CS 760: Machine Learning
Decision Trees & Evaluation

Fred Sala

University of Wisconsin-Madison

Sept. 23, 2021

Announcements

- **Announcements**:
  - HW 2 released today
  - Project info to be released Tuesday
- **Class roadmap:**

Thursday Sept. 23	Evaluation
Tuesday Sept. 28	Regression I
Thursday Sept. 30	Regression II
Tuesday, Oct. 5	Naive Bayes
Thursday, Oct. 7	Neural Networks I

Supervised Learning

Outline

- **Continuing from last time: Decision trees**
  - Information gain, stopping criteria, overfitting, pruning, variations
- **Evaluation: Generalization**
  - Train/test split, random sampling, cross validation
- **Evaluation: Metrics**
  - Confusion matrices, ROC curves, precision/recall

Outline

- **Continuing from last time: Decision trees**
  - Information gain, stopping criteria, overfitting, pruning, variations
- **Evaluation: Generalization**
  - Train/test split, random sampling, cross validation
- **Evaluation: Metrics**
  - Confusion matrices, ROC curves, precision/recall

DT Learning: InfoGain Limitations

- InfoGain is biased towards tests with many outcomes
  - A feature that uniquely identifies each instance
  - Splitting on it results in many branches, each of which is “pure” (has instances of only one class)
  - **Maximal** information gain!

- Use **GainRatio**: normalize information gain by entropy

$$\text{GainRatio}(D, S) = \frac{\text{InfoGain}(D,S)}{H_D(S)} = \frac{H_D(Y) - H_D(Y|S)}{H_D(S)}$$

DT Learning: GainRatio

- Why?
  - Suppose S is a *binary split*. InfoGain limited to 1 bit, no matter what.

$$\text{InfoGain}(D, S) = H_D(Y) - H_D(Y|S)$$

Intuition: at most, S tells us Y is in one half of its classes or the other

  - Now suppose S is different for each instance (i.e., student number).
    - Uniquely determines Y for each point, but useless for generalization.
    - But, then $H_D(Y|S) = 0$, so maximal information gain!
  - Control this by normalizing by $H_D(S)$.
    - Above: for *n* instances, $H_D(S) = \log_2(n)$

$$\text{GainRatio}(D, S) = \frac{\text{InfoGain}(D,S)}{H_D(S)} = \frac{H_D(Y) - H_D(Y|S)}{H_D(S)}$$

Inductive Bias

- Recall: ***Inductive bias***: assumptions a learner uses to predict y_i for a previously unseen instance $\boldsymbol{x}_i$
- Two components
  - *hypothesis space bias*: determines the models that can be represented
  - *preference bias*: specifies a preference ordering within the space of models

learner	hypothesis space bias	preference bias
ID3 decision tree	trees with single-feature, axis-parallel splits	small trees identified by greedy search
k-NN	Voronoi decomposition determined by nearest neighbors	instances in neighborhood belong to same class

DT Learning: Stopping Criteria

Form a leaf when

- All of the given subset of instances are same class
- We've exhausted all of the candidate splits

- Stop earlier?

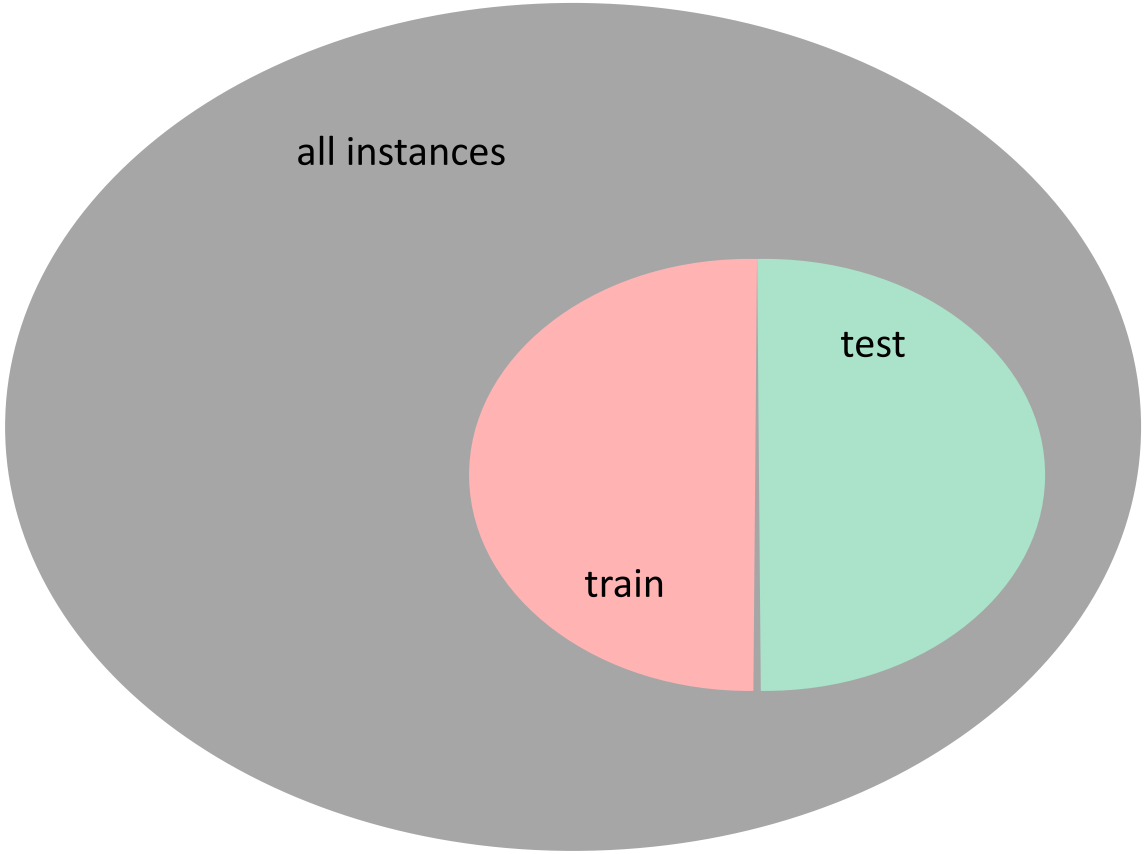
Evaluation: Accuracy

- Can we just calculate the fraction of training instances that are correctly classified?
- Consider a problem domain in which instances are assigned labels at random with $P(Y = 1) = 0.5$
  - How accurate would a learned decision tree be on previously unseen instances?
  - How accurate would it be on its training set?

Evaluation: Accuracy

To get unbiased estimate of model accuracy, we must use a set of instances that are **held-aside** during learning

- This is called a **test set**



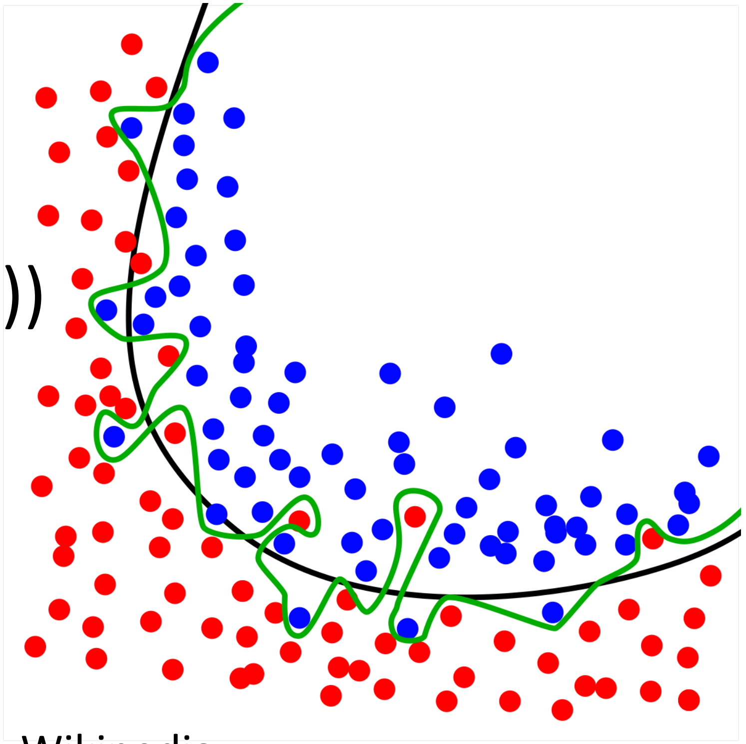
Overfitting

Notation: error of model *h* over

- training data: $error_D(h)$
- entire distribution of data: $error_{\mathcal{D}}(h)$

Model *h* **overfits** training data if it has

- a low error on the training data (low $error_D(h)$)
- high error on the entire distribution (high $error_{\mathcal{D}}(h)$)



