



COMPUTER SCIENCE

INDIANA UNIVERSITY

School of Informatics and Computing  
Bloomington

# Evaluation basics



# Reminders/Comments

- Thought questions due today
- Class mini project updates:
  - sample project reports from last time; they are slightly different in that they were requested to be high on the leaderboard
  - I do not require high leaderboard status (though this is a good thing to do anyway), but do require scientific question in the conclusion
  - You can use packages, such as scikit, for the project



# Crash course in evaluation

- Running a scientific experiment to compare machine learning algorithms necessary to draw conclusions
- Needs to be meticulous, even if sometimes tedious or need to re-run experiments
- Requires an experiment design and statistical significance tests
- See these slides: <http://pages.cs.wisc.edu/~dpage/cs760/evaluating.pdf>





# Hypotheses to test

- Algorithm A is better than Algorithm B
  - this is almost impossible to say
- Algorithm A is better than Algorithm B on this dataset
  - for all possible hyper-parameter settings?!
- With specific settings of hyperparameters, Algorithm A is better than Algorithm B on this dataset
  - but now is Algorithm B better with different hyperparameters?
  - want to ask a stronger question
- Specifying a testable hypothesis is part of the difficulty
- What do you mean by “better”? Why this dataset?



# Selecting the definition of better

- Best classification accuracy
- Best classification accuracy on “most important” samples
- Robust classification accuracy that ignores outliers
- ROC curve and AUC
- Training time and space complexity
- Testing time and space complexity
- Ease of implementing the algorithm and interpretability
- We’ll talk about measures later; for now assume you’ve defined “better”



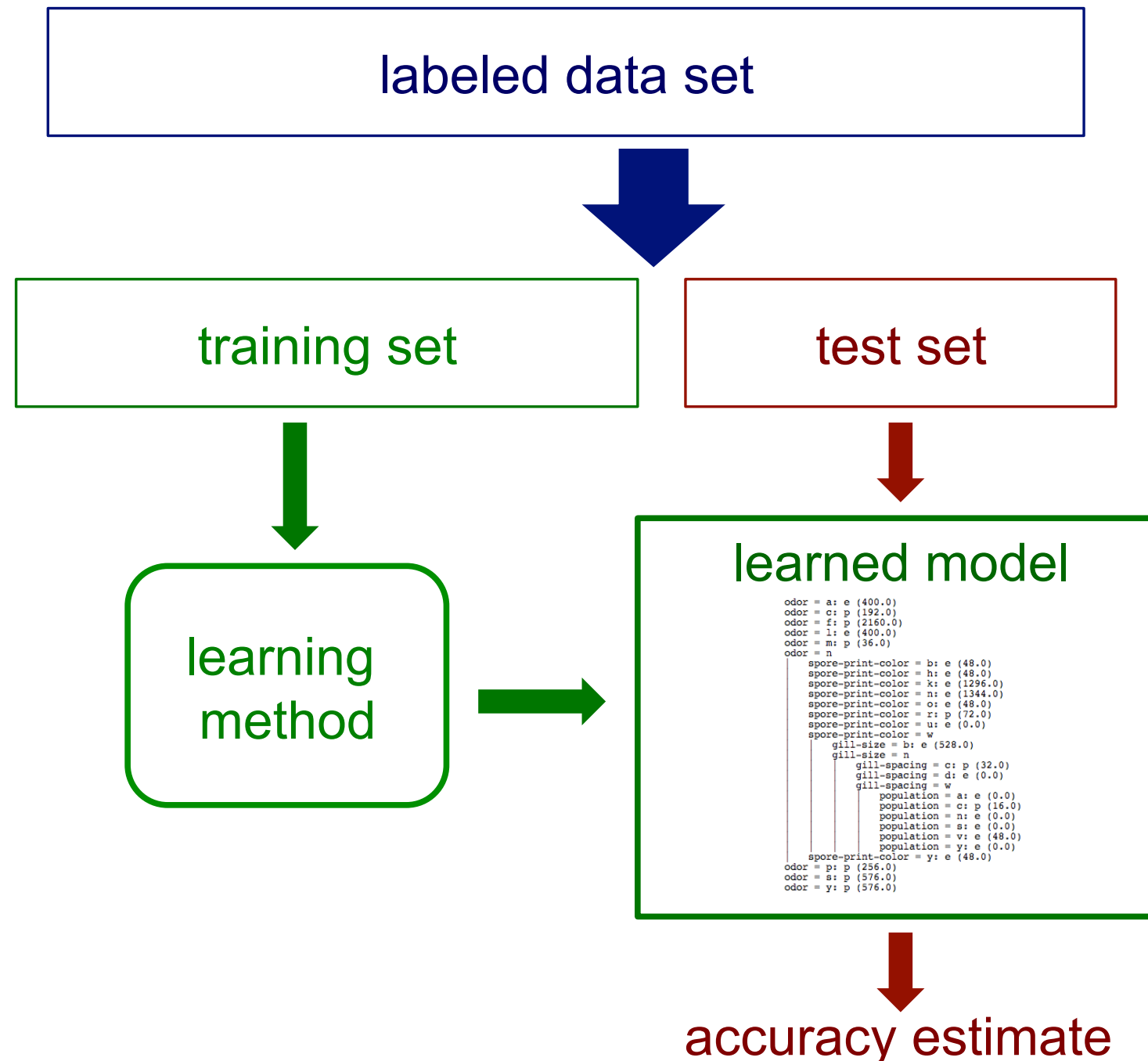
# Select dataset(s)

- Either you have some domain that you care about, and associated data
  - your goal is to predict well on future data, and generally better understand the data for your domain with whatever algorithm
- Or you care about exploring the properties of the algorithm and might find a diverse set of datasets
  - this includes potentially generating synthetic data for which you understand the properties
  - e.g. generate iid data from a Gaussian distribution
  - e.g., generate data from a simple linear dynamical system
- Again for now assume you've selected the data



# Evaluating on the data

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





# Test sets revisited

- 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")
  - in some applications, reasonable to assume that you have access to the feature vector ( $x$ ) but not the targets  $y$  (transductive learning)
- If the test-set labels influence the learned model in any way, accuracy estimates will be biased

