CS 760: Machine Learning
Model Selection & Evaluation

University of Wisconsin-Madison

Outline

- **Wrapping up decision trees**
  - Review, variations, information gain, regression
  - Evaluation in decision trees: overfitting, pruning
- **Evaluation: Measuring generalization**
  - Train/test split, random sampling, cross validation
- **Evaluation: Performance metrics**
  - Confusion matrices, ROC curves, precision/recall

Outline

- **Wrapping up decision trees**
  - Review, variations, information gain, regression
  - Evaluation in decision trees: overfitting, pruning
- **Evaluation: Measuring generalization**
  - Train/test split, random sampling, cross validation
- **Evaluation: Performance metrics**
  - Confusion matrices, ROC curves, precision/recall

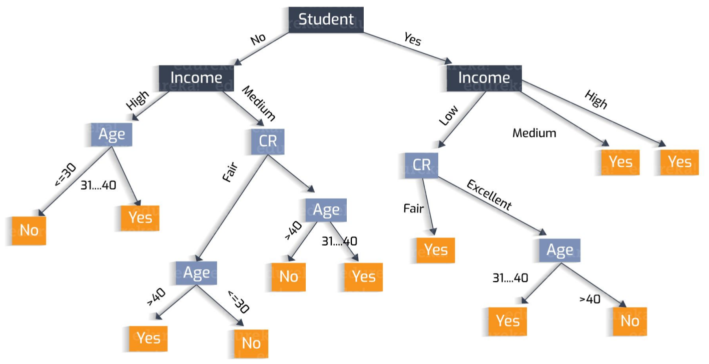
[Review] **Decision Trees:** Learning

- **Learning Algorithm:**

```
MakeSubtree(set of training instances $D$)
    $C$ = DetermineCandidateSplits($D$)
    if stopping criteria is met
        make a leaf node $N$
        determine class label/probabilities for $N$
    else
        make an internal node $N$
        $S$ = FindBestSplit($D$, $C$)
        for each outcome $k$ of $S$
            $D_k$ = subset of instances that have outcome $k$
            $k^{th}$ child of $N$ = MakeSubtree($D_k$)
    return subtree rooted at $N$
```

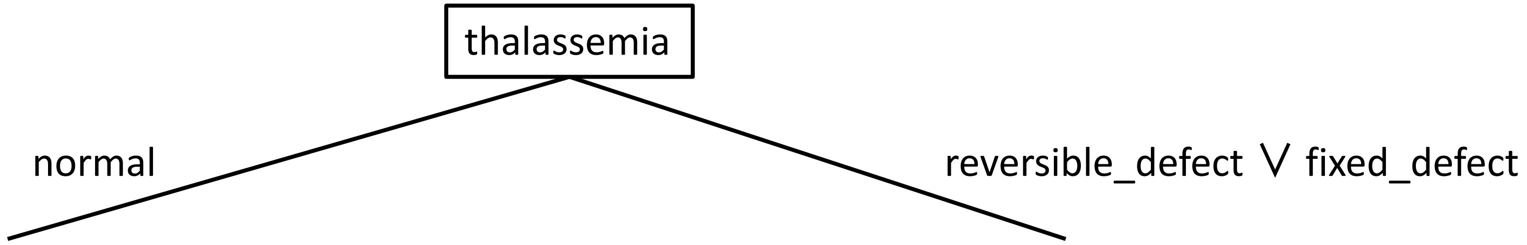
$$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$$

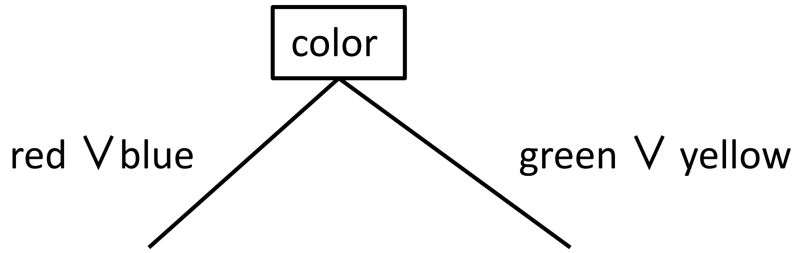
$$\Downarrow$$



[Review] **Candidate Splits -** Nominal Features

Instead of using k-way splits for k-valued features, could require binary splits on all nominal features
(Classification and Regression Trees / CART does this)

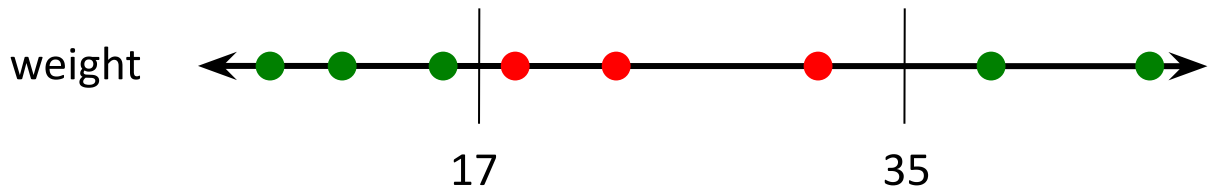


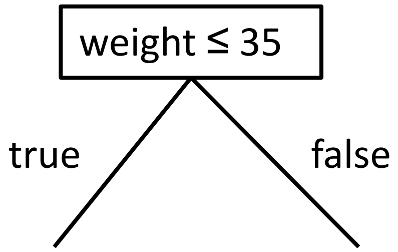


[Review] **Candidate Splits** - Numeric Features

Given a set of training instances D and a specific feature X_i

- Sort the values of X_i in D
- Evaluate split thresholds in intervals between instances of different classes





- Do this for every numeric feature and add it to the candidate splits

[Review] **Find Best Split**

Hypothesis: simplest tree that classifies the training instances accurately will generalize

Occam's razor: "when you have two competing theories that make the same predictions, the simpler one is the better"

Want to choose split S that maximizes

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

i.e. mutual information.

[Review] InfoGain Limitations

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

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

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

Stopping Criteria

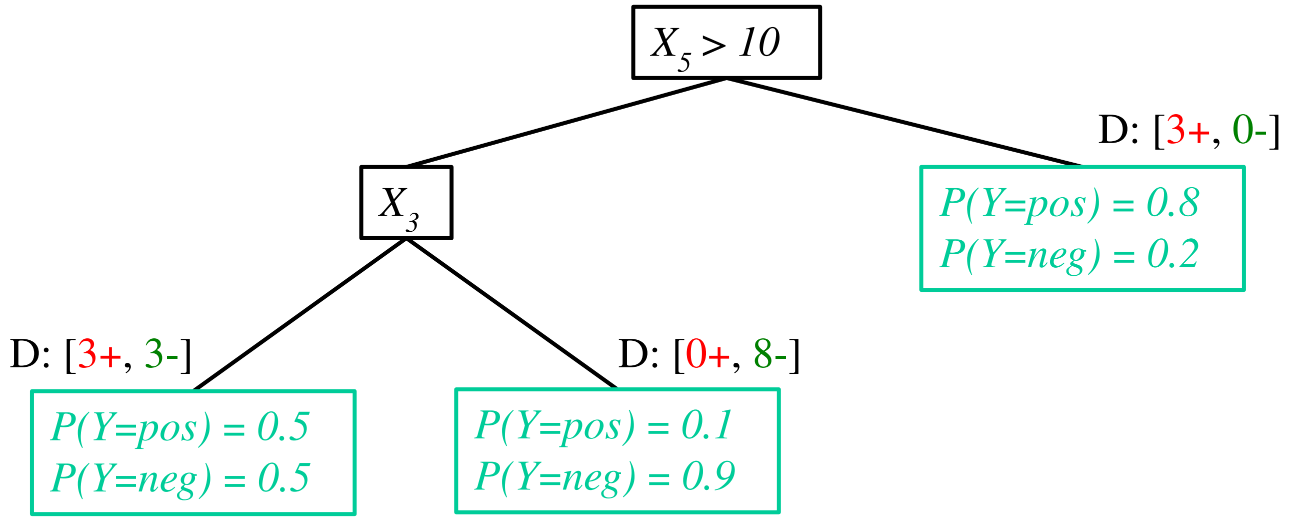
Some ideas

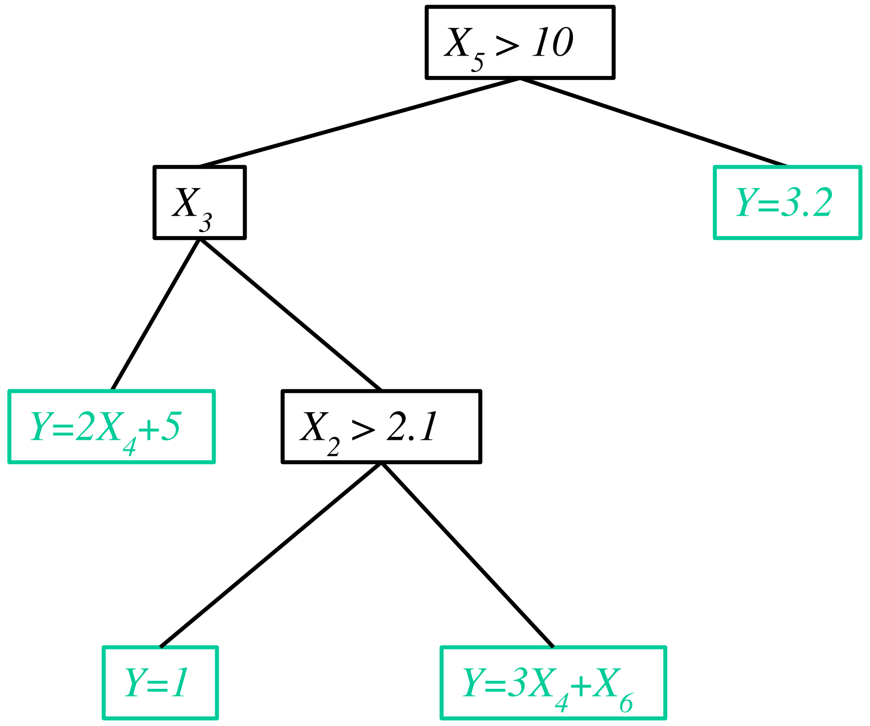
- Stop when you reach a single data point?
- Stop when the subset of instances are all in the same class?
- Stop when we a large fraction of the instances are all in the same class?
- We have exhausted all of the candidate splits

Stop earlier?

Variations

- Probability estimation trees
  - Leaves: estimate the probability of each class
- Regression trees
  - Either numeric values (e.g. average label) or functions (e.g. linear functions) at each leaf.