Wikipedia

Overfitting Example: Noisy Data

Target function is $Y = X_1 \wedge X_2$

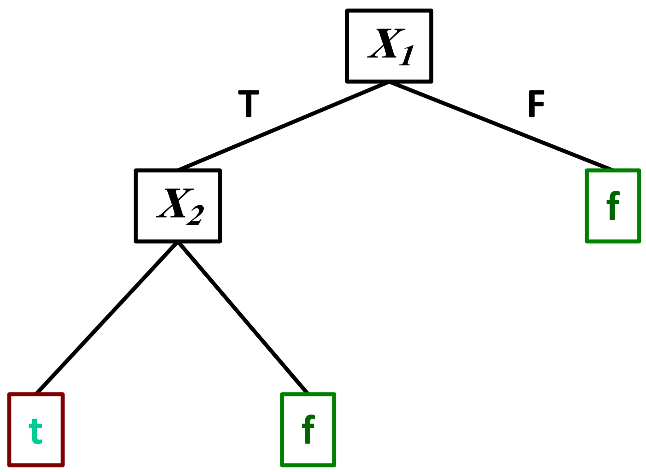
- There is **noise** in some feature values
- Training set:

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

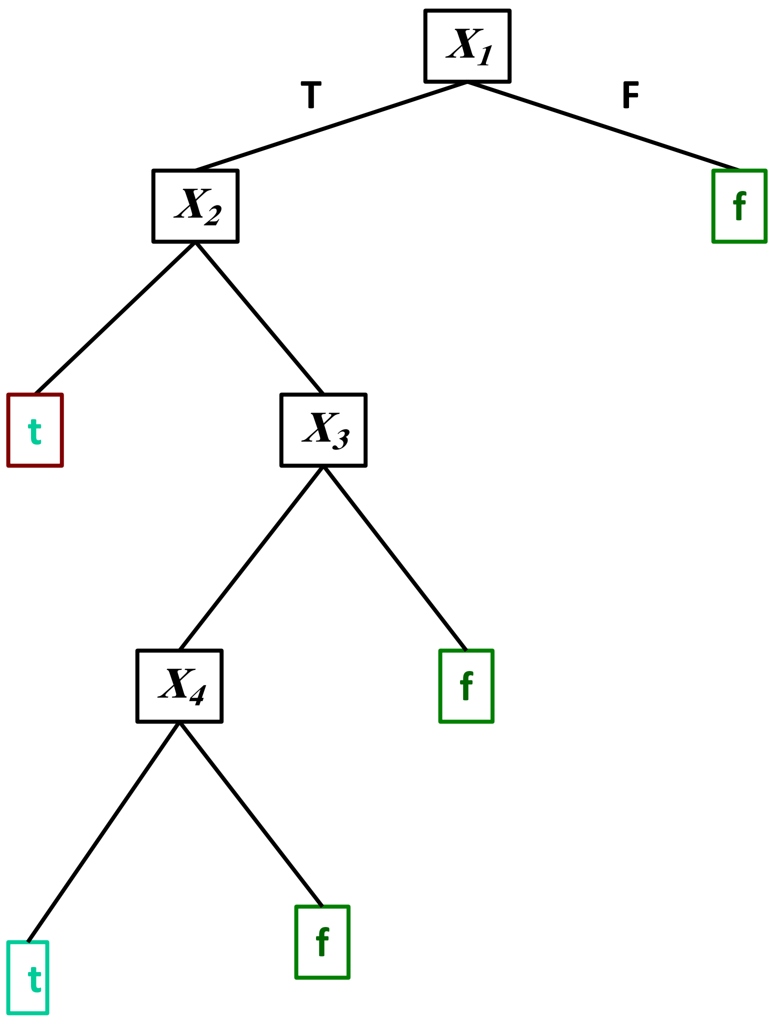
noisy value

Overfitting Example: Noisy Data

Correct tree



Tree that fits noisy training data



Overfitting Example: Noise-Free Data

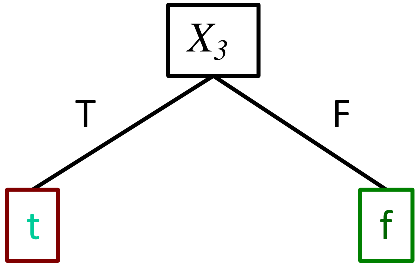
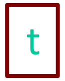
Target function is $Y = X_1 \wedge X_2$

- P(X_3 = t) = 0.5 for both classes
- P(Y = t) = 0.67
- 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

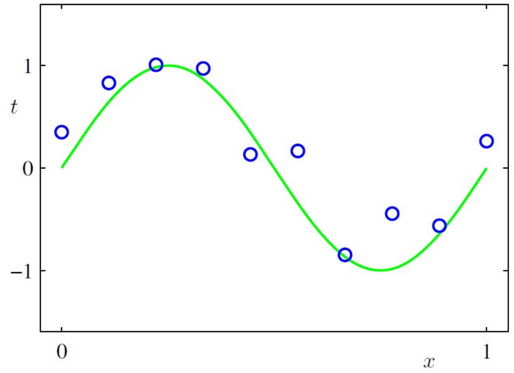
Overfitting Example: Noise-Free Data

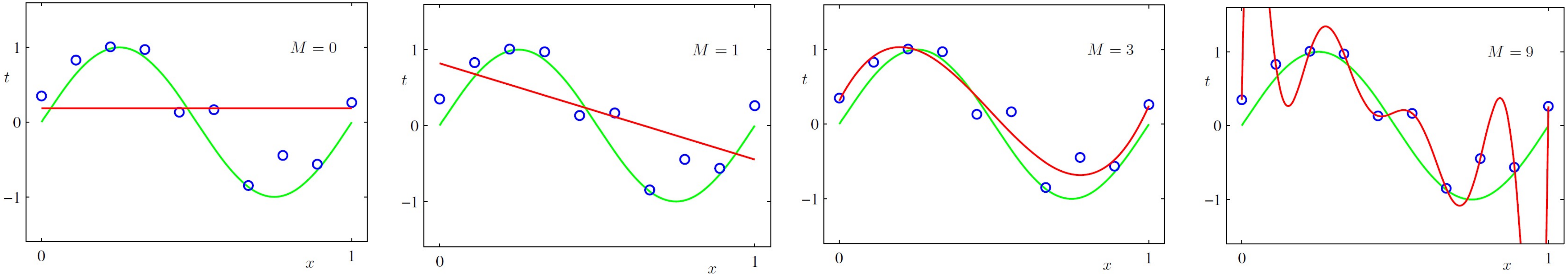
- Training set is a **limited sample.** There might be (combinations of) features that are correlated with the target concept by chance