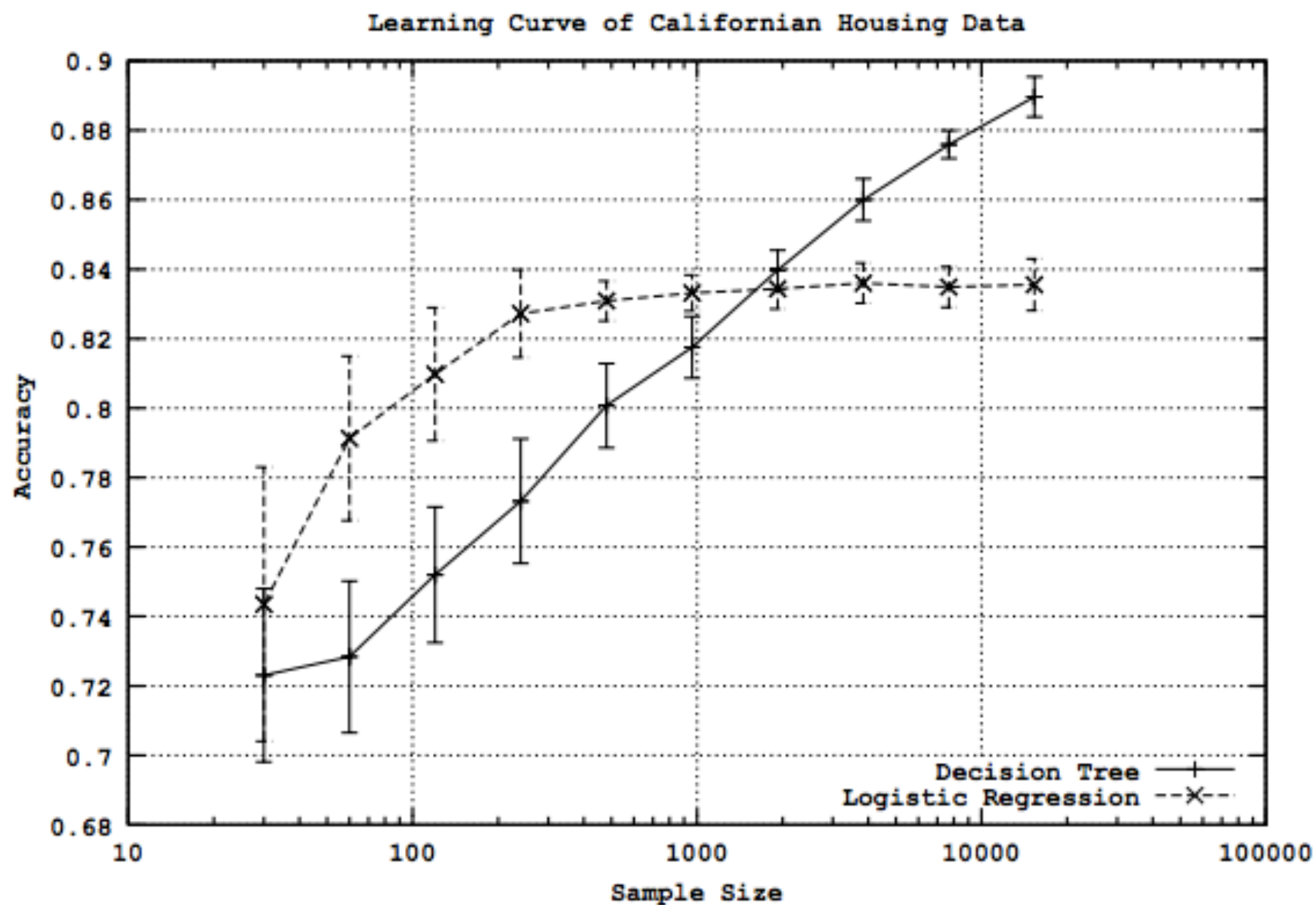




# Learning curves

- How does the accuracy of a learning method change as a function of the training-set size?

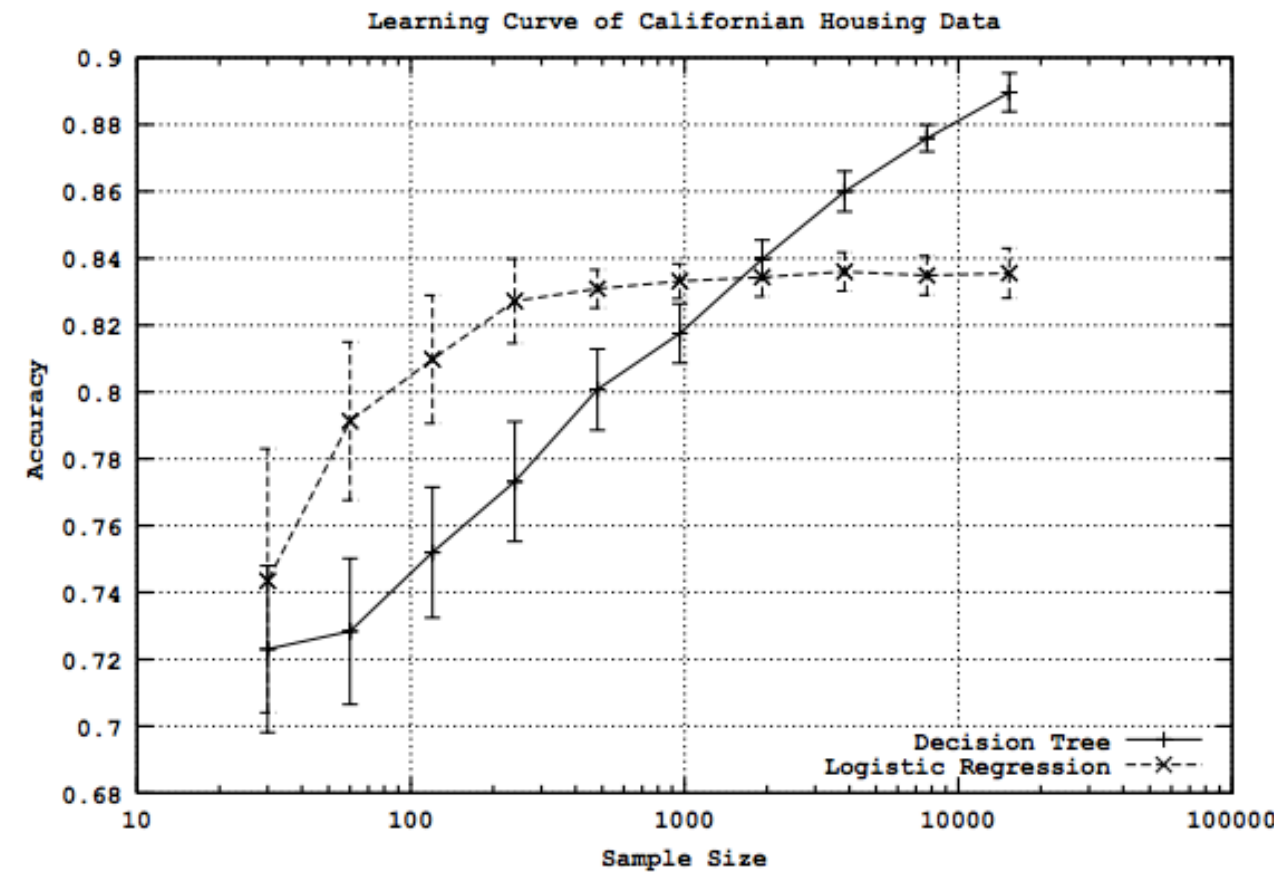
this can be assessed by plotting *learning curves*