Variations

- Probability estimation trees
  - Leaves: estimate the probability of each class
- Regression trees
  - Either numeric values (e.g. average label) or functions (e.g. linear functions) at each leaf.
- Tree ensembles
  - Random forests [Breiman, 2001]
  - XGBoost [Chen & Guestrin, 2016]
    - Winner of many Kaggle competitions

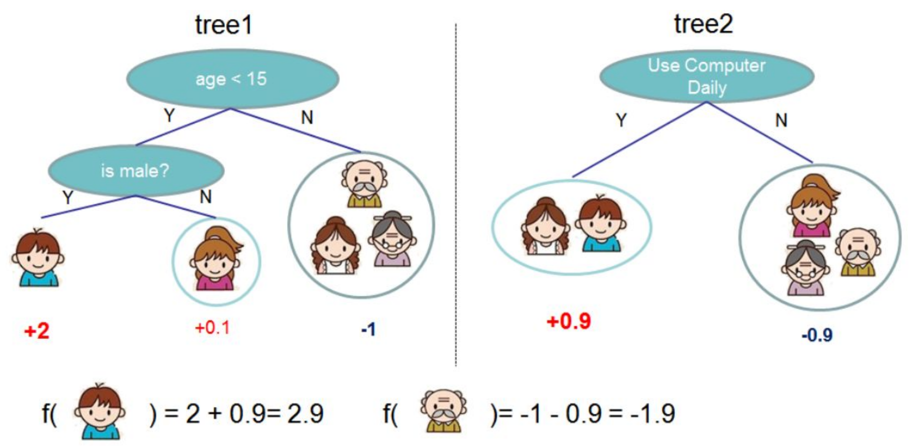


Figure 1: Tree Ensemble Model. The final prediction for a given example is the sum of predictions from each tree.

Decision Trees: Comments

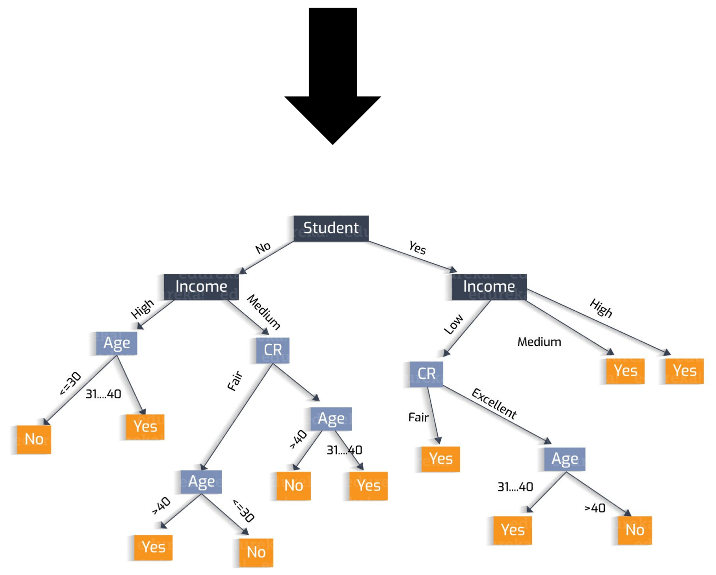
- Widely used approach
  - Many variations
- Provides humanly comprehensible models
  - Not true for big trees / tree ensembles
- Insensitive to monotone transformations of numeric features
- Implementation can (and does) vary, performance may depend on specific choices.

Decision Trees: Learning

- **Learning Algorithm:**

```
MakeSubtree(set of training instances $D$)
    $C$ = DetermineCandidateSplits($D$)
    if stopping criteria is met
        make a leaf node $N$
        determine class label/probabilities for $N$
    else
        make an internal node $N$
        $S$ = FindBestSplit($D$, $C$)
        for each outcome $k$ of $S$
            $D_k$ = subset of instances that have outcome $k$
            $k^{th}$ child of $N$ = MakeSubtree($D_k$)
    return subtree rooted at $N$
```

$$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$$



Break & Quiz

Q1-1: How many distinct (binary classification) decision trees are possible with *4* Boolean attributes? Here distinct means representing different functions.

1. 2^4
2. 2^8
3. 2^{16}
4. 2^{32}

Q1-1: How many distinct (binary classification) decision trees are possible with *4* Boolean attributes? Here distinct means representing different functions.

1. 2^4
2. 2^8
3. [✓ 2^{16}]
4. 2^{32}

#distinct decision trees
= #distinct Boolean functions
= #functions of 2^4 = 16 inputs, binary label for each input
= $\mathbf{2^{16}}$

Model Selection in Decision Trees

Evaluation: Accuracy

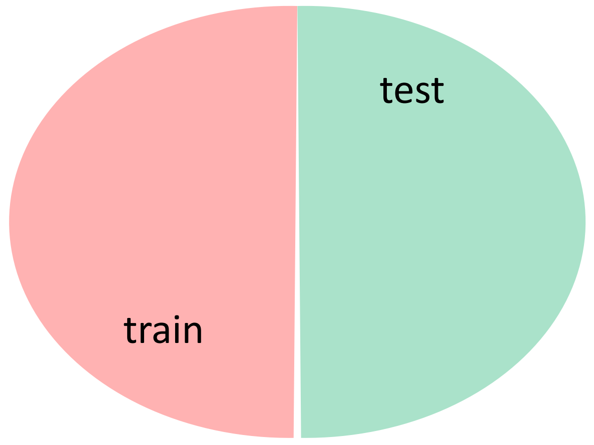
- Can we just calculate the fraction of training instances that are correctly classified?
- Consider a problem domain in which instances are assigned labels at random with $P(Y = 1) = 0.5$
    - How accurate would it be on its training set, if you stop when all instances are in the same class? (training accuracy = 100%)
    - How accurate would a learned decision tree be on previously unseen instances?

- Recall: our goal is to do well on *future data*.

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**



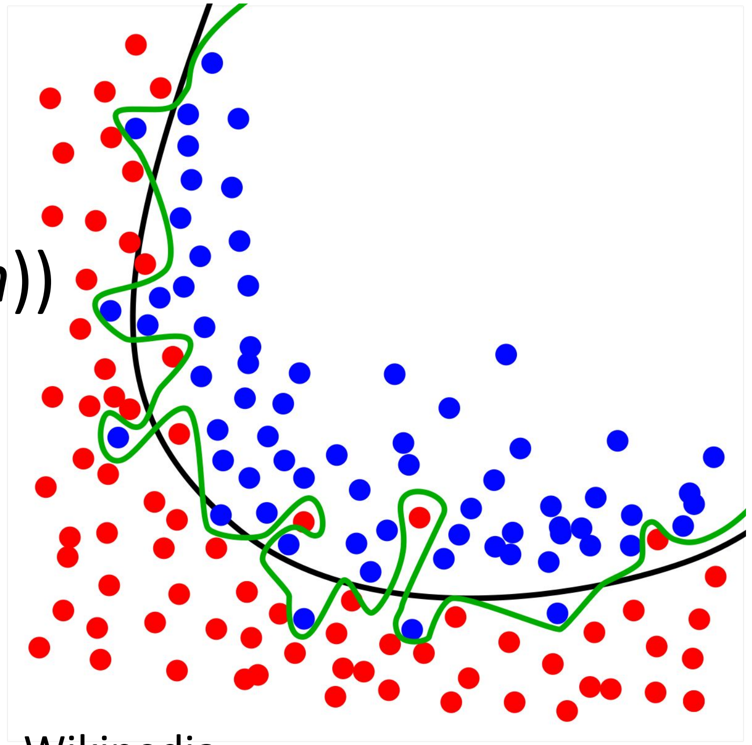
Overfitting

Notation: error of model *h* over

- training data: $\text{error}_D(h)$
- entire distribution of data: $\text{error}_P(h)$

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

- low error on the training data (low $\text{error}_D(h)$)
- high error on the entire distribution (high $\text{error}_P(h)$)


Wikipedia

Overfitting Example: Noisy Data

(unknown) Target function is $Y = X_1 \wedge X_2$

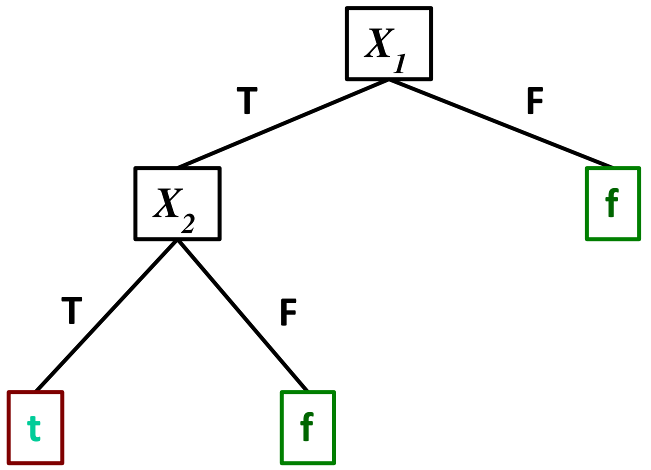
- There is **noise** in some feature values
- Training set:

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

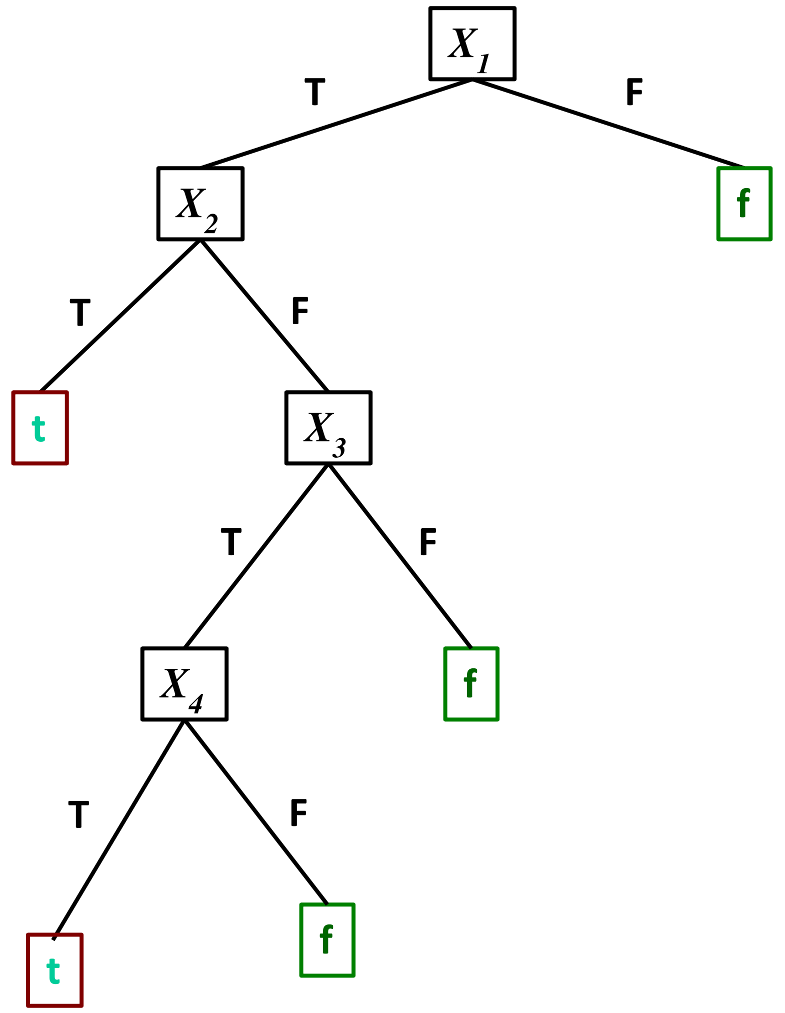
noisy value

Overfitting Example: Noisy Data

Correct tree



Tree that fits noisy training data



Overfitting Example: Noise-Free Data

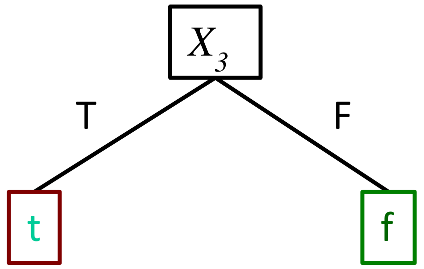
Target function is $Y = X_1 \wedge X_2$

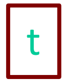
- P(X_3 = t) = 0.5 for both classes
- P(Y = t) = 0.67
- Training set:

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



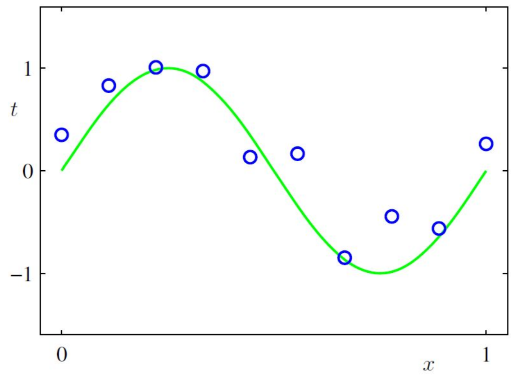


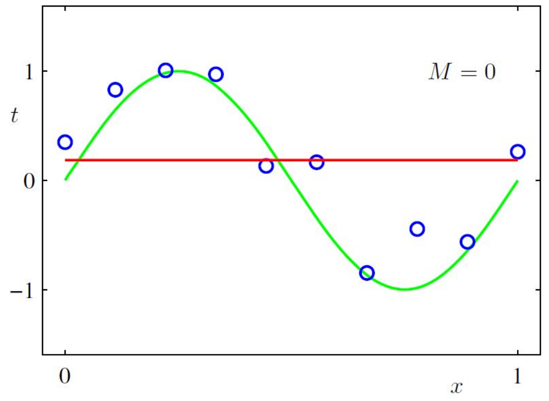
	Training set accuracy	**Test set accuracy**
Tree splitting on X_3	100%	50%
Single leaf t	66%	66%

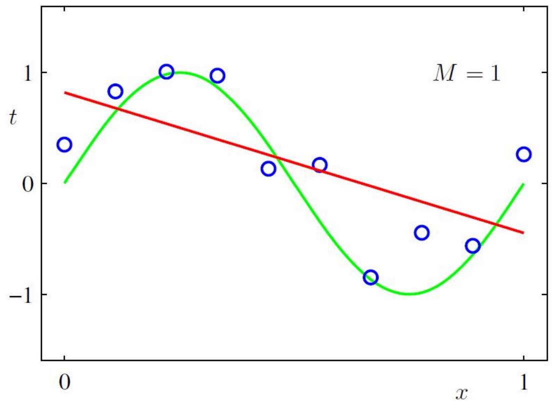
X_3	Y
t	t
t	t
t	t
f	f
f	f

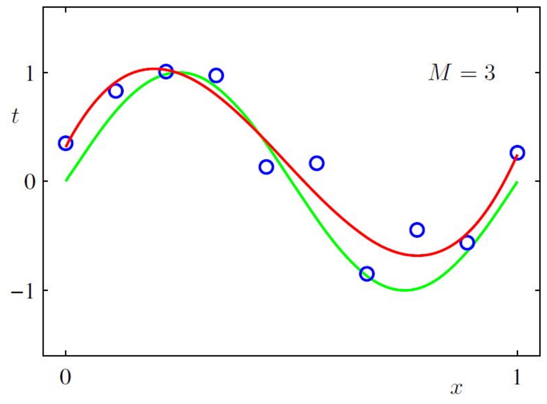
Overfitting Example: Polynomial Regression

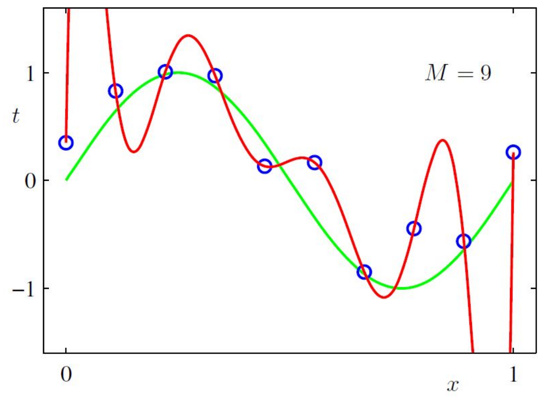
- Is higher-degree = better?