	Training set accuracy	**Test set accuracy**
	100%	50%
	66%	66%

Overfitting Example: Polynomial Regression

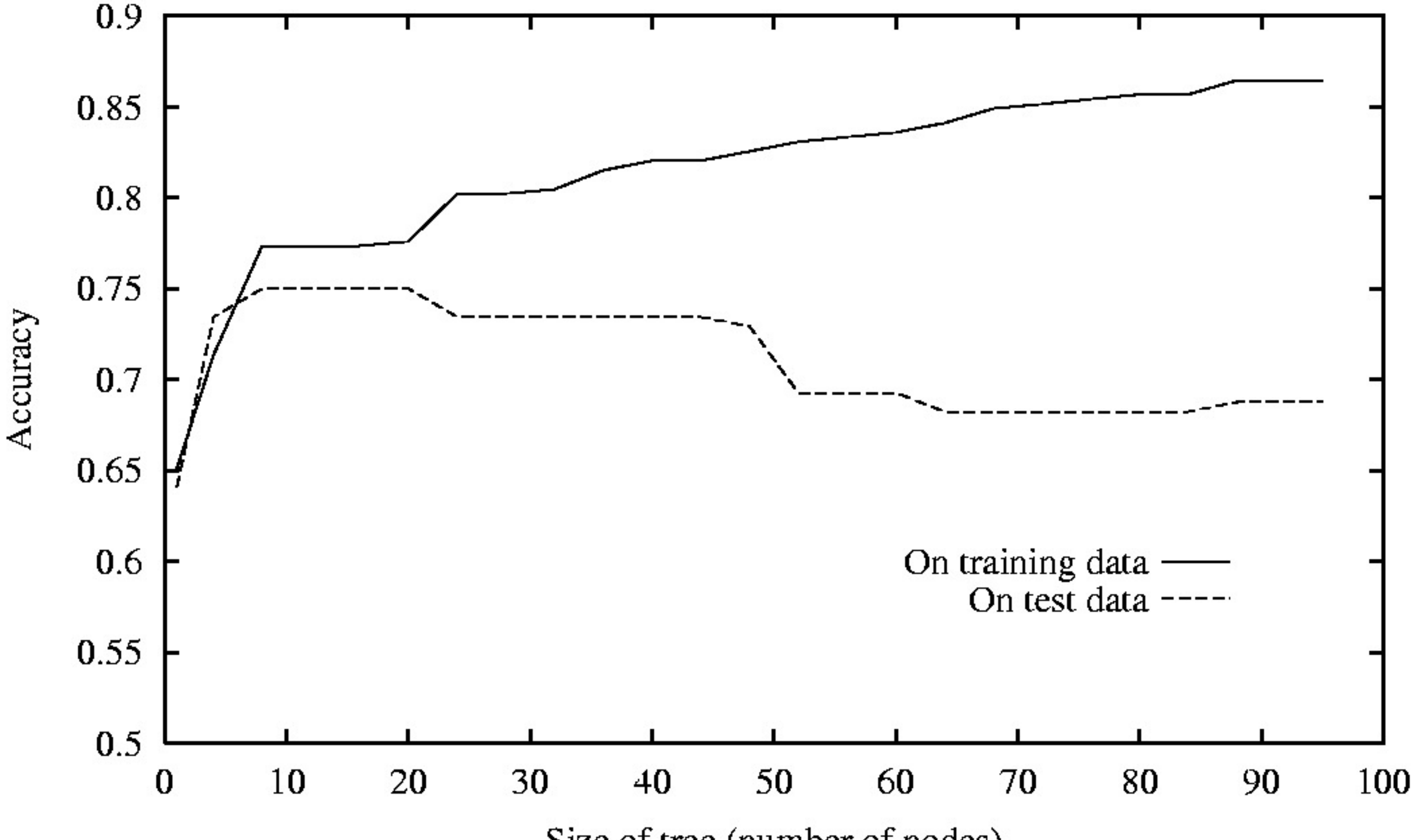
- Training set is a **limited sample.** There might be (combinations of) features that are correlated with the target concept by chance





Overfitting: Tree Size vs. Accuracy

- Tree size vs accuracy



General Phenomenon

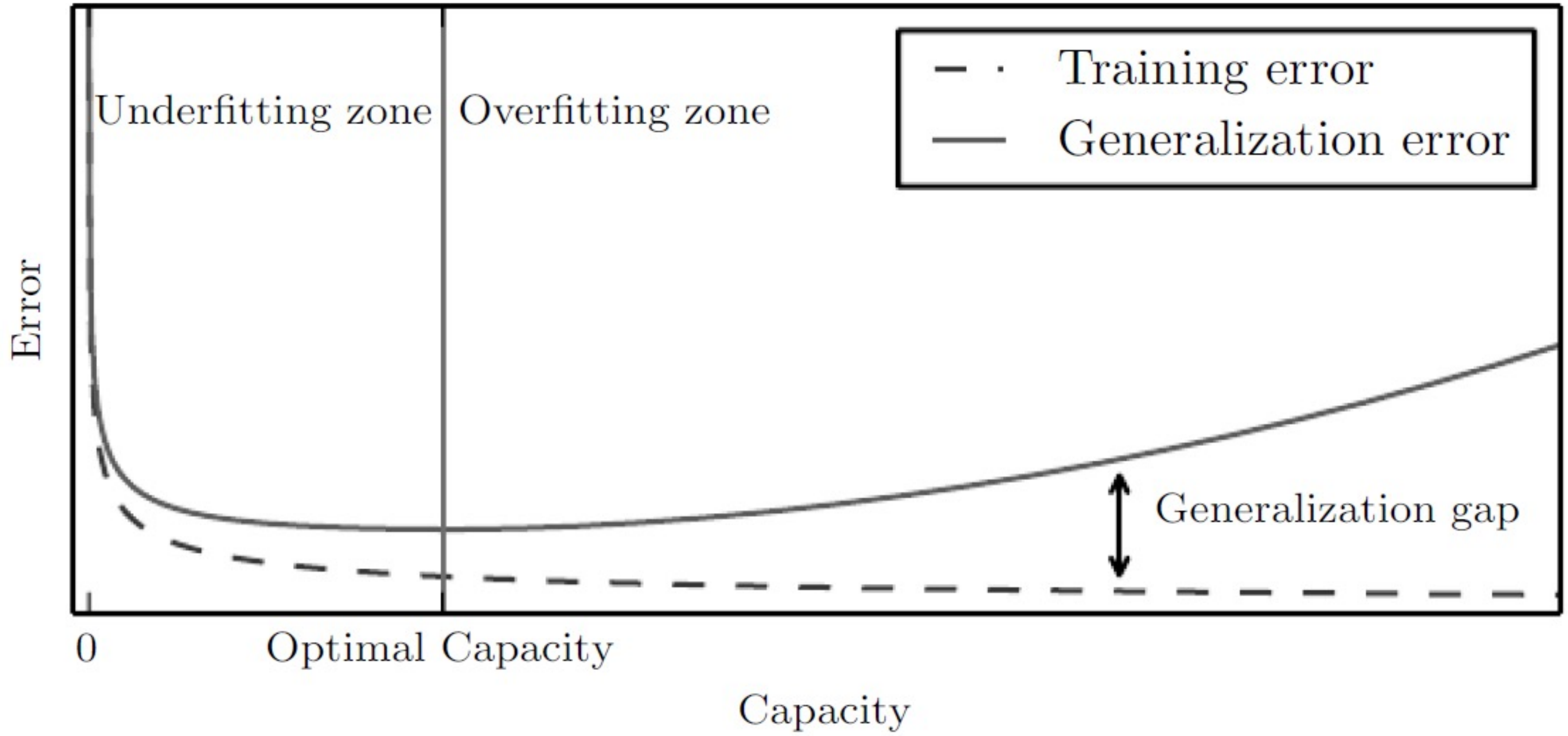


Figure from *Deep Learning*, Goodfellow, Bengio and Courville

DT Learning: Avoiding Overfitting

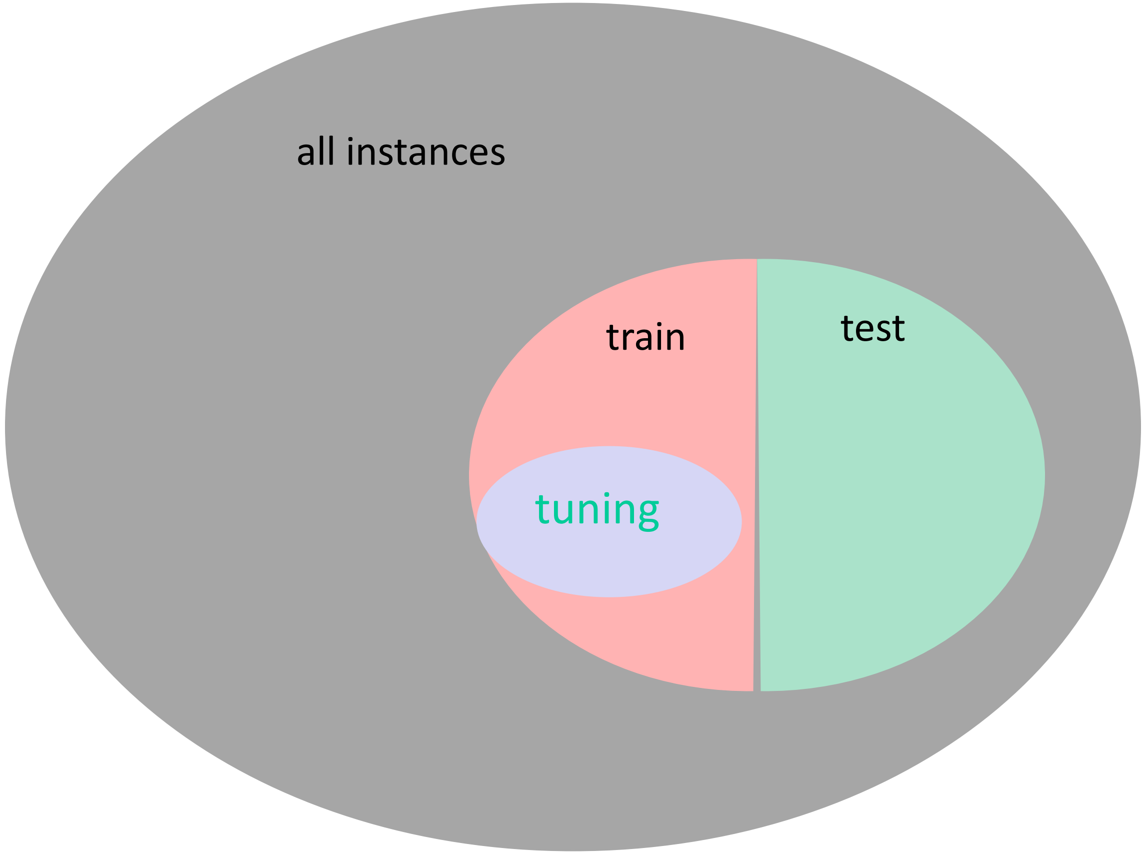
Two **general strategies** to avoid overfitting