# Learning curves

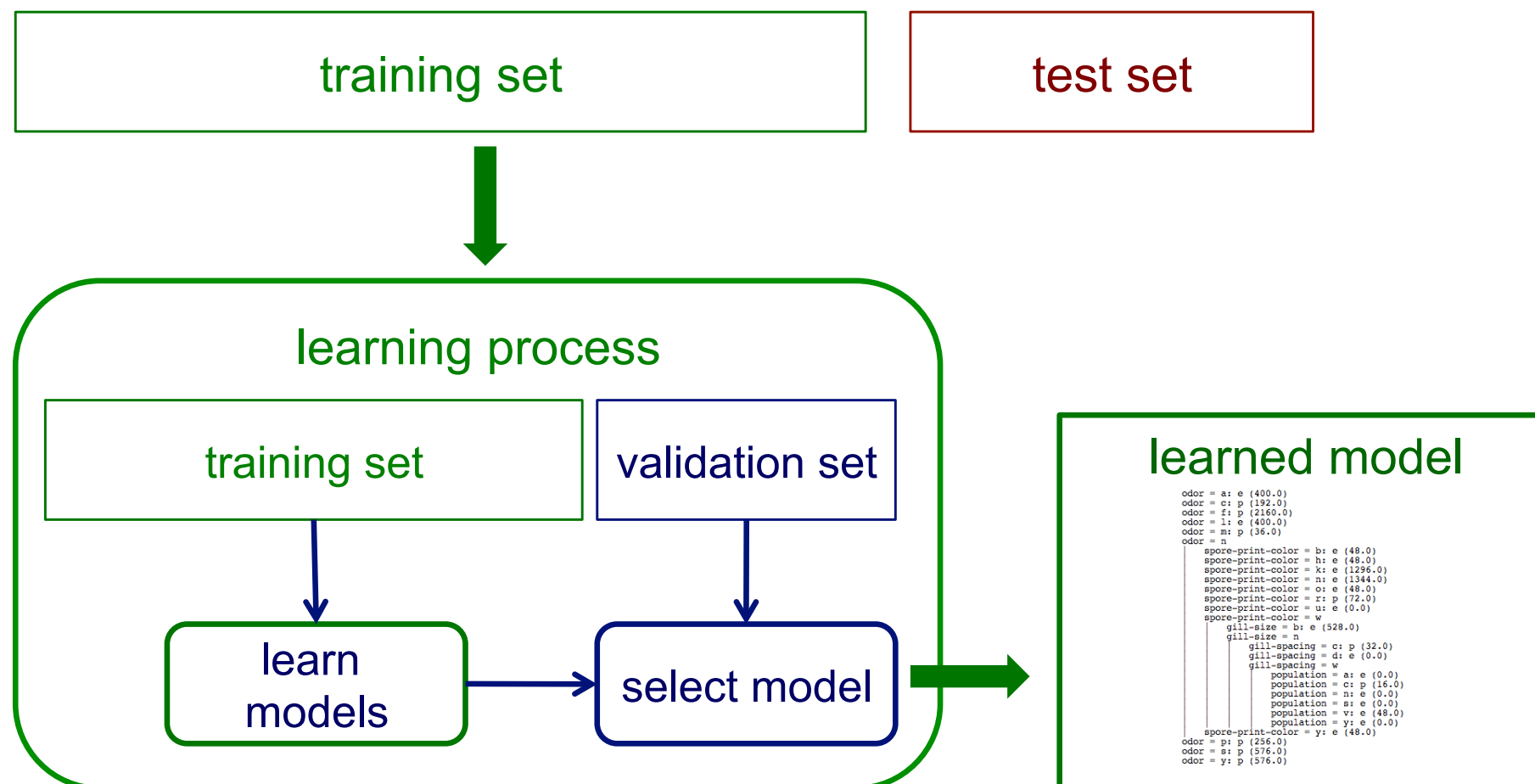
- Given training/test set partition
  - for each sample size  $s$  on learning curve
    - repeat  $n$  times
      - randomly select  $s$  instances from training set
      - learn model
      - evaluate model on test set to determine the accuracy  $a$
      - plot( $s, a$ ) or ( $s$ , avg. accuracy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





# Validating (tuning) sets

- Suppose we want unbiased estimates of accuracy during the learning process (e.g. to choose the best regularization parameter for linear regression)?