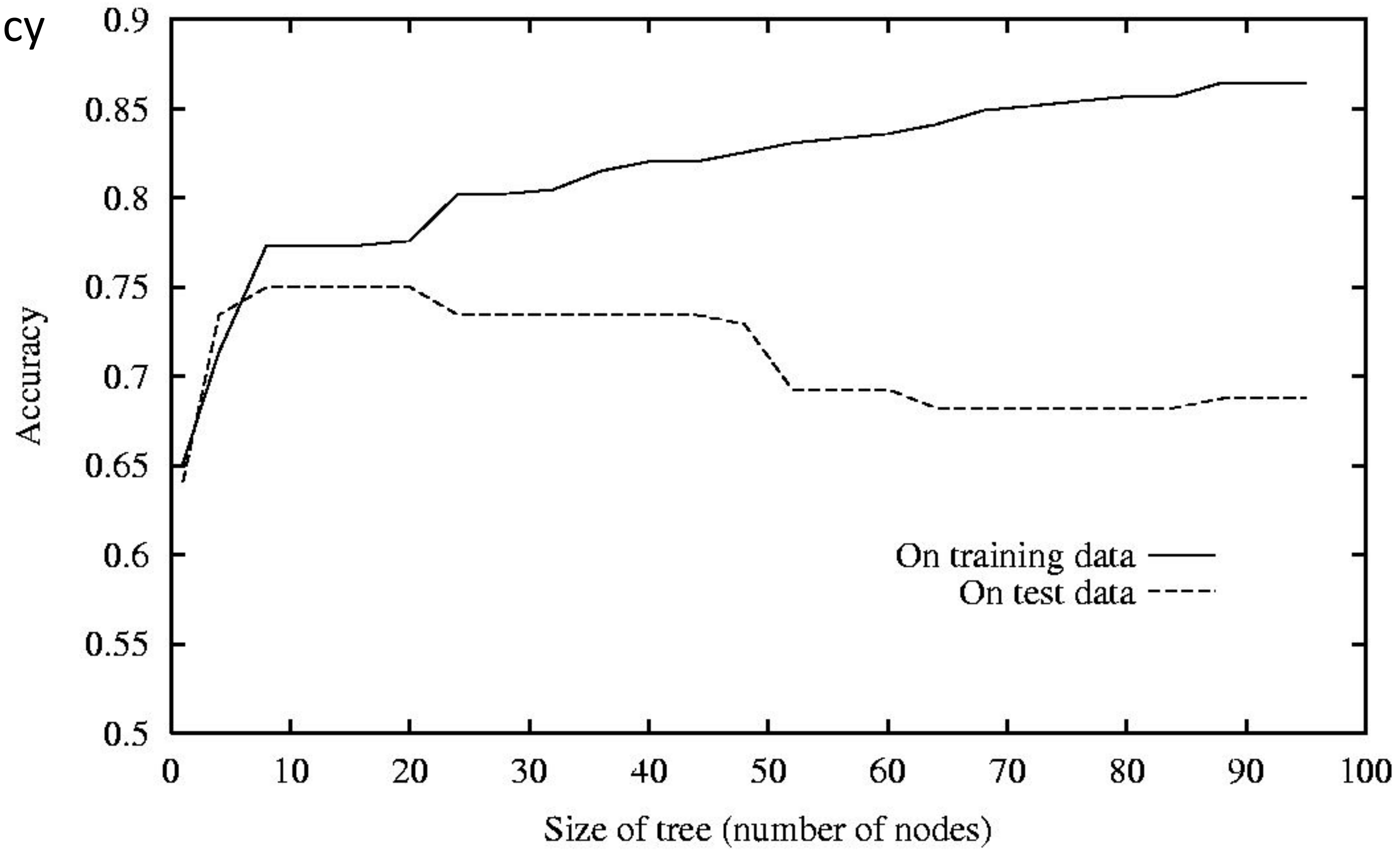






Overfitting: Tree Size vs. Accuracy

- Tree size vs accuracy



General Phenomenon

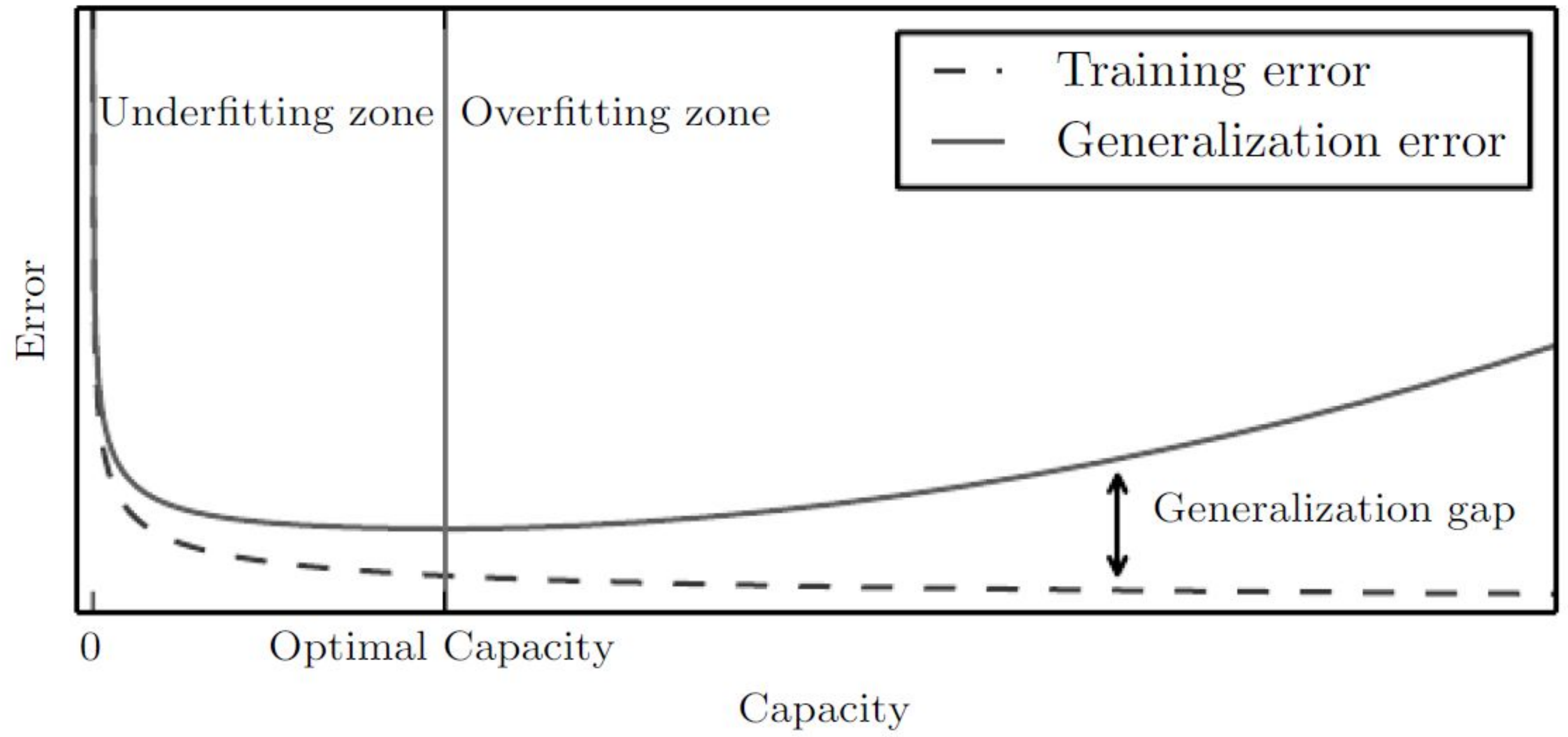
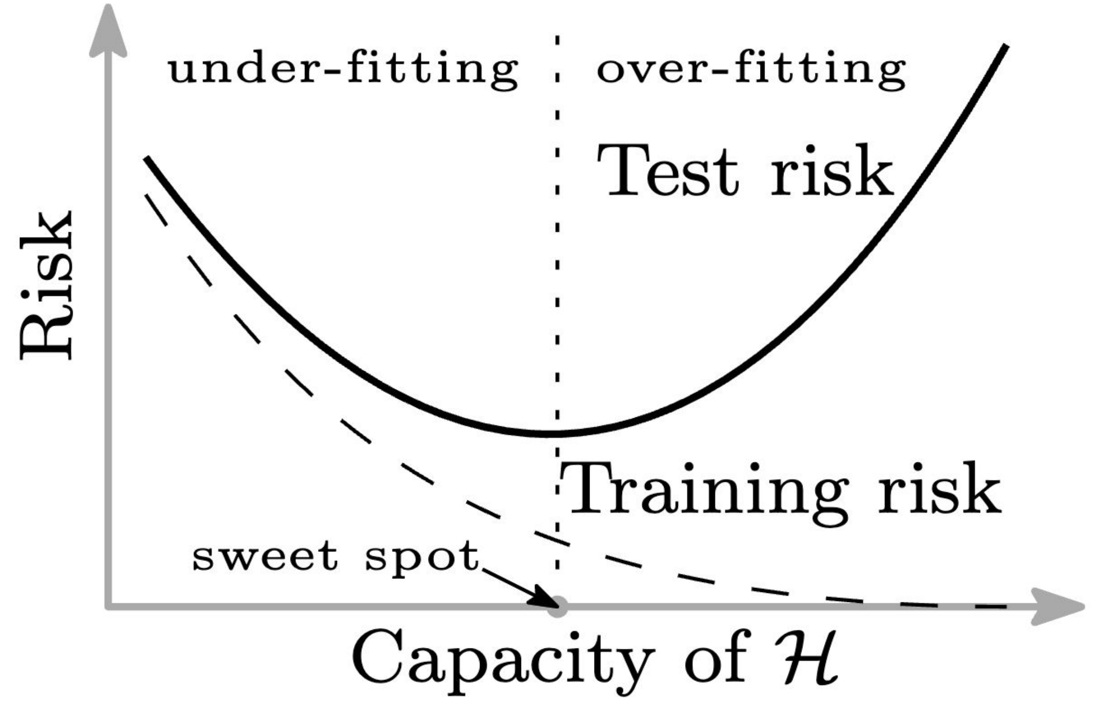


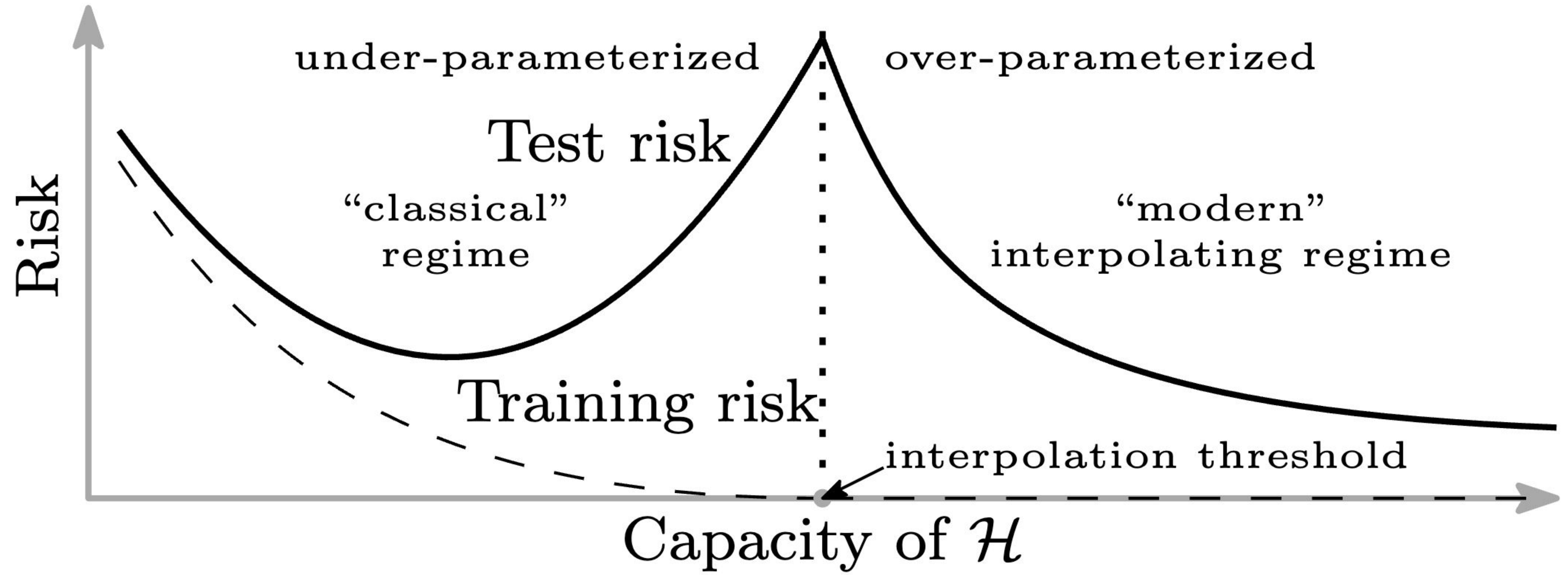
Figure from *Deep Learning*, Goodfellow, Bengio and Courville

General Phenomenon



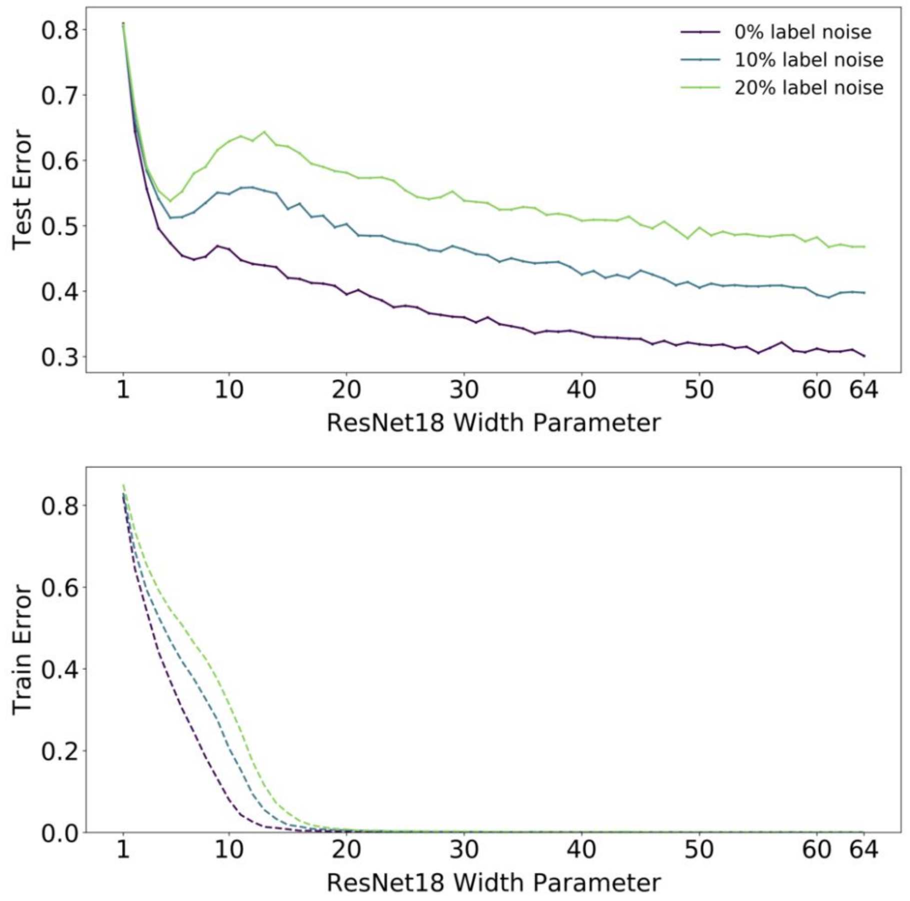
Reconciling modern machine learning practice and the bias-variance trade-off. Belkin et al

Modern Understanding – Double Descent

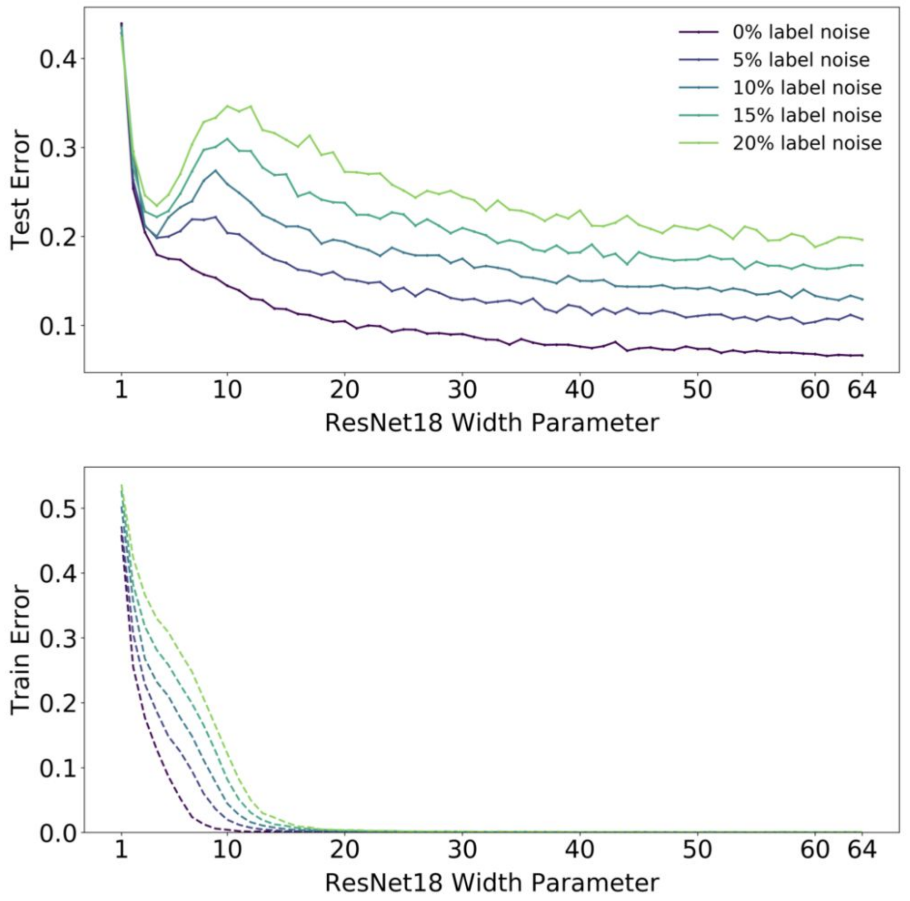


Belkin et al. Reconciling modern machine learning practice and the bias-variance trade-off.