1. ***early stopping***: stop if further splitting not justified by a statistical test

2. ***post-pruning***: grow a large tree, then prune back some nodes
    - Ex: evaluate impact on tuning-set accuracy of pruning each node
    - Greedily remove the one that most improves tuning-set accuracy

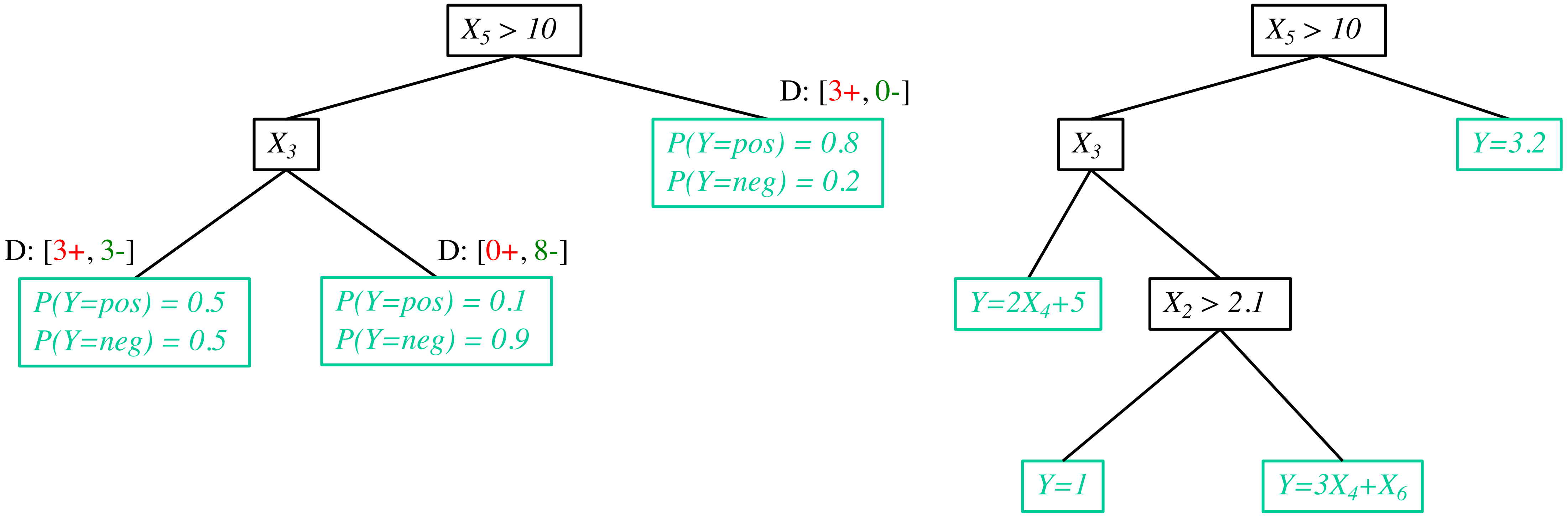
Validation Sets

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



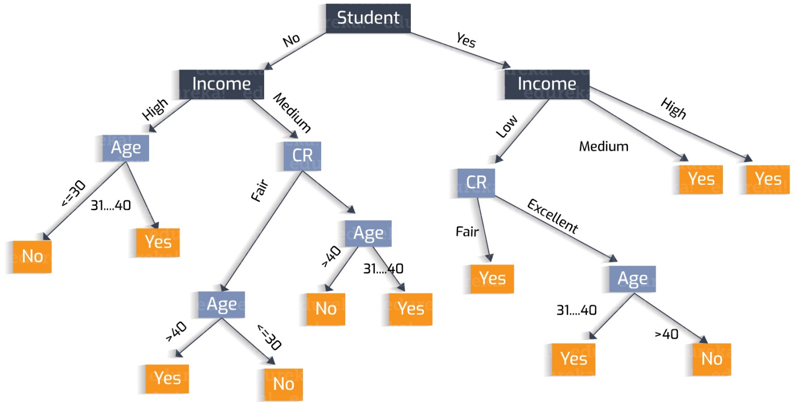
Variations

- Probability estimation trees
  - Leaves: estimate the probability of each class
- Regression trees
  - Either numeric values on leaves, or functions (e.g., linear functions)



Decision Trees: Comments

- Widely used approach
  - Many variations
- Provides humanly comprehensible models
  - When trees not too big
- Insensitive to monotone transformations of numeric features
- Standard methods not suited to on-line setting
- **Usually** not among most accurate learning methods



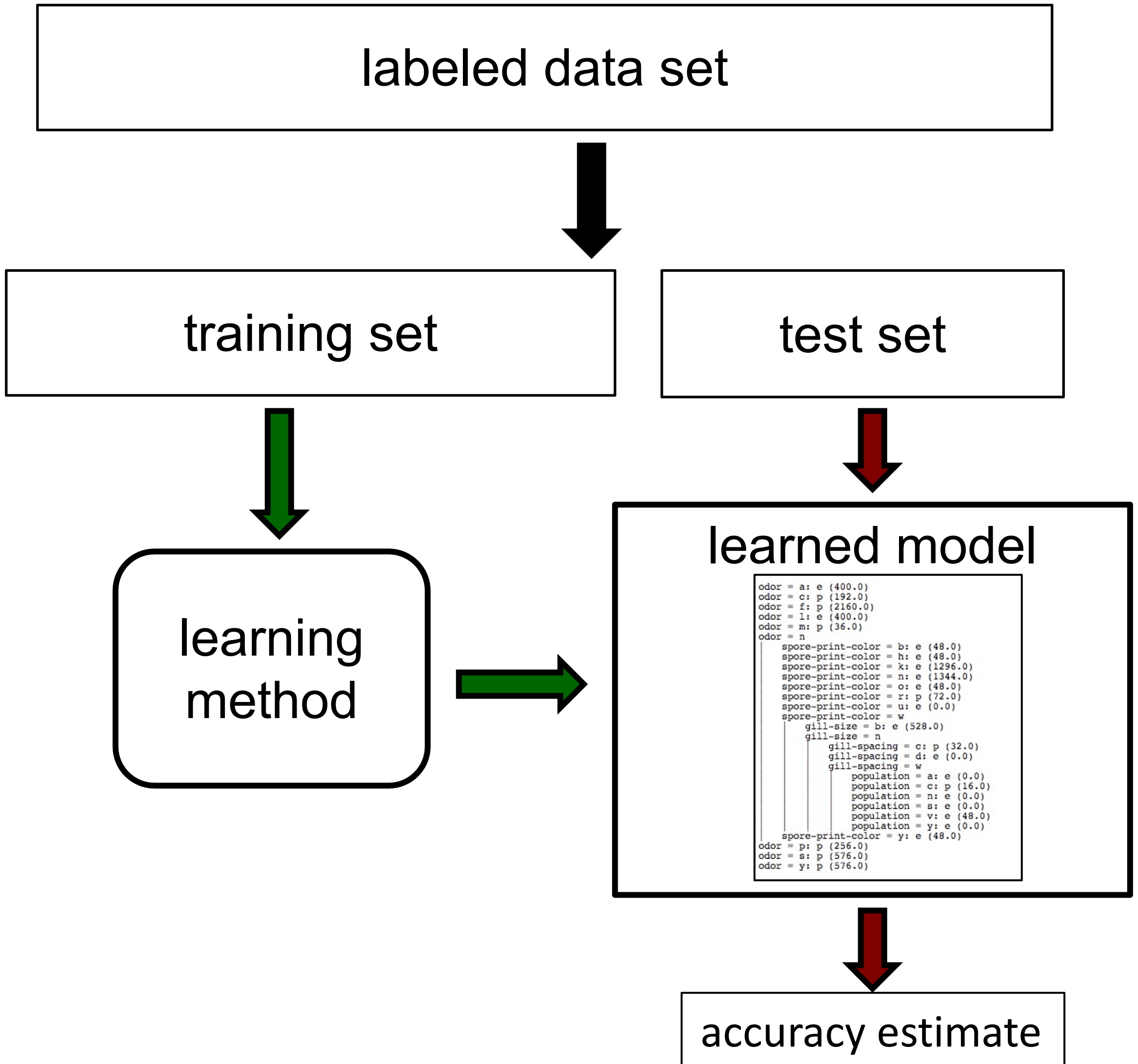
Break & Quiz

Outline

- **Continuing from last time: Decision trees**
  - Information gain, stopping criteria, overfitting, pruning, variations
- **Evaluation: Generalization**
  - Train/test split, random sampling, cross validation
- **Evaluation: Metrics**
  - Confusion matrices, ROC curves, precision/recall

Bias: Accuracy of a Model

- How can we get an **unbiased** estimate of the accuracy of a learned model?