Partition training data into separate training/validation sets



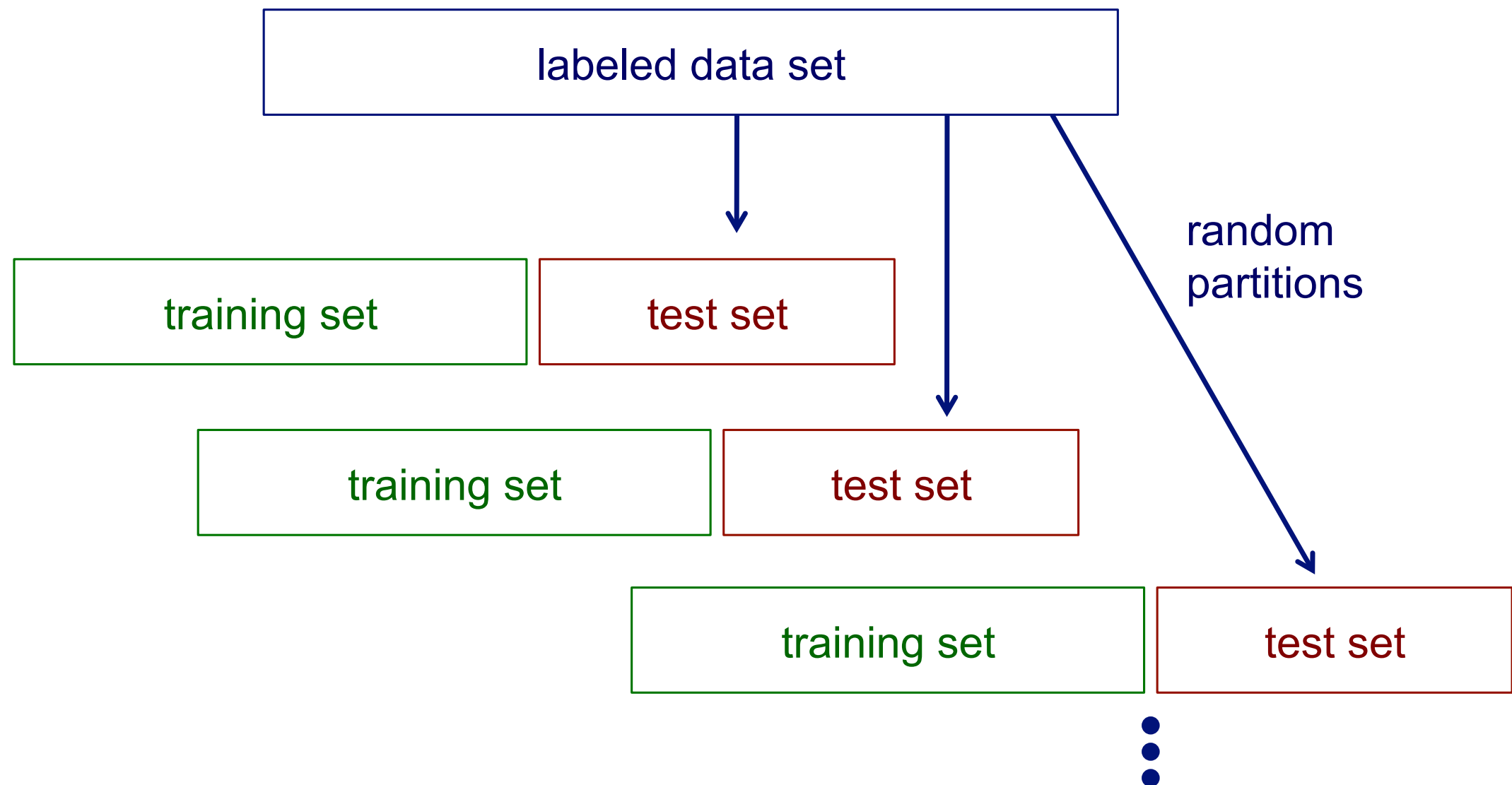
# Limitations of using a single training/test partition

- We may not have enough data to make sufficiently large training and 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 doesn't tell us how sensitive accuracy is to a particular training sample



# Random resampling

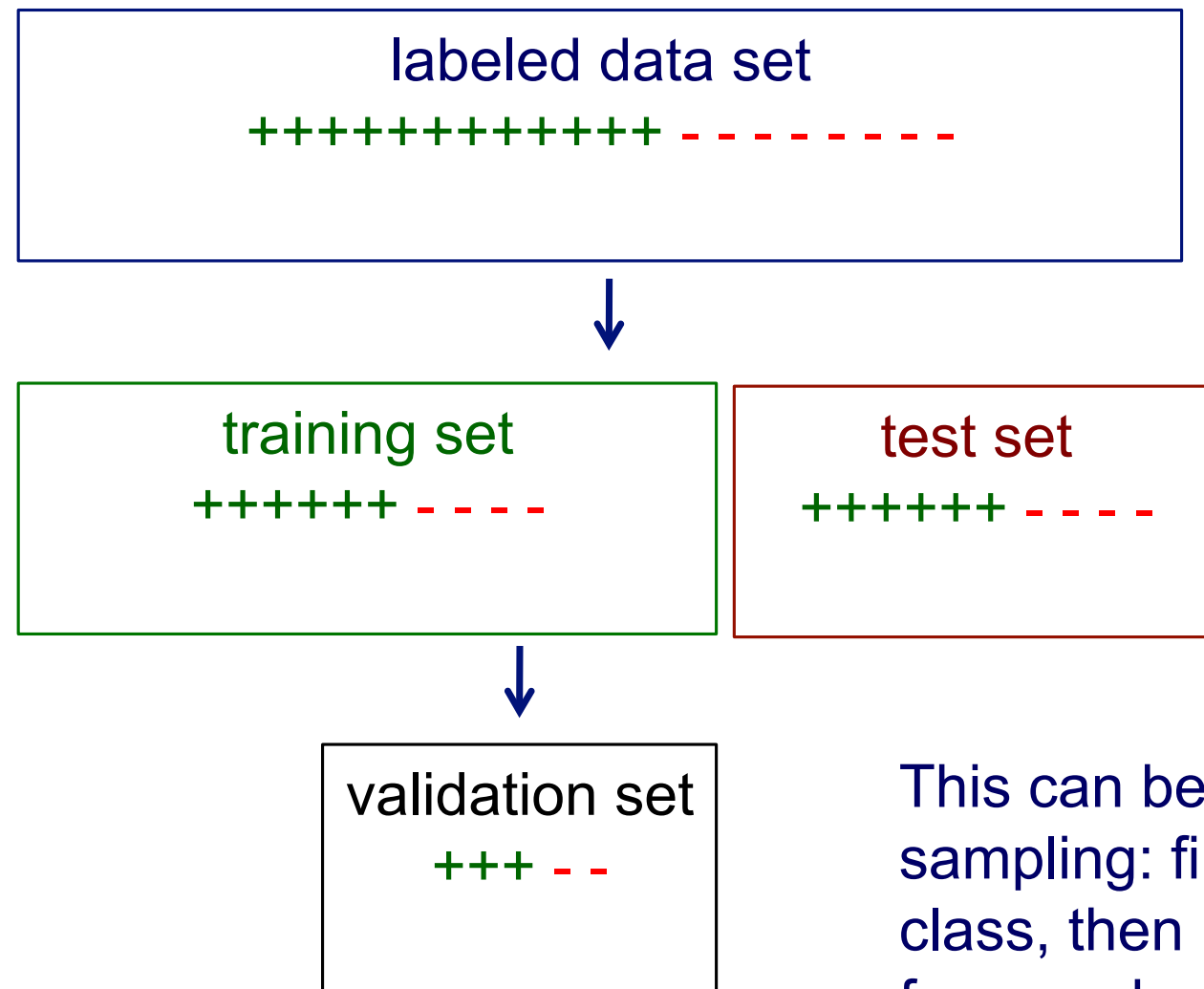
- We can address the second issue by repeatedly randomly partitioning the available data into training and set sets.