Modern Understanding – Double Descent



(a) **CIFAR-100.** There is a peak in test error even with no label noise.

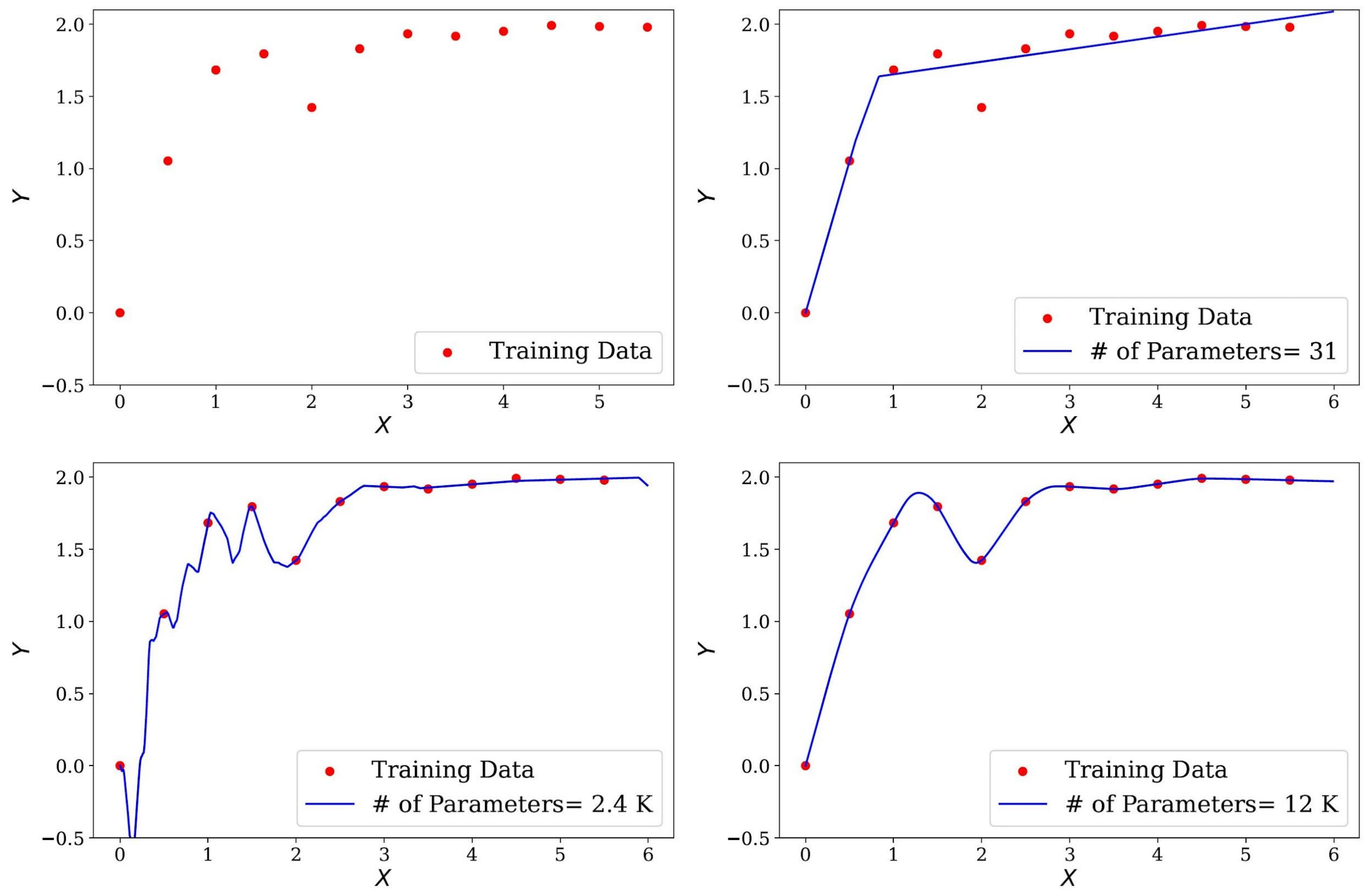


(b) **CIFAR-10.** There is a "plateau" in test error around the interpolation point with no label noise, which develops into a peak for added label noise.

Nakkiran et al. *Deep Double Descent: Where Bigger Models and More Data Hurt.*

Modern Understanding – Double Descent

even our curve-fitting example might not be the full story!



Decision Tree Learning: Avoiding Overfitting

Two **general strategies** to avoid overfitting

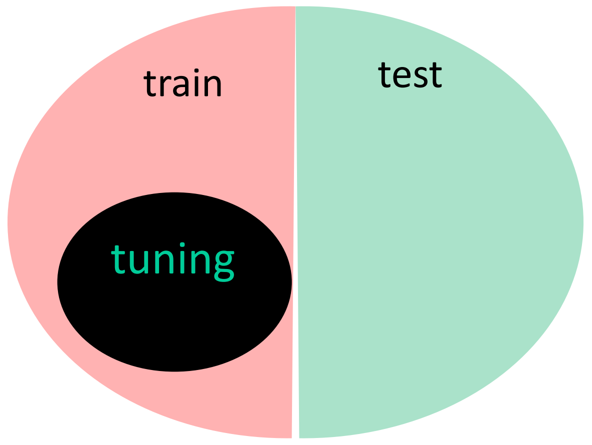
1. ***During training***: create two-way instead of multi-way splits, stop if further splitting not justified by a statistical test

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

Tuning Sets

- A *tuning set* (a.k.a. *validation 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)

- Why can't you use the training set to prune?
- Why can't you use the test set to prune?



Break & Quiz

Q2-2: Which of the following statements is TRUE?

1. If there is no noise, then there is no overfitting.
2. Overfitting may improve the generalization ability of a model.
3. Generalization error is monotone with respect to the capacity/complexity of a model.
4. More training data may help preventing overfitting.

Q2-2: Which of the following statements is TRUE?

1. If there is no noise, then there is no overfitting.
2. Overfitting may improve the generalization ability of a model.
3. Generalization error is monotone with respect to the capacity/complexity of a model.
4. [✓ More training data may help preventing overfitting.]

1. We can still have false correlation that leads to overfitting.
2. Overfitting would undermine the generalization ability.
3. Generalization error often first decreases and then increases as the model capacity increases.
4. Increasing training data size would help better approximate the true distribution.

True or False:

In k-NN, using large k leads to over-fitting.

True or False:

In k-NN, using large k leads to over-fitting.

False!

Outline

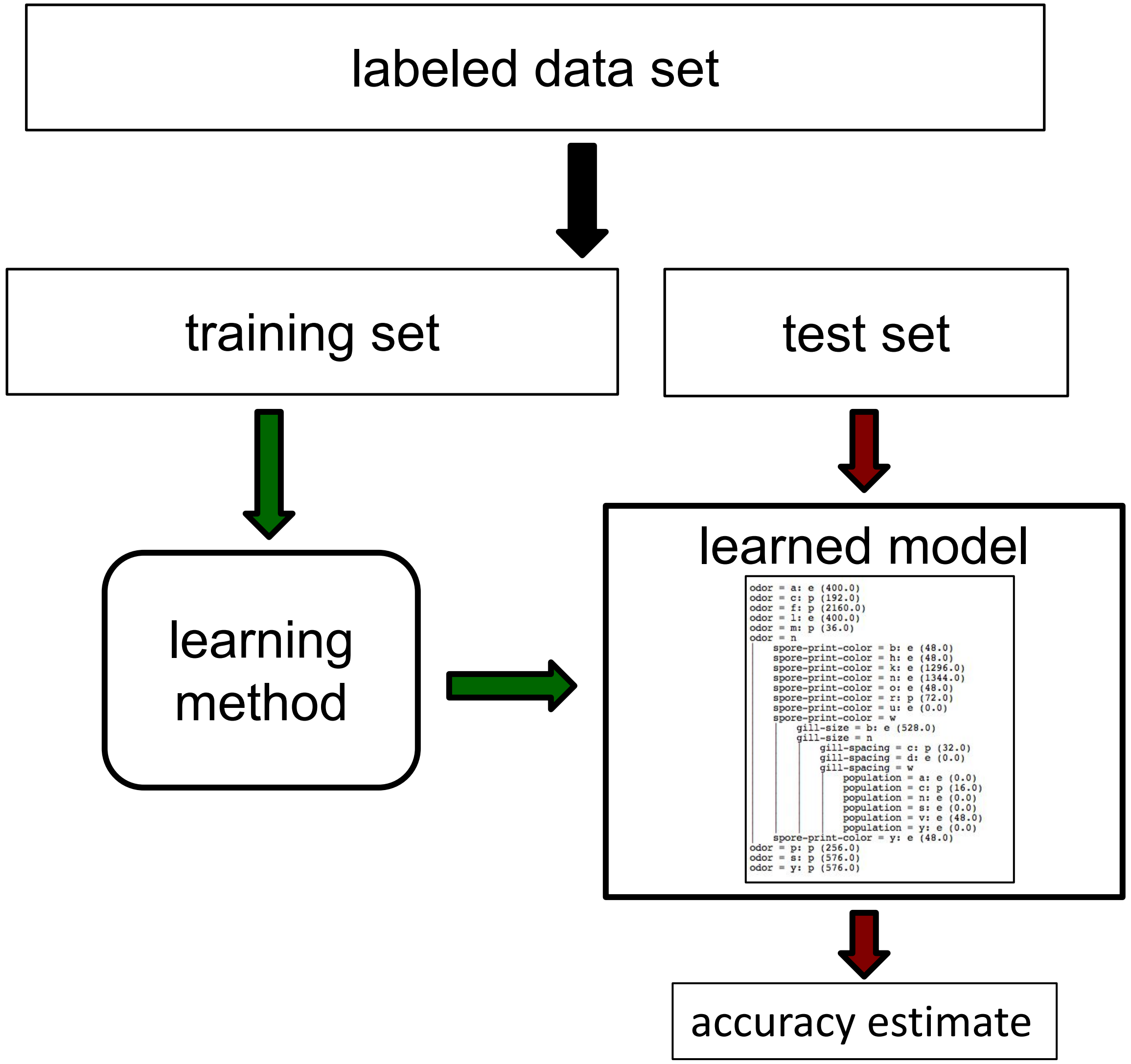
- **Wrapping up decision trees**
  - Review, Variations, information gain, regression
  - Model selection in decision trees: overfitting, pruning, variations
- **Evaluation: Generalization**
  - Train/test split, random sampling, cross validation
- **Evaluation: Metrics**
  - Confusion matrices, ROC curves, precision/recall

Accuracy of a Model

- How can we estimate the accuracy of a learned model?