- Unbiased estimate of θ

$$\mathbb{E}[\hat{\theta}] = \theta$$

Bias: Using a Test Set

- How can we get an unbiased estimate of the accuracy of a learned model?
  - When learning a model, you should pretend that you don't have the test data yet (it is "in the mail")
  - If the test-set labels influence the learned model in any way, accuracy estimates will be **biased**

- **Don't train on the test set!**

Bias: Learning Curves

- Accuracy of a method as a function of the train set size?
  - Plot ***learning curves***

Training/test set partition

- for each sample size s on learning curve
  - (optionally) repeat n times
    - randomly select s instances from training set
    - learn model
    - evaluate model on test set to determine accuracy a
    - plot (s, a) or (s, avg. accuracy and error bars)

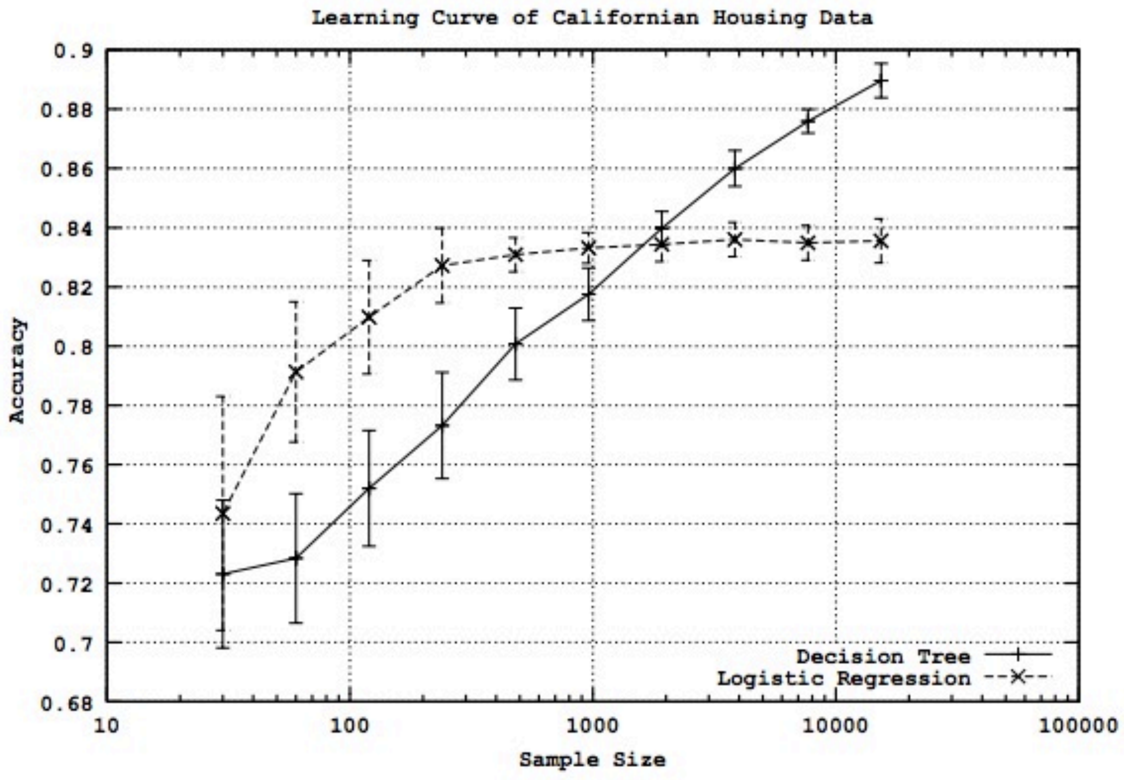


Figure from Perlich et al. *Journal of Machine Learning Research*, 2003

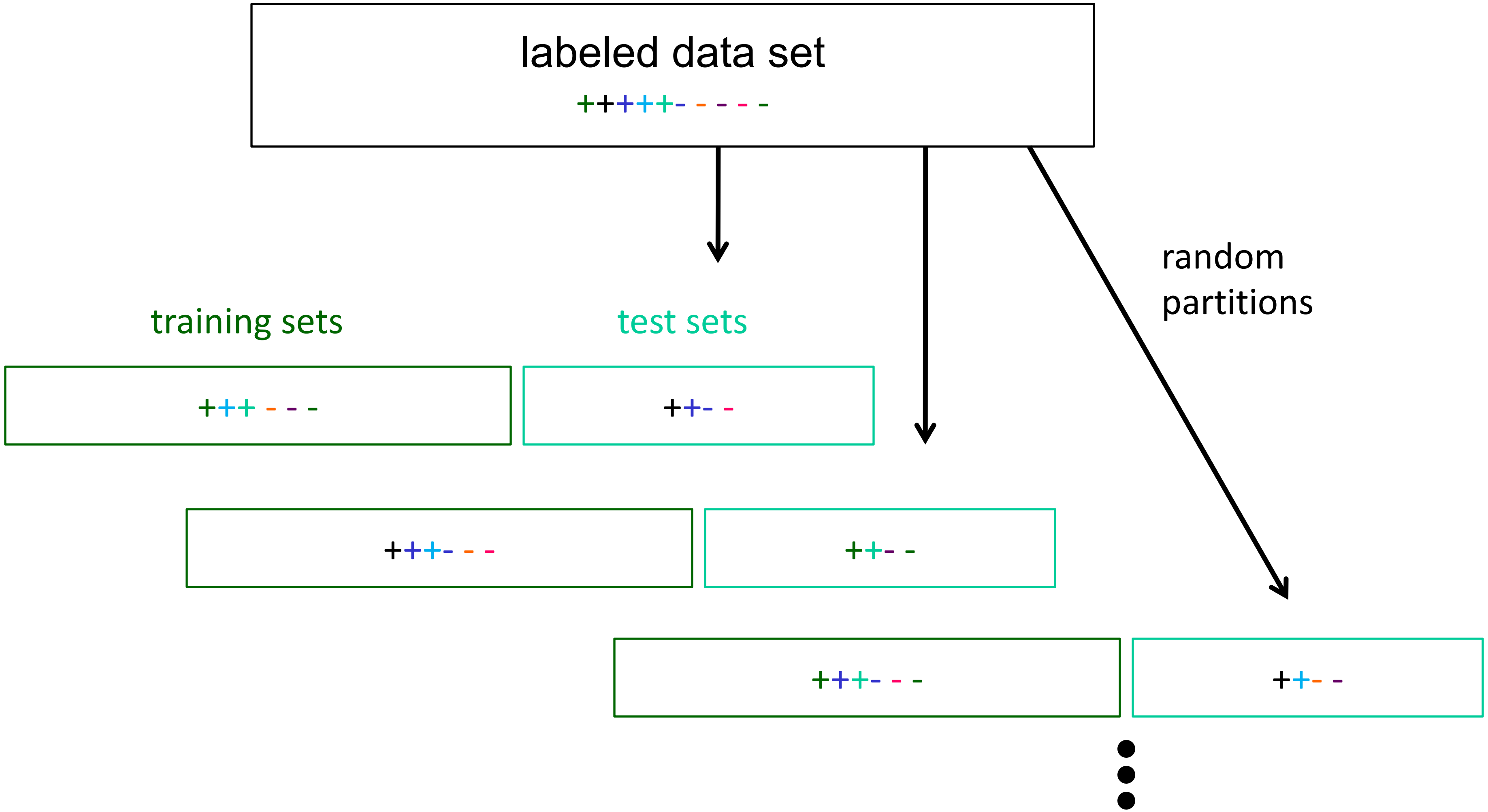
Single Train/Test Split: Limitations

- May not have enough data for sufficiently large training/test sets
  - A **larger test set** gives us more reliable estimate of accuracy (i.e. a lower variance estimate)
  - But… a **larger training set** will be more representative of how much data we actually have for learning process

- A single training set does not tell us how sensitive accuracy is to a particular training sample