# Stratified sampling

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

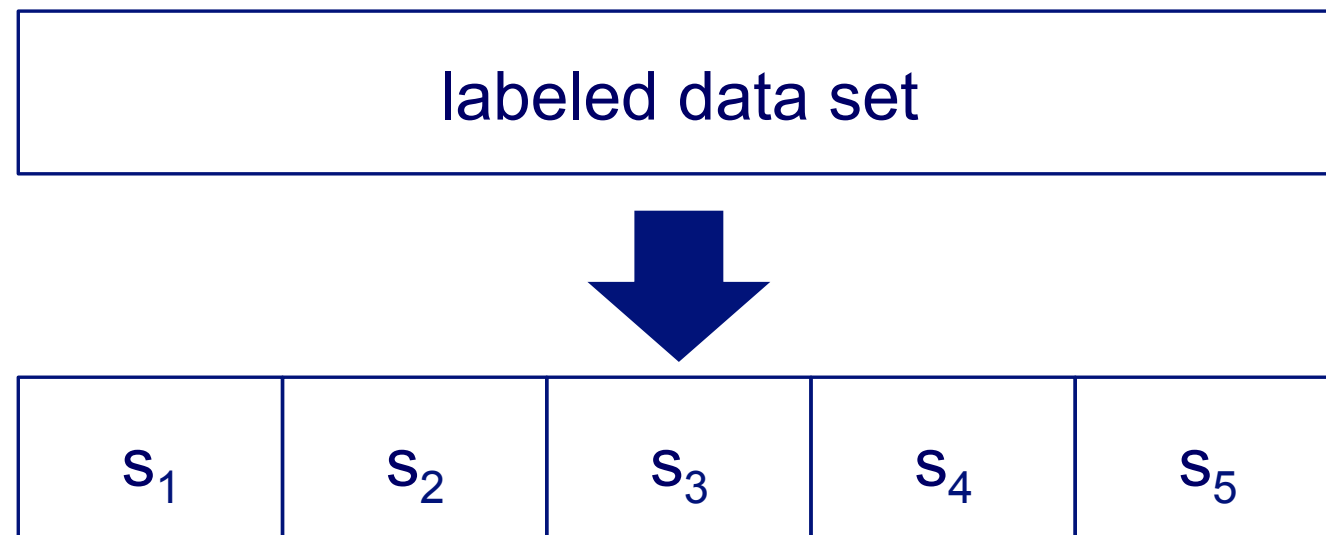


This can be done via stratified sampling: first stratify instances by class, then randomly select instances from each class proportionally.



# Cross validation

partition data  
into  $n$  subsamples

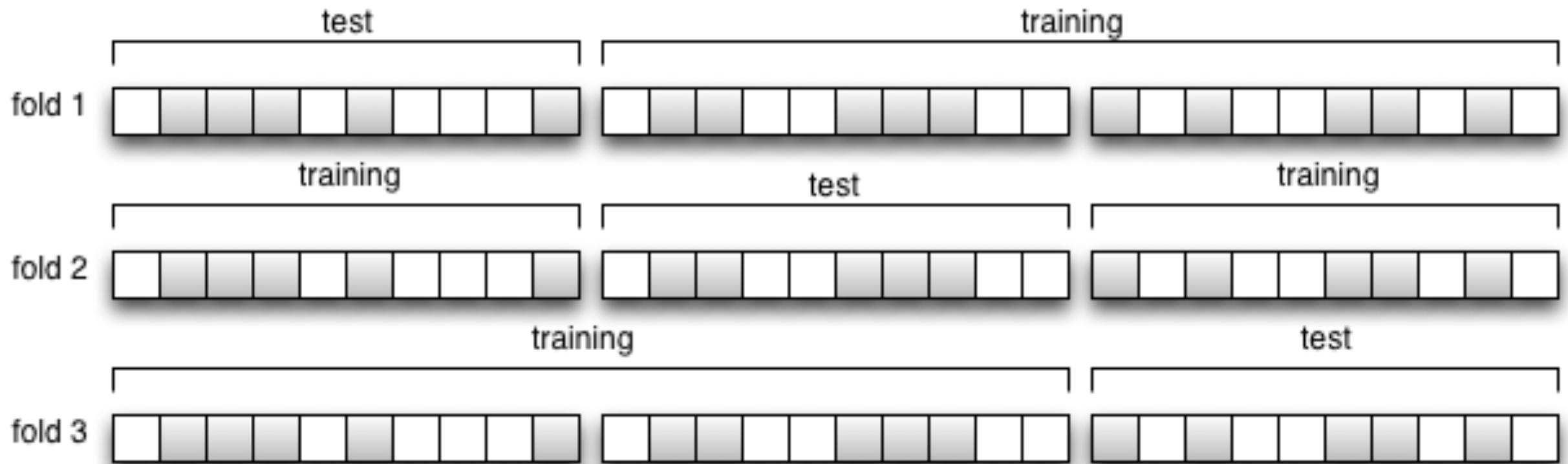


iteratively leave one  
subsample out for  
the test set, train on  
the rest

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	$S_1$ $S_2$ $S_3$ $S_4$	$S_5$