- Typically: use a statistic $\hat{\theta}$ that is an **unbiased estimator** of θ computed over an **independent** test set

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



```
odor = a: e (400.0)
odor = c: p (192.0)
odor = f: p (2160.0)
odor = l: e (400.0)
odor = m: p (36.0)
odor = n
|   spore-print-color = b: e (48.0)
|   spore-print-color = h: e (48.0)
|   spore-print-color = k: e (1296.0)
|   spore-print-color = n: e (1344.0)
|   spore-print-color = o: e (48.0)
|   spore-print-color = r: p (72.0)
|   spore-print-color = u: e (0.0)
|   spore-print-color = w
|   |   gill-size = b: e (528.0)
|   |   gill-size = n
|   |   |   gill-spacing = c: p (32.0)
|   |   |   gill-spacing = d: e (0.0)
|   |   |   gill-spacing = w
|   |   |   |   population = a: e (0.0)
|   |   |   |   population = c: p (16.0)
|   |   |   |   population = n: e (0.0)
|   |   |   |   population = s: e (0.0)
|   |   |   |   population = v: e (48.0)
|   |   |   |   population = y: e (0.0)
|   spore-print-color = y: e (48.0)
odor = p: p (256.0)
odor = s: p (576.0)
odor = y: p (576.0)
```

Using a Test Set

- How can we estimate the accuracy of a learned model?
  - When learning a model, you should pretend that you don’t have the test data yet
  - If the test-set labels influence the learned model in any way, accuracy estimates will not be correct, as you may have fitted to your test set.

- **Don’t train on the test set!!!**

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)

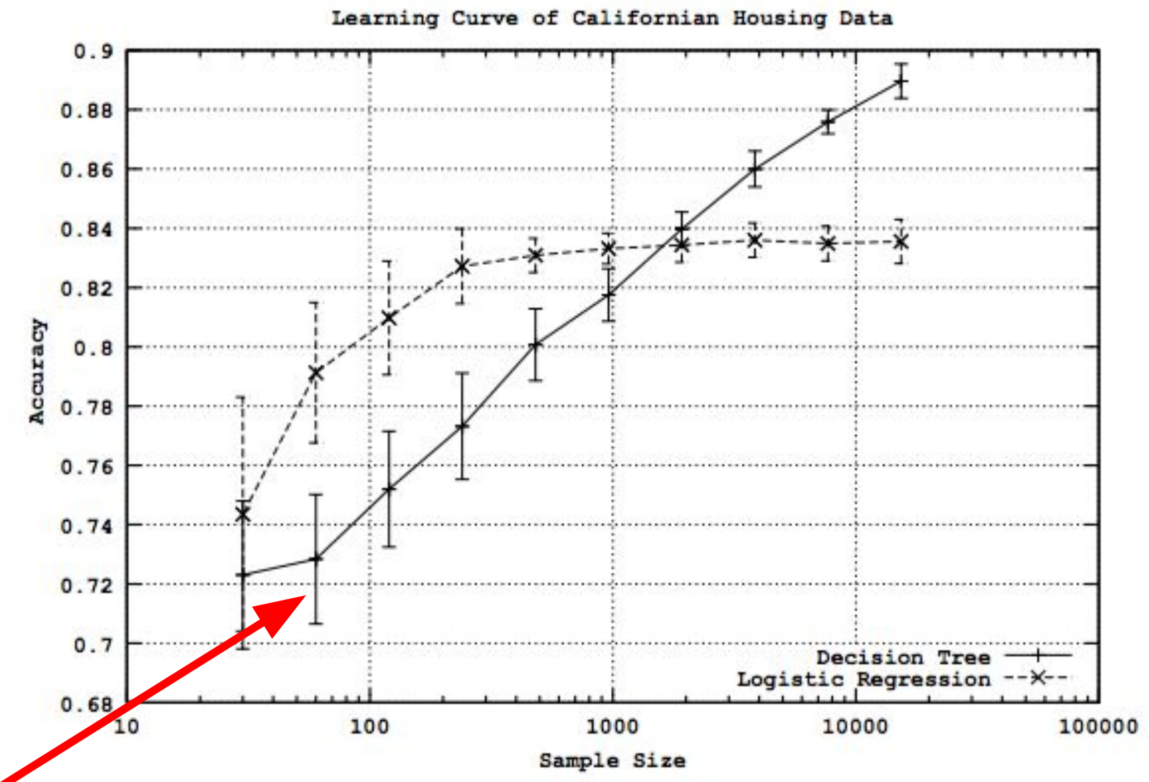


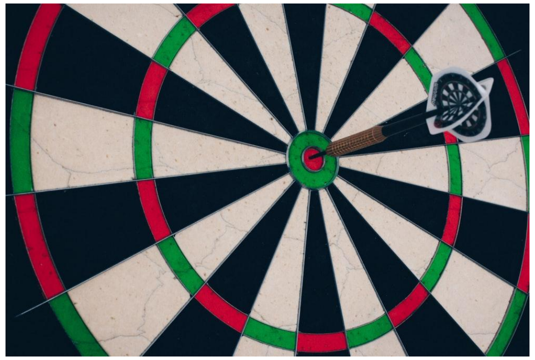
Figure from Perlich et al. *Journal of Machine Learning Research*, 2003

What are these intervals?