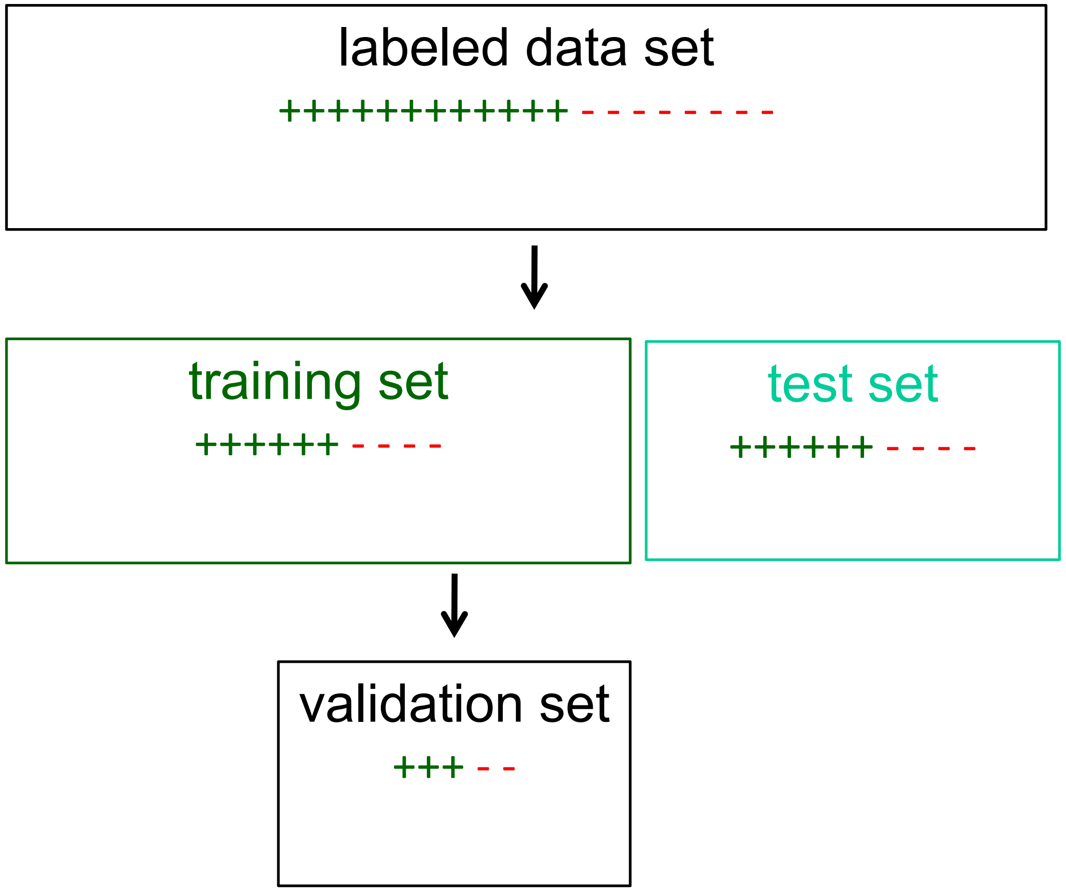
Strategy I: Random Resampling

- Address the second issue by repeatedly randomly partitioning the available data into training and test sets.



Strategy I: Stratified Sampling

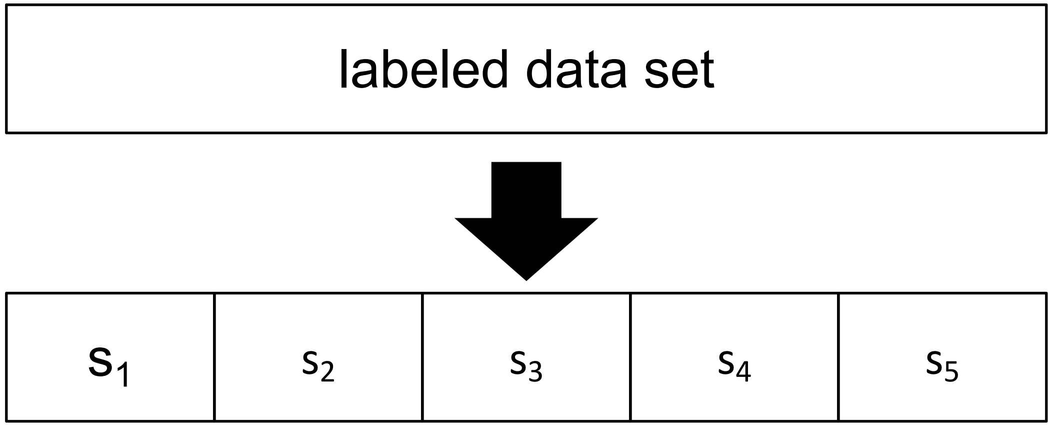
- When randomly selecting training or validation sets, we may want to ensure that **class proportions** are maintained in each selected set



This can be done via stratified sampling: first stratify instances by class, then randomly select instances from each class proportionally.

Strategy II: Cross Validation

Partition data into *n* subsamples



Iteratively leave one subsample out for the test set, train on the rest

iteration	**train on**	**test on**
1	s_2 s_3 s_4 s_5	s_1
2	s_1 s_3 s_4 s_5	s_2
3	s_1 s_2 s_4 s_5	s_3
4	s_1 s_2 s_3 s_5	s_4
5	s_1 s_2 s_3 s_4	s_5

Strategy II: Cross Validation Example

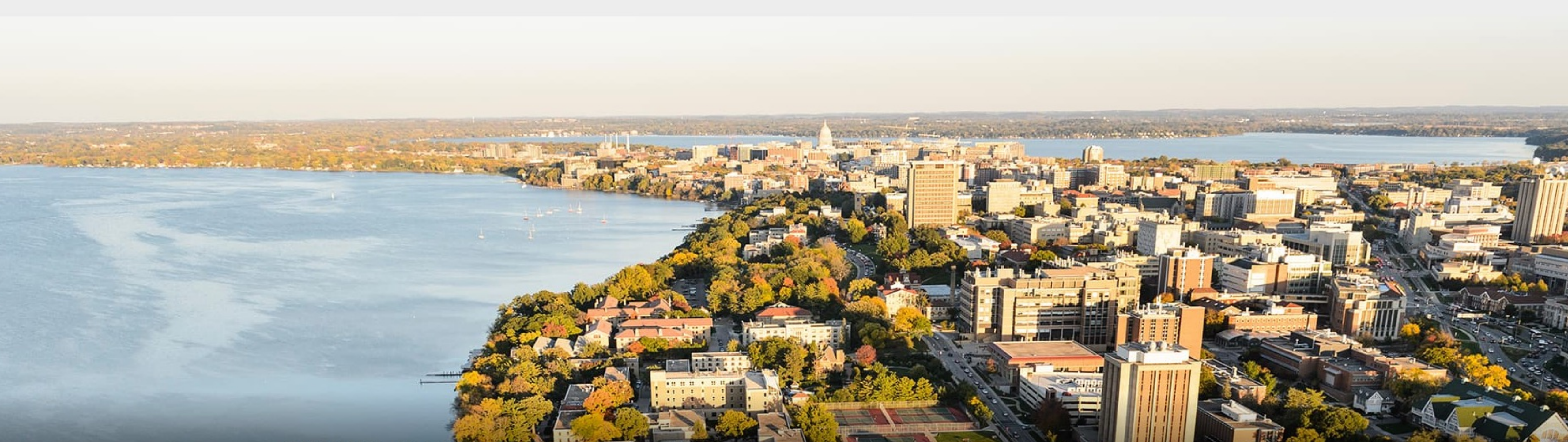
- Suppose we have 100 instances, and we want to estimate accuracy with cross validation

iteration	train on	test on	correct
1	s_2 s_3 s_4 s_5	s_1	11 / 20
2	s_1 s_3 s_4 s_5	s_2	17 / 20
3	s_1 s_2 s_4 s_5	s_3	16 / 20
4	s_1 s_2 s_3 s_5	s_4	13 / 20
5	s_1 s_2 s_3 s_4	s_5	16 / 20

accuracy = 73/100 = 73%

Strategy II: Cross Validation Tips

- 10-fold cross validation is common, but smaller values of *n* are often used when learning takes a lot of time
- in *leave-one-out* cross validation, *n* = # instances
- in *stratified* cross validation, stratified sampling is used when partitioning the data
- CV makes efficient use of the available data for testing
- note that whenever we use multiple training sets, as in CV and random resampling, we are evaluating a <u>learning method</u> as opposed to an <u>individual learned hypothesis</u>



Break & Quiz

Outline

- **Continuing from last time: Decision trees**
  - Information gain, stopping criteria, overfitting, pruning, variations
- **Evaluation: Generalization**
  - Train/test split, random sampling, cross validation
- **Evaluation: Metrics**
  - Confusion matrices, ROC curves, precision/recall

Beyond Accuracy: Confusion Matrices

- How can we understand what types of mistakes a learned model makes?

task: activity recognition from video

actual class

	bend	jack	jump	pjump	run	side	skip	walk	wave1	wave2
bend	100	0	0	0	0	0	0	0	0	0
jack	0	100	0	0	0	0	0	0	0	0
jump	0	0	89	0	0	0	11	0	0	0
pjump	0	0	0	100	0	0	0	0	0	0
run	0	0	0	0	89	0	11	0	0	0
side	0	0	0	0	0	100	0	0	0	0
skip	0	0	0	0	0	0	100	0	0	0
walk	0	0	0	0	0	0	0	100	0	0
wave1	0	0	0	0	0	0	0	0	67	33
wave2	0	0	0	0	0	0	0	0	0	100

predicted class

Confusion Matrices: 2-Class Version

		actual class: positive	actual class: negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

$$\text{accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

$$\text{error} = 1 - \text{accuracy} = \frac{FP + FN}{TP + FP + FN + TN}$$

Accuracy: Sufficient?

