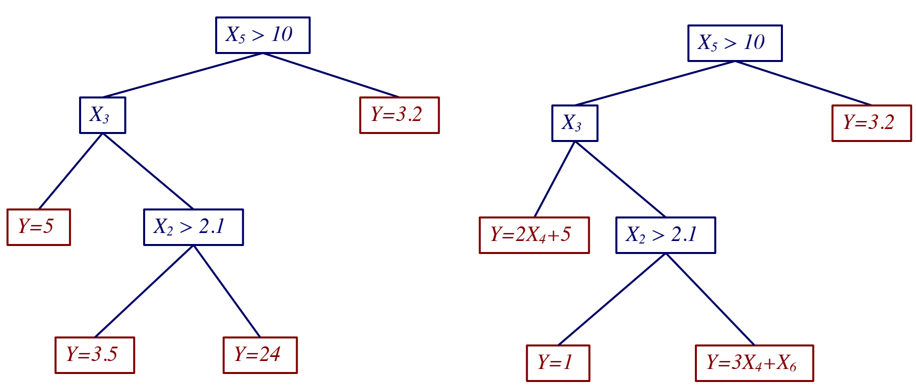
Tuning sets

- a *tuning set* (a.k.a. *validation set*) is a subset of the training set that is held aside
  - not used for primary training process (e.g. tree growing)
  - but used to select among models (e.g. trees pruned to varying degrees)



Regression trees

- in a regression tree, leaves have functions that predict numeric values instead of class labels
- the form of these functions depends on the method
  - CART uses constants
  - some methods use linear functions (model trees)



Regression trees in CART

- CART does *least squares regression* which tries to minimize

$$\sum_{i=1}^{|D|}\left(y^{(i)} - \hat{y}^{(i)}\right)^2 = \sum_{L \in \text{leaves}} \sum_{i \in L}\left(y^{(i)} - \hat{y}^{(i)}\right)^2$$

target value for i^{th} training instance

value predicted by tree for i^{th} training instance (average value of y for training instances reaching the leaf)

- at each internal node, CART chooses the split that most reduces this quantity

Comments on decision tree learning

- widely used approach
- many variations
- fast in practice
- provides humanly comprehensible models when trees are not too big
- insensitive to monotone transformations of numeric features
- standard methods learn axis-parallel hypotheses*
- standard methods not suited to on-line setting*
- usually not among most accurate learning methods

* although variants exist that are exceptions to this