Confidence Intervals

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, similar to a z-score

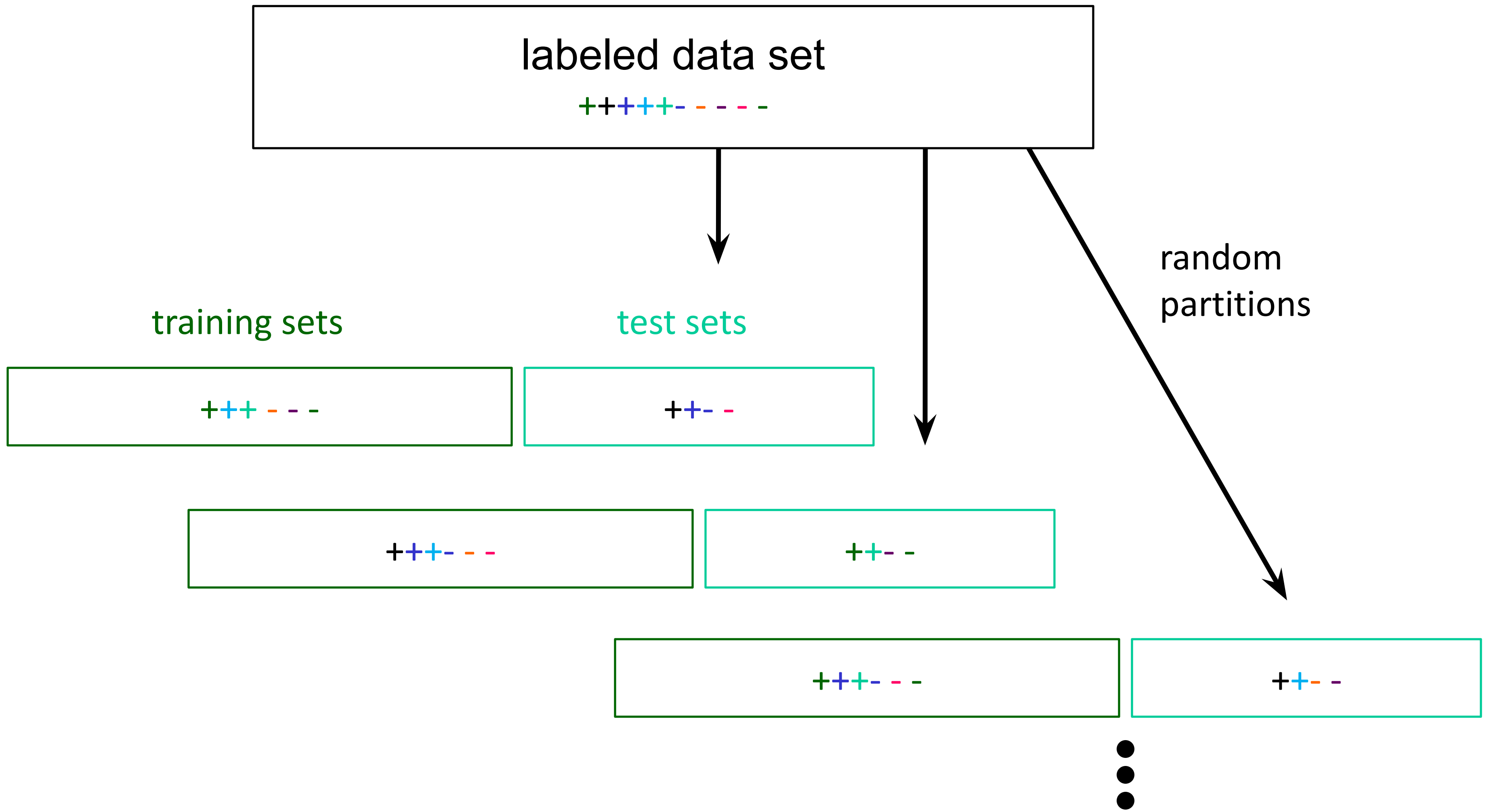
Single Train/Test Split: Limitations

1. 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 the learning process

2. A single training set cannot reveal how sensitive accuracy is to specific training samples.

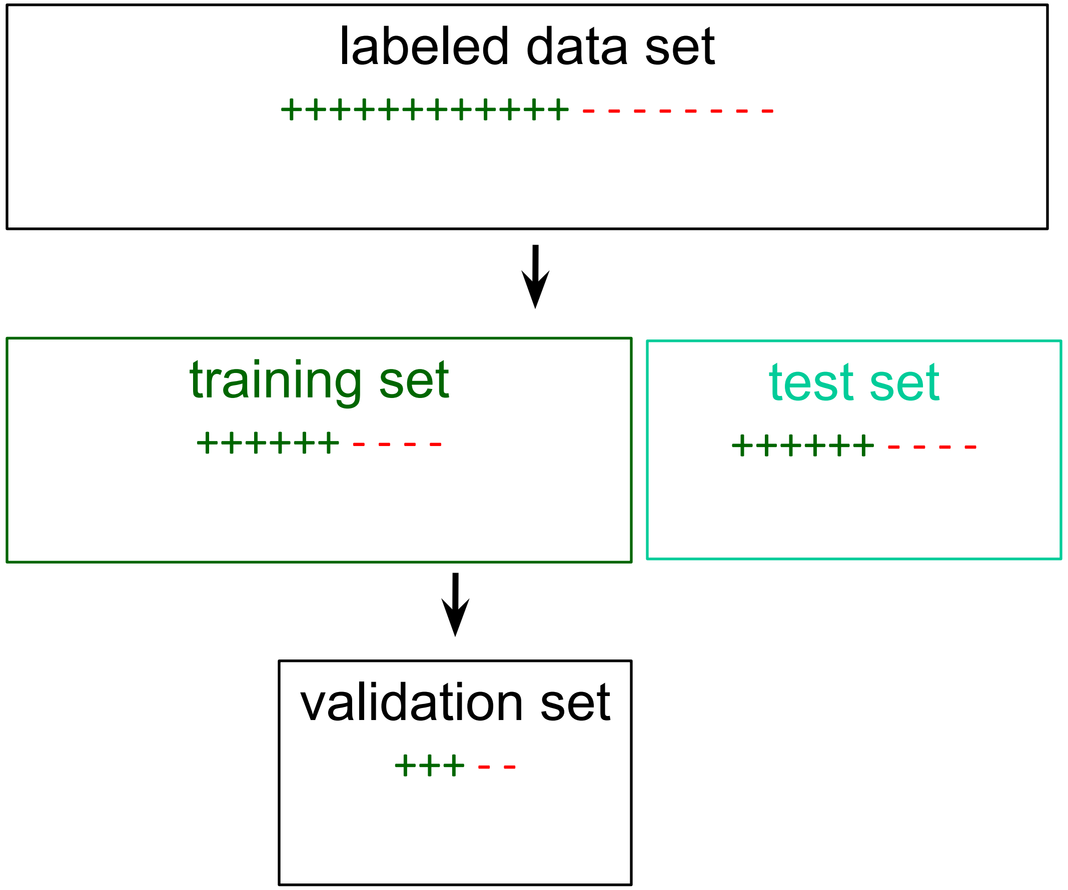
Strategy I: Random Resampling

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



Strategy I: Stratified Sampling

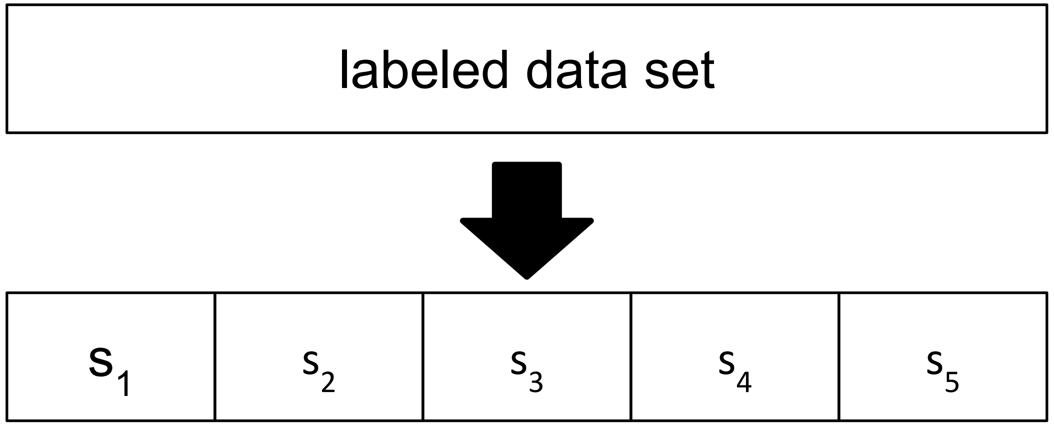
- When randomly selecting training or validation sets, we may want to ensure that **class proportions** are maintained in each selected set



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

Strategy II: *k*-fold Cross Validation

Partition data
into *k* 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: 5-fold Cross Validation Example

- Suppose we have 100 instances, and we want to estimate accuracy with 5-fold 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 folds are often used when learning takes a lot of time
- in *leave-one-out* cross validation, *k* = # 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 (with specific choices)</u> as opposed to an <u>individual learned hypothesis</u>
- You can use CV for tuning as well!

Break & Quiz

Q2-1: Are these statements true or not?
(A) The accuracy of a model is the training set accuracy, and its estimator is the test set accuracy.
(B) An unbiased estimator $\hat{\theta}$ always equals its correspond true parameter θ.

1. True, True
2. True, False
3. False, True
4. False, False

Q2-1: Are these statements true or not?

(A) The accuracy of a model is the training set accuracy, and its estimator is the test set accuracy.

(B) An unbiased estimator $\hat{\theta}$ always equals its correspond true parameter θ.

1. True, True
2. True, False
3. False, True
4. [✓ False, False]

(A) The accuracy of a model is based on its true distribution; training/test sets only approximate this.

(B) It only equals the true parameter in expectation, i.e. it's true in the limit of a large number of estimates. This means there's no systematic error.

Q2-2: Are these statements true or not?
(A) The sample size on the learning curve is the size of *test* set.
(B) A larger *training* set would provide a lower variance estimate of the accuracy of a learned model.

1. True, True
2. True, False
3. False, True
4. False, False

Q2-2: Are these statements true or not?
(A) The sample size on the learning curve is the size of *test* set.
(B) A larger *training* set would provide a lower variance estimate of the accuracy of a learned model.

1. True, True
2. True, False
3. False, True
4. [✓ False, False]

(A) The sample size on the learning curve is for training set.
(B) A larger test set rather than a larger training set does so.

Q2-3: Which of the following is NOT true?

1. Class proportions are maintained same in the stratified sampling.
2. In leave-one-out cross validation, the number of partition equals to the number of instances.
3. In cross validation, we are evaluating the performance of an individual learned hypothesis.

Q2-3: Which of the following is NOT true?

1. Class proportions are maintained same in the stratified sampling.
2. In leave-one-out cross validation, the number of partition equals to the number of instances.
3. In cross validation, we are evaluating the performance of an individual learned hypothesis.

In cross validation, we are evaluating a learning method as opposed to a specific individual learned hypothesis.

Outline

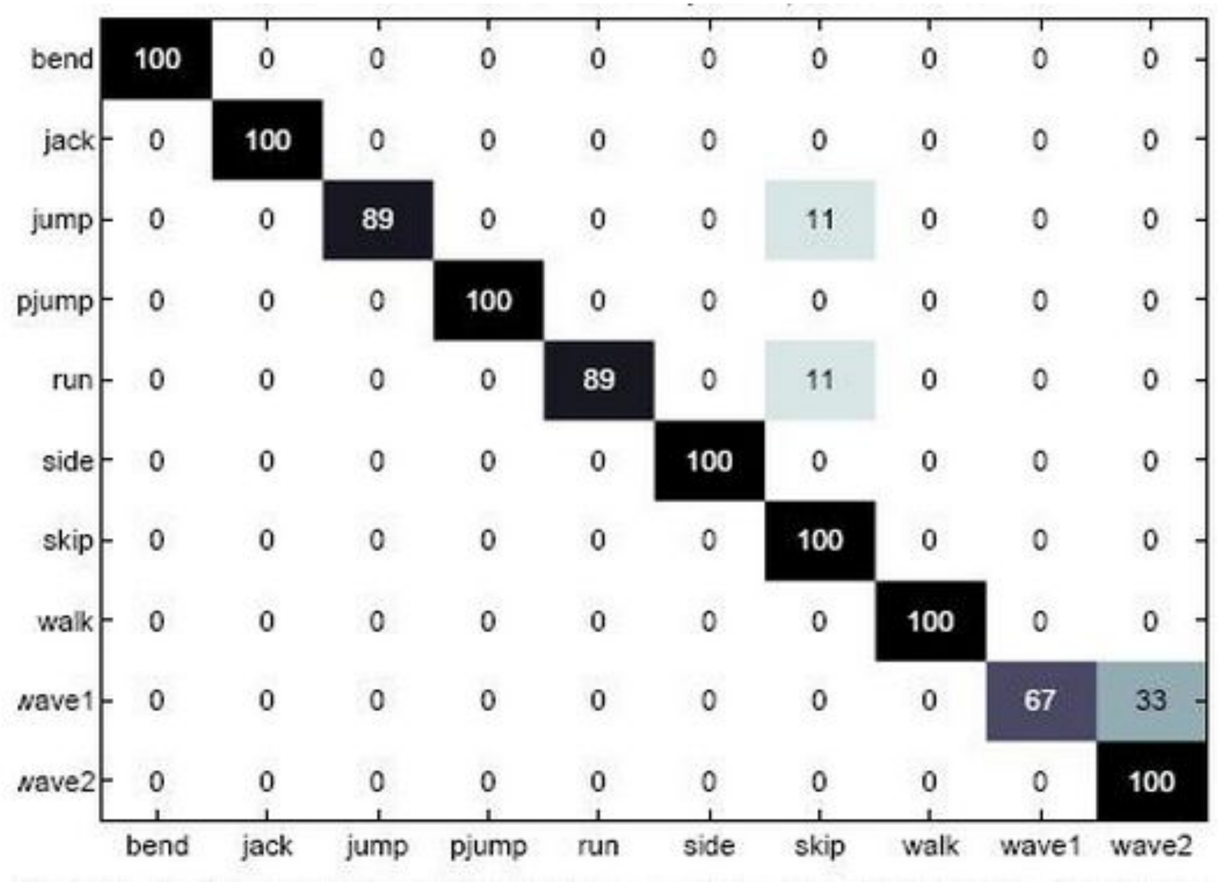
- **Wrapping up decision trees**
  - Review, Variations, information gain, regression
  - Evaluation in decision trees: overfitting, pruning, variations
- **Evaluation: Generalization**
  - Train/test split, random sampling, cross validation
- **Evaluation: Metrics**
  - Confusion matrices, ROC curves, 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	negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

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

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

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

Other Metrics

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

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

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

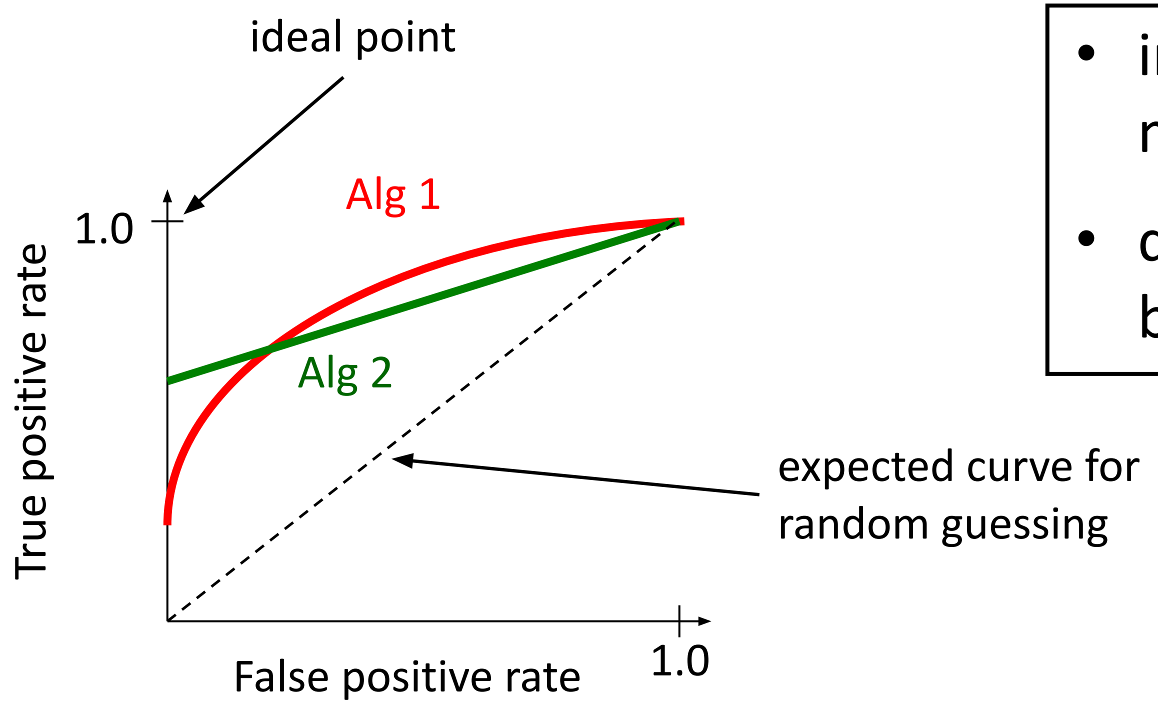
Suppose a classifier returns label probabilities rather than a label. How do you decide the label?

i.e. suppose a classifier returns Pr(Y|x) instead of Y.