Accuracy may not be useful measure in cases where

- There is a large class skew
  - Is 98% accuracy good when 97% of the instances are negative?
- There are differential misclassification costs – say, getting a positive wrong costs more than getting a negative wrong
  - Consider a medical domain in which a false positive results in an extraneous test but a false negative results in a failure to treat a disease
- We are most interested in a subset of high-confidence predictions

Other Metrics

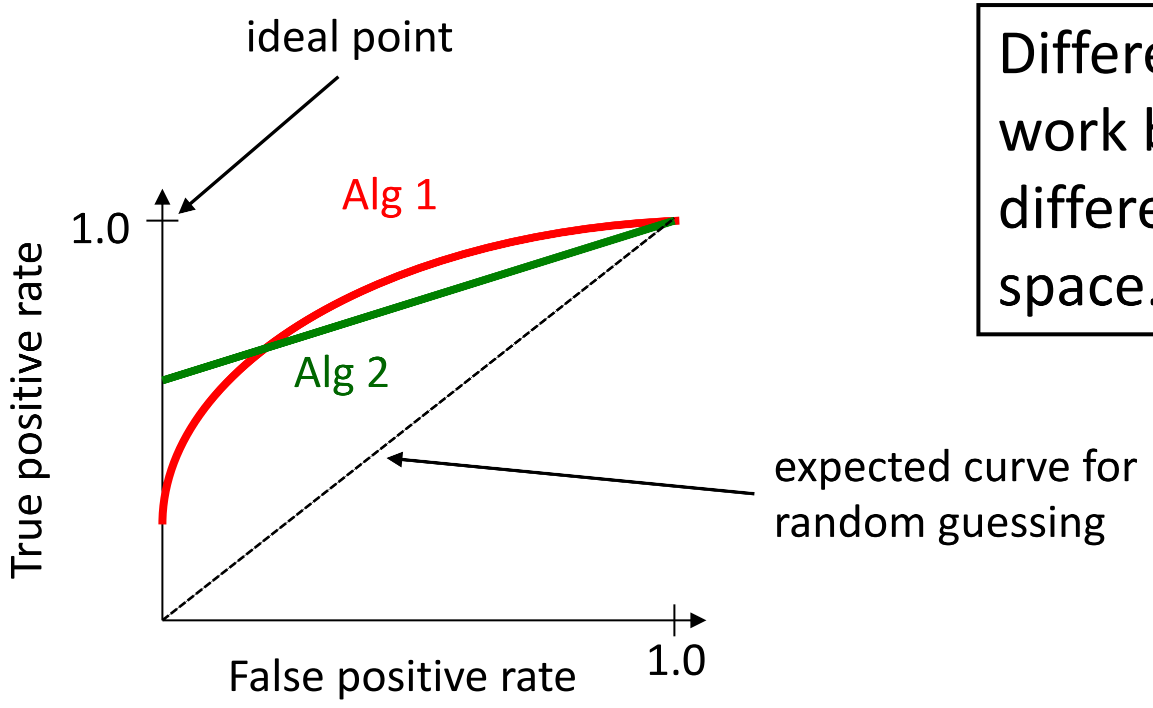
		actual class: positive	actual class: negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

$$\text{true positive rate (recall)} = \frac{\text{TP}}{\text{actual pos}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{false positive rate} = \frac{\text{FP}}{\text{actual neg}} = \frac{\text{FP}}{\text{TN} + \text{FP}}$$

Other Metrics: ROC Curves

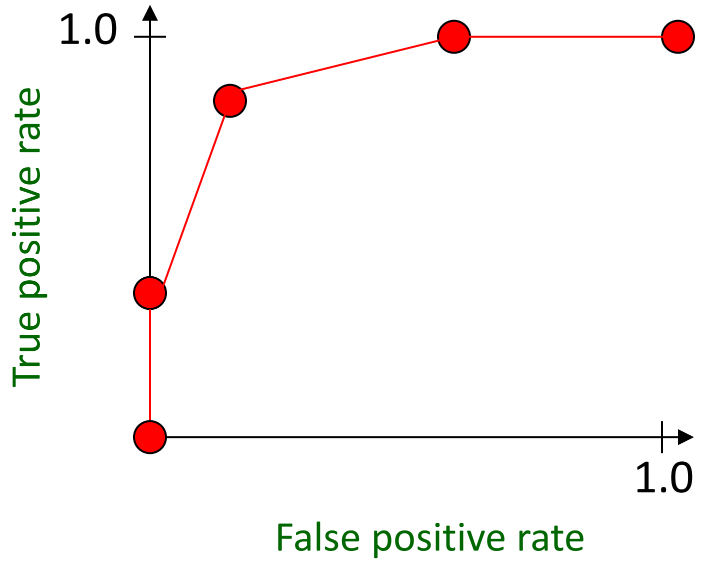
- A *Receiver Operating Characteristic* (*ROC*) curve plots the TP-rate vs. the FP-rate as a threshold on the confidence of an instance being positive is varied



Different methods can work better in different parts of ROC space.