# Another view of cross validation







# 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 =  $73/100 = 73\%$



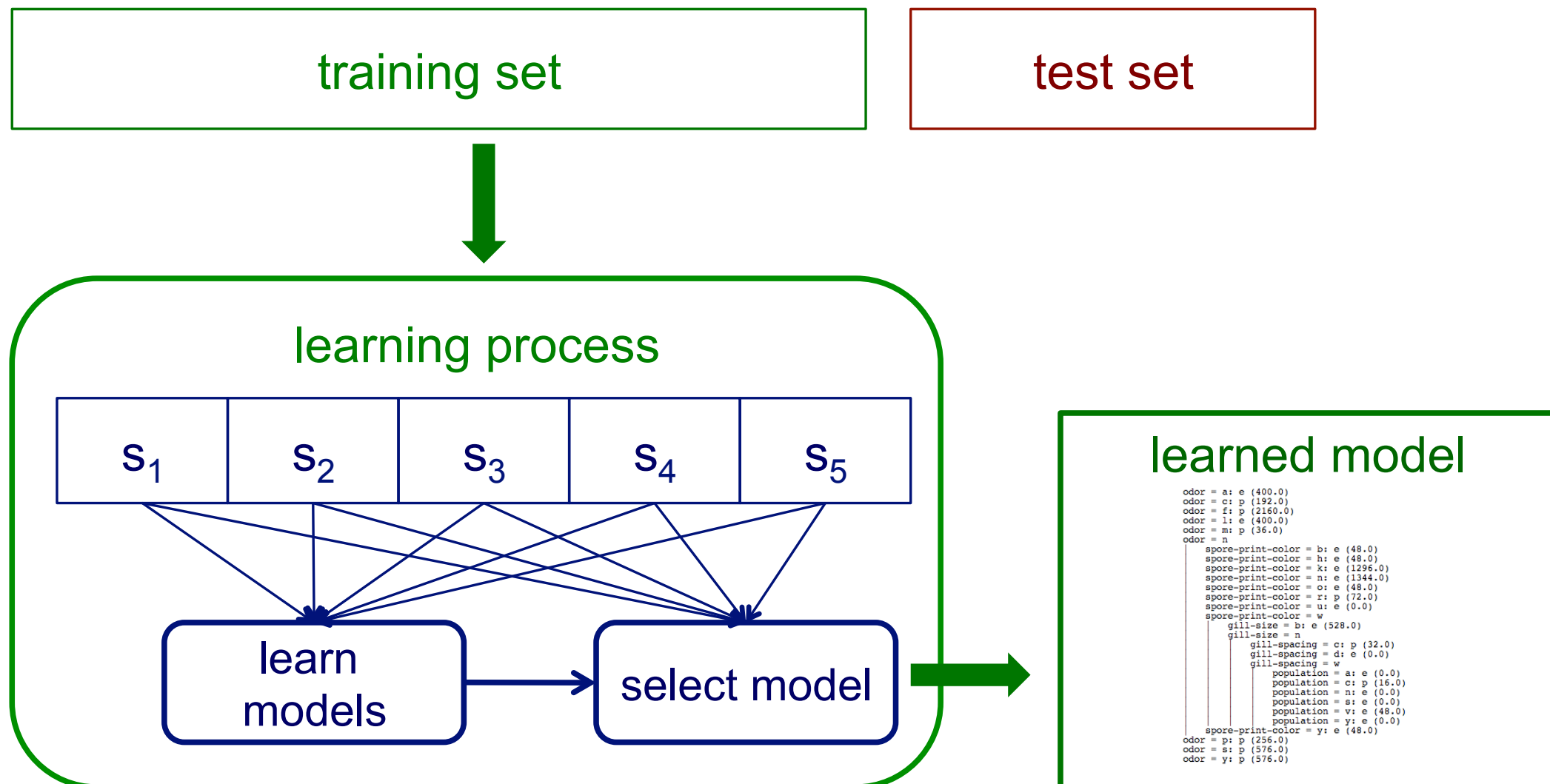
# Cross validation

- 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 learning method as opposed to an individual learned model



# Internal cross validation

- Instead of a single validation set, we can use cross validation within a training set to select a model (e.g. to choose the best regularization parameter for linear regression)





# Example: selecting the regularization parameter with CV

- Given a training set
  1. partition the training set into  $n$  folds,  $s_1, \dots, s_n$
  2. for each value of  $\lambda$  considered
    - for  $i = 1$  to  $n$ 
      - learn regression model using all folds but  $s_i$
      - evaluate accuracy on  $s_i$
  3. select  $\lambda$  that resulted in the best accuracy for  $s_1, \dots, s_n$
  4. learn model using entire training set and selected  $\lambda$
- This is run separately for each training set
  - would not likely use this to pick hyperparameters across training sets



# Overall experiment design

- Hyperparameters to try for each algorithm
  - e.g., have to decide on the range of  $\lambda$  to try
  - e.g., which optimizer to use in algorithm
- Depending on choices, answering a different question
  - e.g., one training/test split tells you how a learned model performs, on that test set
  - e.g., multiple training/test splits tells you how a learning method, with given hyperparameters, performs on that test set
  - e.g., could report algorithm with parameter settings as a single learning method, understand parameter sensitivity
- Running a thorough experiment can be difficult, but rewarding



# Testing for significance

- Imagine now that you have 100 error values from your experiment (obtained from 100 random training/test splits)
- Imagine you choose hyperparameters with CV each time on the training set, and so are evaluating Algorithm A and B
  - rather than just a specific learned model
  - assuming you tested a reasonably large range of hyperparameters
- The average error value for Algorithm A is smaller than Algorithm B: can you conclude that A is better?
- Need statistical significance tests, the mean not enough info



# Statistical significance tests

- Null hypothesis: A and B have the same generalization performance (i.e., A and B have the same expected error)
  - by running on multiple random test sets, obtaining unbiased estimates of expected error
- Alternative hypothesis: A and B have different generalization performance
- Confidence intervals and standard error
- Paired t-test



# Computing confidence intervals

- Can obtain a confidence interval around the expected error of an algorithm
  - commonly make a normal assumption (because of central limit theorem)

$$0.95 = P(\bar{X} - 1.96 \frac{\sigma}{\sqrt{N}} \leq \mu \leq \bar{X} + 1.96 \frac{\sigma}{\sqrt{N}})$$

- If two confidence intervals do not overlap, can say that the two algorithms have statistically significantly different expected errors (and so that one has statistically significantly lower expected error than the other)
- If the two confidence intervals do overlap, need to use a statistical significance test
  - see <http://www.cs.iastate.edu/~honavar/dietterich98approximate.pdf> for a comparison of the performance of several statistical tests