- not optimal to simply take the maximum probability if cost of FP and FN differ
- one solution: choose threshold c and output Y=True if Pr(Y=True|x) $\geq$ c else Y=False
- but how do you compare methods across thresholds?

Other Metrics: ROC Curves

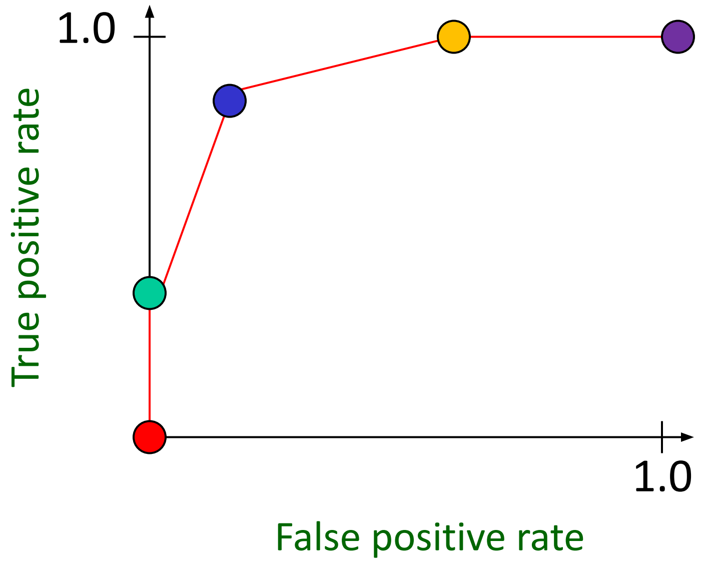
- A *Receiver Operating Characteristic* (*ROC*) curve plots the TP-rate vs. the FP-rate as a threshold on the confidence of an instance being positive is varied



- increasing the threshold c moves down along the curve
- different methods can work better at different points

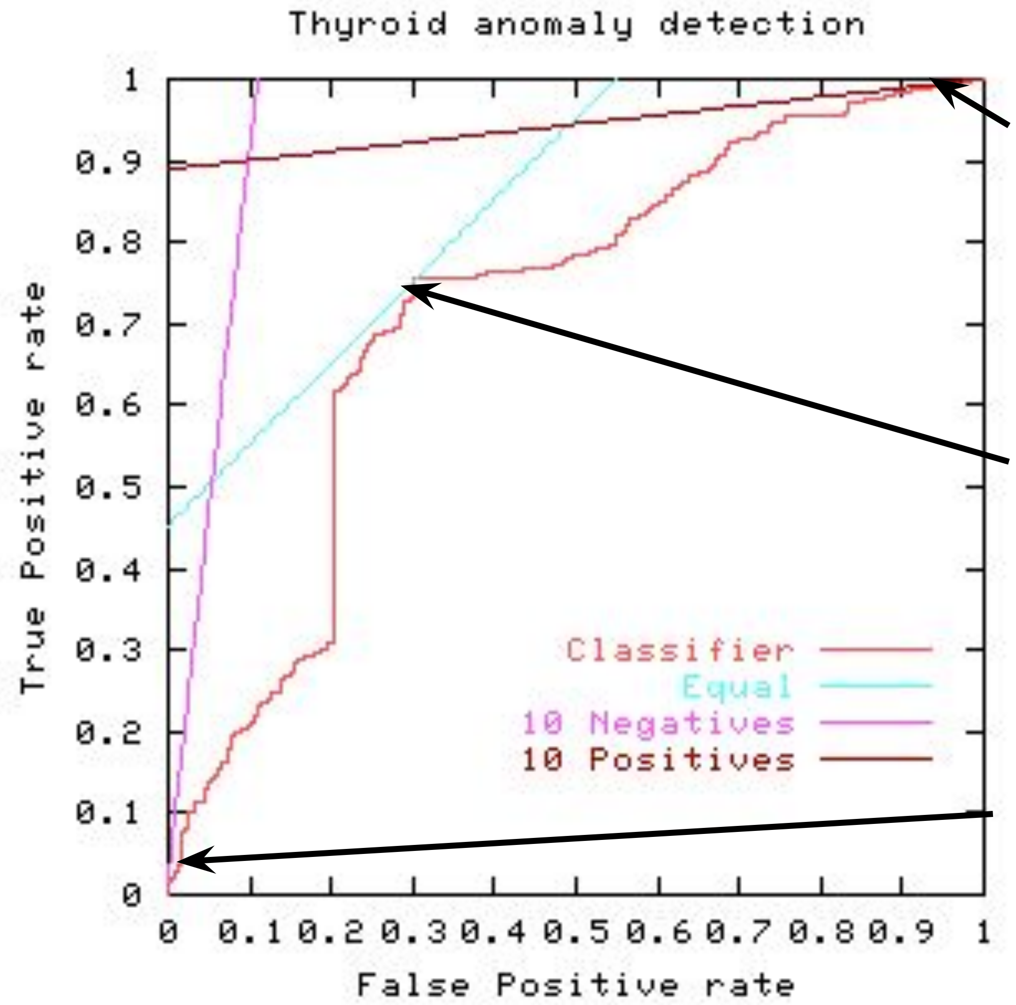
ROC Curves: Plotting

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 and Recall

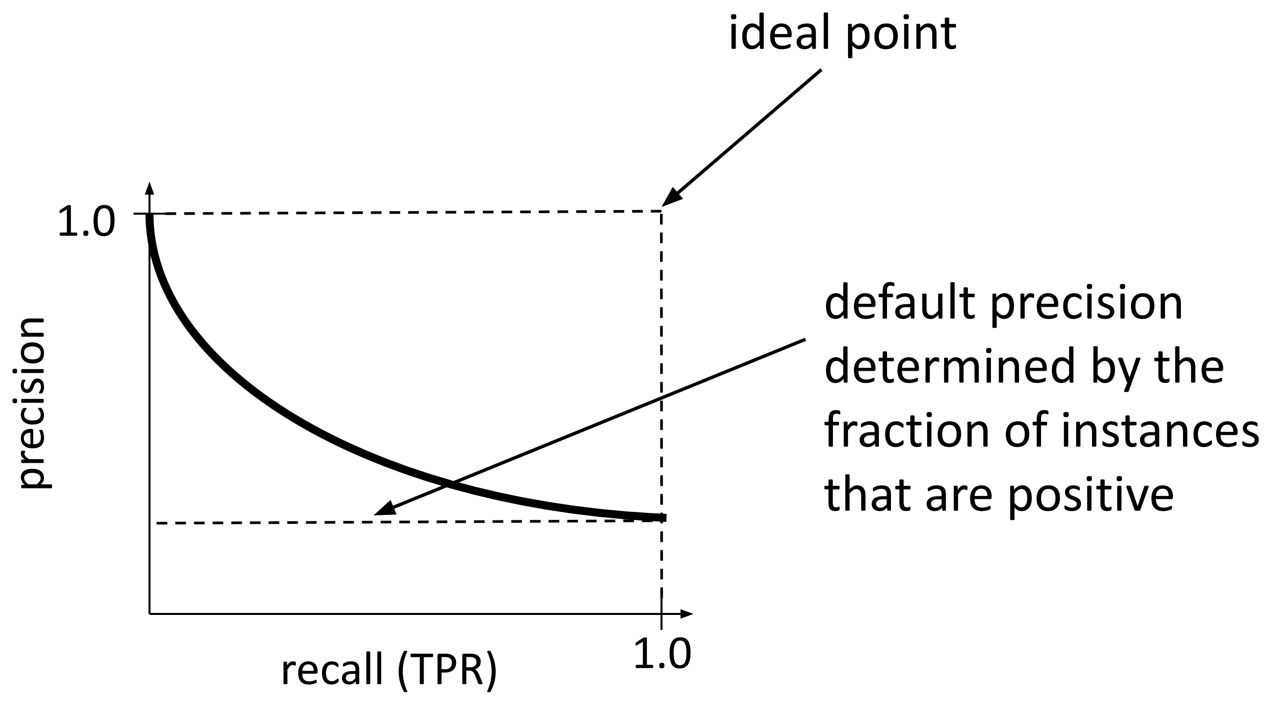
		actual class	
		positive	negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

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

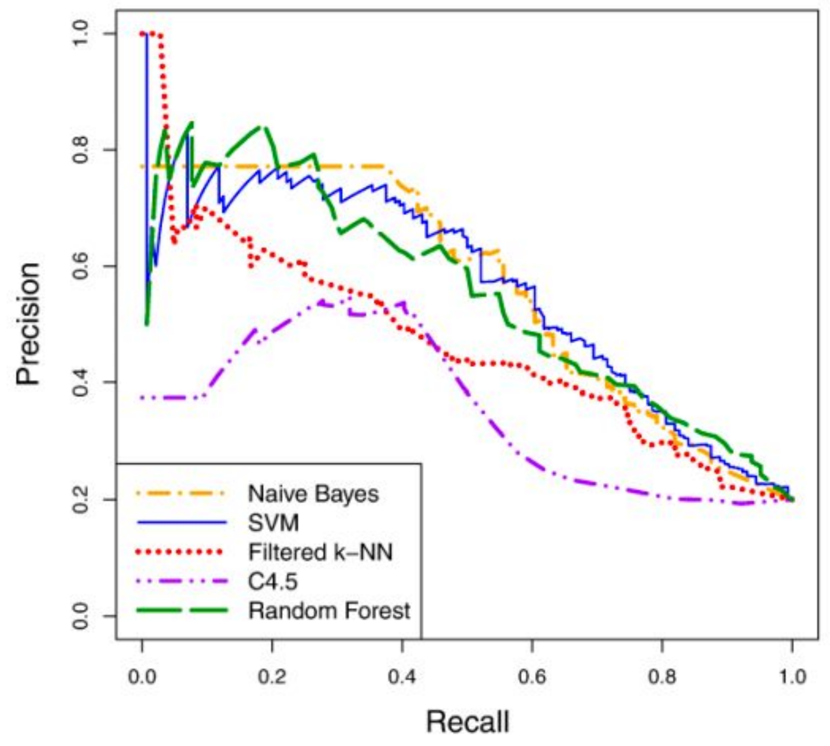
$$\text{precision (positive predictive value)} = \frac{\text{TP}}{\text{predicted pos}} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Other Metrics: Precision/Recall Curve

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


ideal point
1.0
precision
default precision determined by the fraction of instances that are positive
recall (TPR)
1.0

predicting patient risk for VTE



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

Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Misha Khodak, Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, Pedro Domingos, Jerry Zhu, Yingyu Liang, Volodymyr Kuleshov, Fred Sala, Kirthi Kandasamy, Josiah Hanna, Tengyang Xie