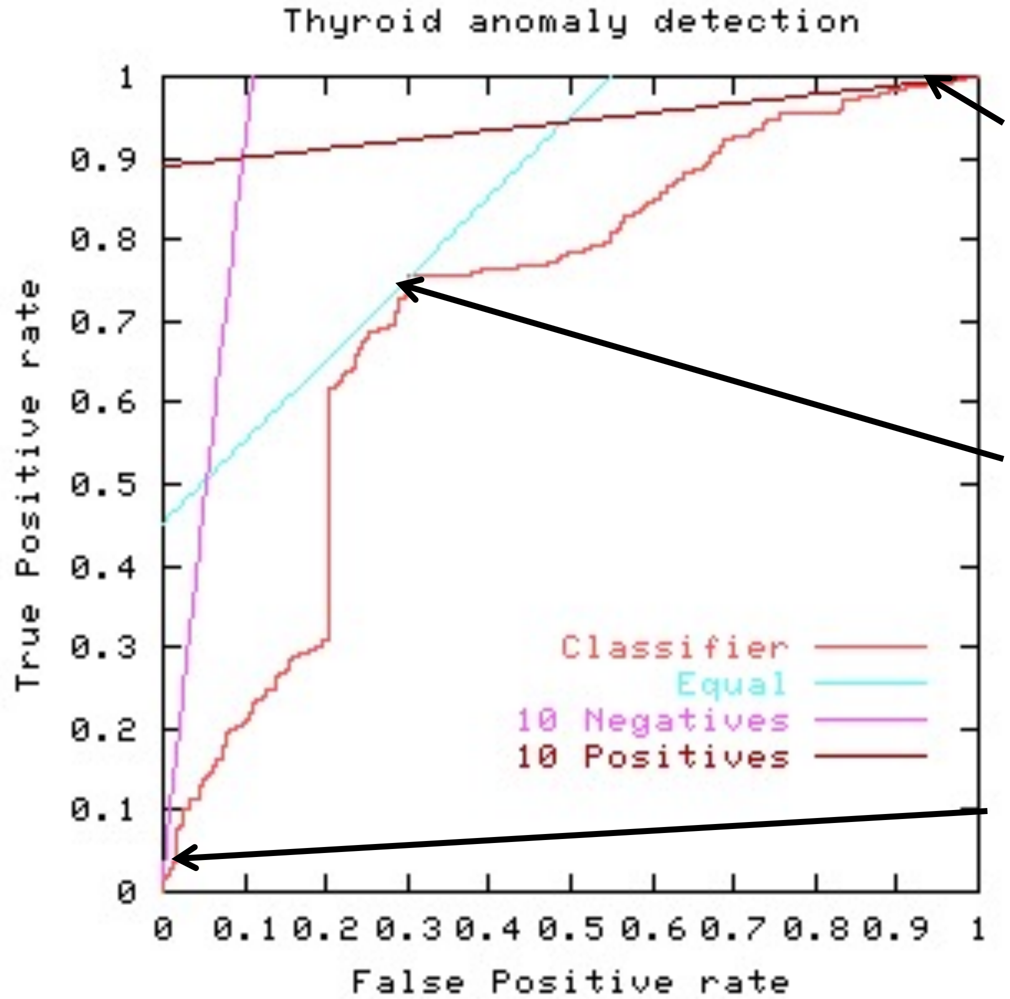
ROC Curves: Plotting

instance	confidence positive		correct class
Ex 9	.99		+
Ex 7	.98	TPR= 2/5, FPR= 0/5	+
Ex 1	.72		-
Ex 2	.70		+
Ex 6	.65	TPR= 4/5, FPR= 1/5	+
Ex 10	.51		-
Ex 3	.39		-
Ex 5	.24	TPR= 5/5, FPR= 3/5	+
Ex 4	.11		-
Ex 8	.01	TPR= 5/5, FPR= 5/5	-



ROC Curves: Misclassification Cost

- The best operating point depends on relative cost of FN and FP misclassifications



best operating point when
FN costs 10× FP

best operating point when
cost of misclassifying positives and
negatives is equal

best operating point when
FP costs 10× FN

Other Metrics: Precision

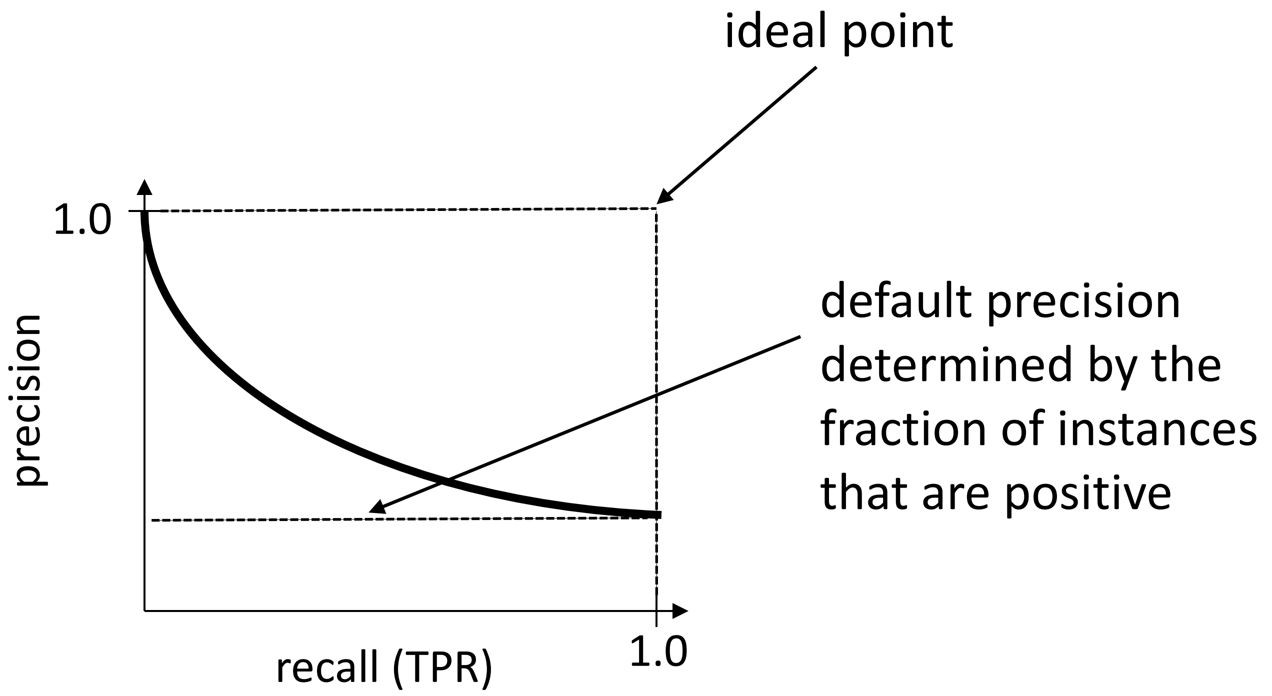
		actual class: positive	actual class: negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

$$\text{recall (TP rate)} = \frac{\text{TP}}{\text{actual pos}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{precision (positive predictive value)} = \frac{\text{TP}}{\text{predicted pos}} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Other Metrics: Precision/Recall Curve

- A *precision/recall curve* (TP-rate): threshold on the confidence of an instance being positive is varied



predicting patient risk for VTE

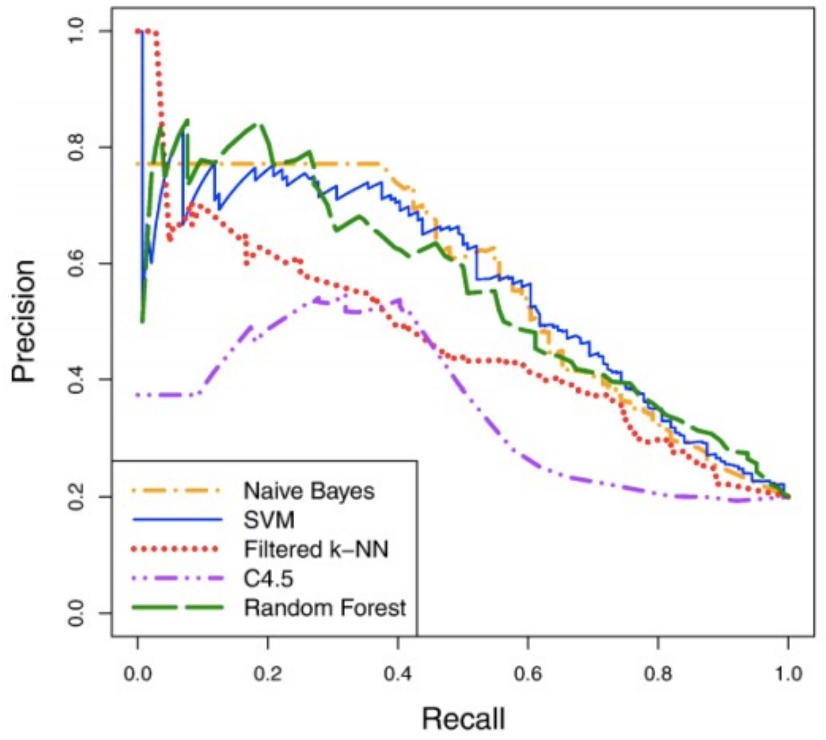


figure from Kawaler et al., *Proc. of AMIA Annual Symposium*, 2012

ROC vs. PR curves

Both

- Allow predictive performance to be assessed at various levels of confidence
- Assume binary classification tasks
- Sometimes summarized by calculating *area under the curve*

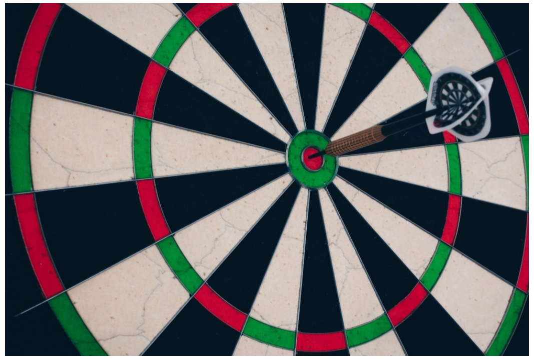
ROC curves

- Insensitive to changes in class distribution (ROC curve does not change if the proportion of positive and negative instances in the test set are varied)
- Can identify optimal classification thresholds for tasks with differential misclassification costs

Precision/recall curves

- Show <u>the fraction of predictions </u>that are false positives
- Well suited for tasks with lots of negative instances

Confidence Intervals



- Back to looking at accuracy on new data.
- **Scenario**:
  - For some model *h*, a test set S with *n* samples
  - We have *h* producing *r* errors out of *n*.
  - Our estimate of the error rate: $\text{error}_S(h) = r/n$

- With C% probability, true error is in interval

$$error_S(h) \pm z_C \sqrt{\frac{error_S(h)(1 - error_S(h))}{n}}$$

  - z_C depends on C. For 95% confidence, it is ~1.96

Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, Pedro Domingos, Jerry Zhu, Yingyu Liang, Volodymyr Kuleshov