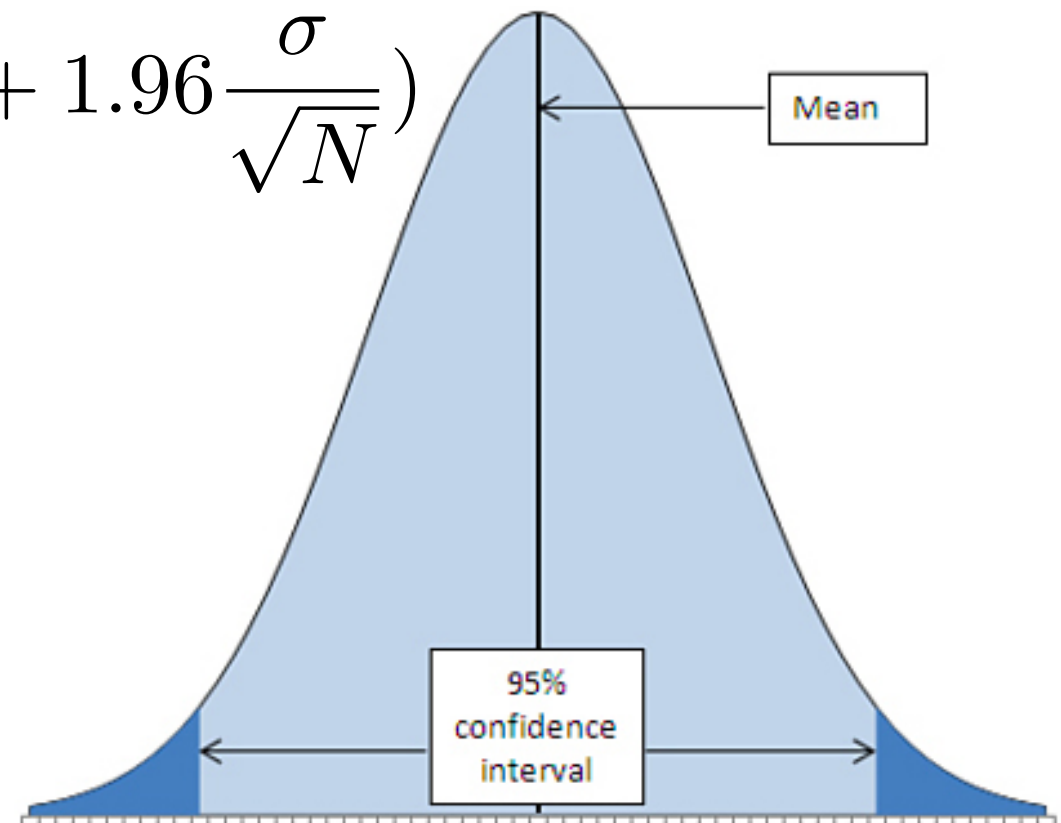




# Normal confidence interval

- What if we plot our 100 errors and they do not look normally distributed? We'll talk about this later
- The normal 95% confidence interval determines the endpoints that contain 95% of the mass between them, under the curve

$$0.95 = P\left(\bar{X} - 1.96 \frac{\sigma}{\sqrt{N}} \leq \mu \leq \bar{X} + 1.96 \frac{\sigma}{\sqrt{N}}\right)$$





# Paired t-test

- Mean accuracy for System 1 is better, but the standard deviations for the two clearly overlap
- Notice that System 1 is always better than System 2

	<u>Accuracies on test sets</u>				
System 1:	80%	50	75	...	99
System 2:	79	49	74	...	98
$\delta$ :	+1	+1	+1	...	+1



# Comparing systems using a paired t-test

1. calculate the sample mean

$$\bar{\delta} = \frac{1}{n} \sum_{i=1}^n \delta_i$$

2. calculate the  $t$  statistic

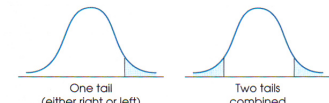
$$t = \frac{\bar{\delta}}{\sqrt{\frac{1}{n(n-1)} \sum_{i=1}^n (\delta_i - \bar{\delta})^2}}$$

3. determine the corresponding  $p$ -value, by looking up  $t$  in a table of values for the Student's  $t$ -distribution with  $n-1$  degrees of freedom

APPENDIX B STATISTICAL TABLES 691

TABLE B.2 THE  $t$  DISTRIBUTION

Table entries are values of  $t$  corresponding to proportions in one tail or in two tails combined.



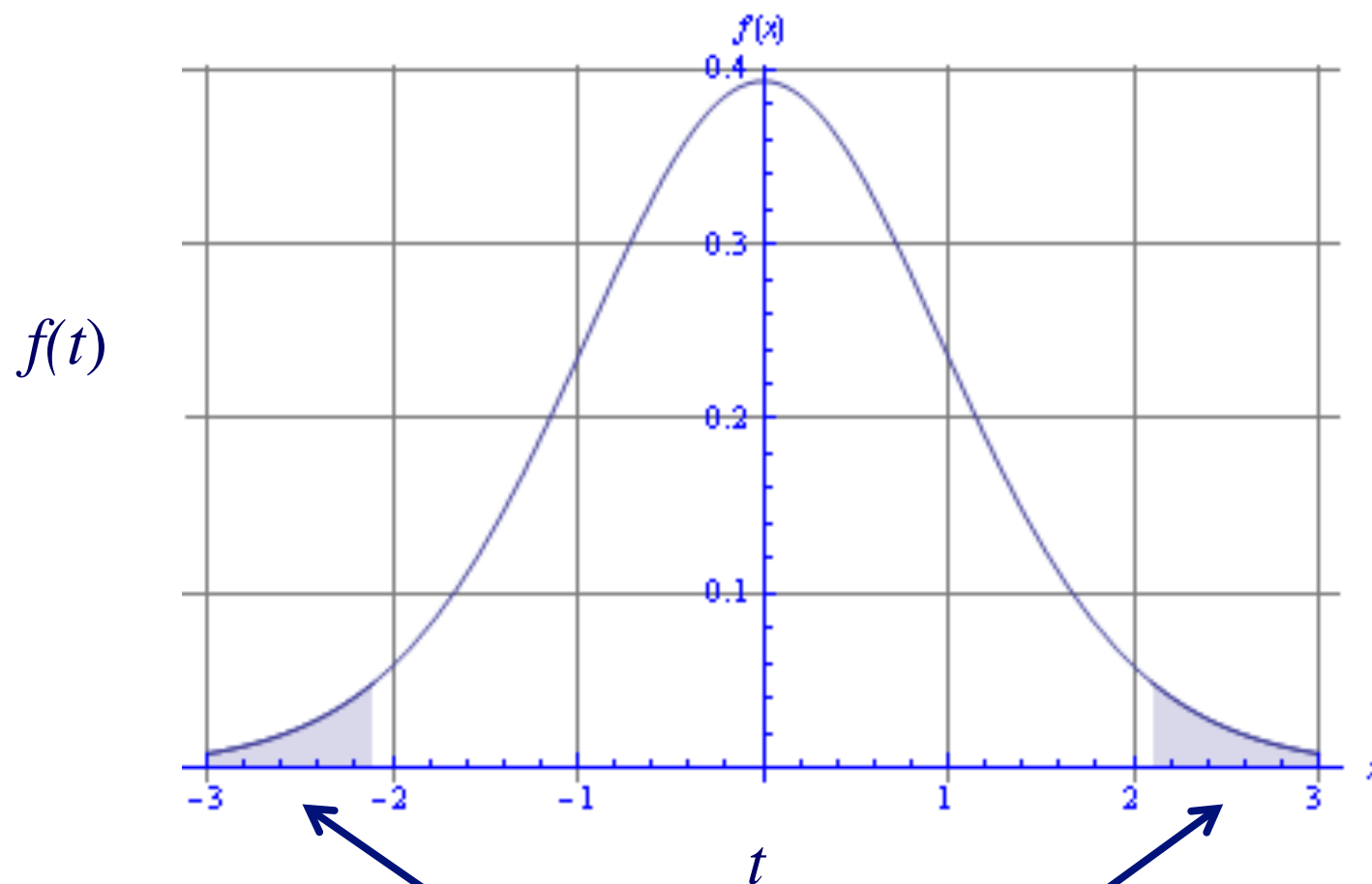
One tail (either right or left)      Two tails combined

df	PROPORTION IN ONE TAIL					PROPORTION IN TWO TAILS COMBINED
	0.25	0.10	0.05	0.025	0.01	
1	1.000	3.078	6.314	12.706	31.821	65.057
2	0.816	1.886	2.920	4.303	6.965	9.925
3	0.765	1.638	2.353	3.182	4.541	5.841
4	0.741	1.533	2.132	2.776	3.747	4.604
5	0.727	1.476	2.015	2.571	3.365	4.032
6	0.718	1.440	1.943	2.447	3.143	3.707
7	0.711	1.415	1.895	2.365	2.998	3.499
8	0.706	1.397	1.860	2.306	2.896	3.355
9	0.703	1.385	1.833	2.262	2.821	3.250
10	0.700	1.372	1.812	2.228	2.764	3.169
11	0.697	1.363	1.796	2.201	2.718	3.106
12	0.695	1.356	1.782	2.179	2.681	3.055
13	0.694	1.350	1.771	2.160	2.650	3.012
14	0.692	1.345	1.761	2.145	2.624	2.977
15	0.691	1.341	1.753	2.131	2.602	2.947
16	0.690	1.337	1.746	2.120	2.583	2.921
17	0.689	1.333	1.740	2.110	2.567	2.898
18	0.688	1.330	1.734	2.101	2.552	2.878
19	0.688	1.328	1.729	2.093	2.539	2.861
20	0.687	1.325	1.725	2.086	2.528	2.845
21	0.686	1.323	1.721	2.080	2.518	2.831
22	0.686	1.321	1.717	2.074	2.508	2.819
23	0.685	1.319	1.714	2.069	2.500	2.807
24	0.685	1.318	1.711	2.064	2.492	2.797
25	0.684	1.316	1.708	2.060	2.485	2.787
26	0.684	1.315	1.706	2.056	2.479	2.779
27	0.684	1.314	1.703	2.052	2.473	2.771
28	0.683	1.313	1.701	2.048	2.467	2.763
29	0.683	1.311	1.699	2.045	2.462	2.756
30	0.683	1.310	1.697	2.042	2.457	2.750
40	0.681	1.303	1.684	2.021	2.423	2.704
60	0.679	1.296	1.671	2.000	2.390	2.660
120	0.677	1.289	1.658	1.980	2.358	2.617
$\infty$	0.674	1.282	1.645	1.960	2.326	2.576

File B11 of R. A. Fisher and F. Yates, *Statistical Tables for Biological, Agricultural and Medical Research*, 6th ed. London: Longman Group Ltd., 1963 (previously published by Oliver and Boyd Ltd., Edinburgh). Adapted and reprinted with permission of the Addison Wesley Longman Publishing Co.



# Comparing systems using a paired t-test



The null distribution of our  $t$  statistic looks like this

The  $p$ -value indicates how far out in a tail our  $t$  statistic is

If the  $p$ -value is sufficiently small, we reject the null hypothesis, since it is unlikely we'd get such a  $t$  by chance

for a two-tailed test, the  $p$ -value represents the probability mass in these two regions

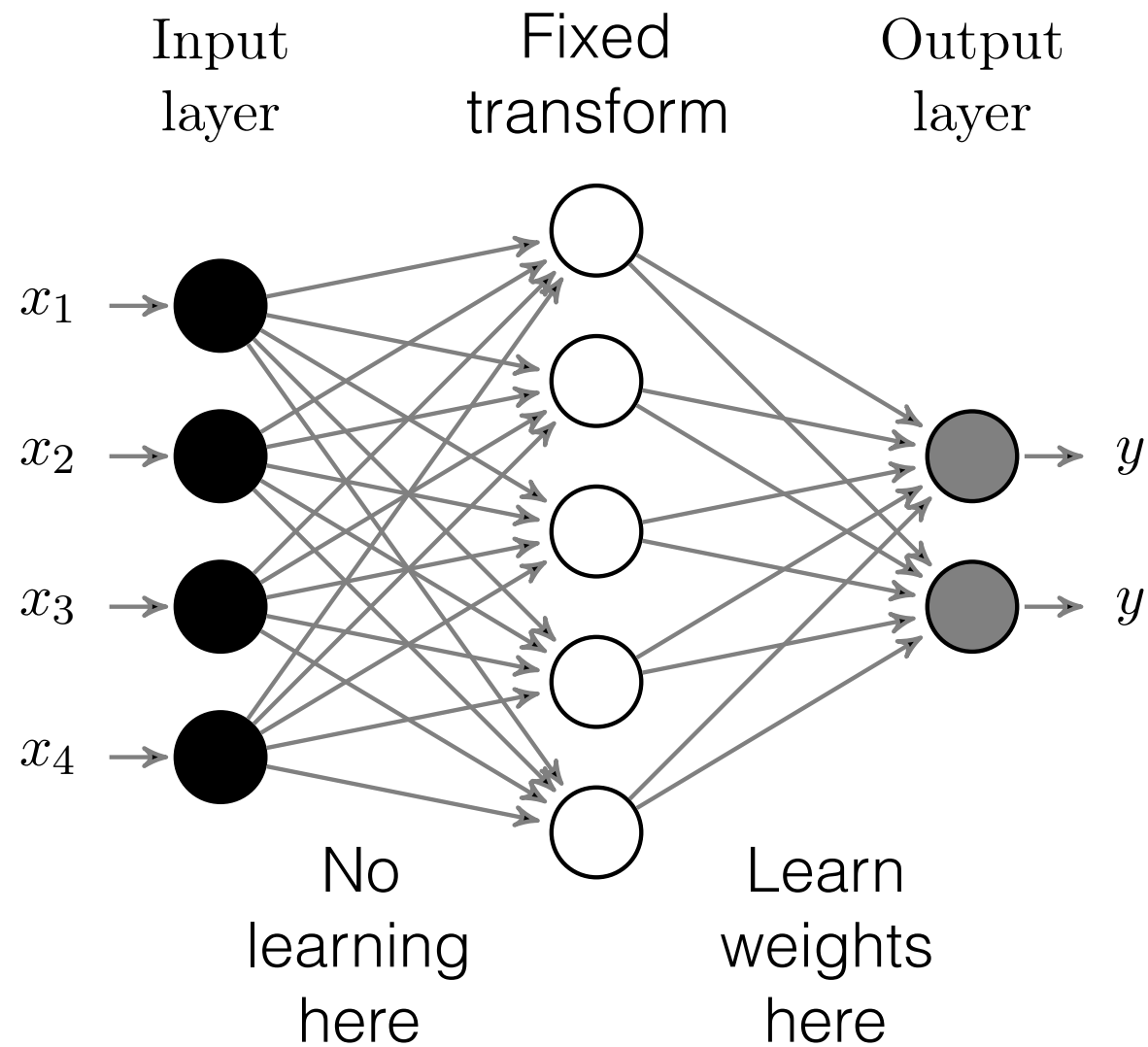


# Evaluation summary

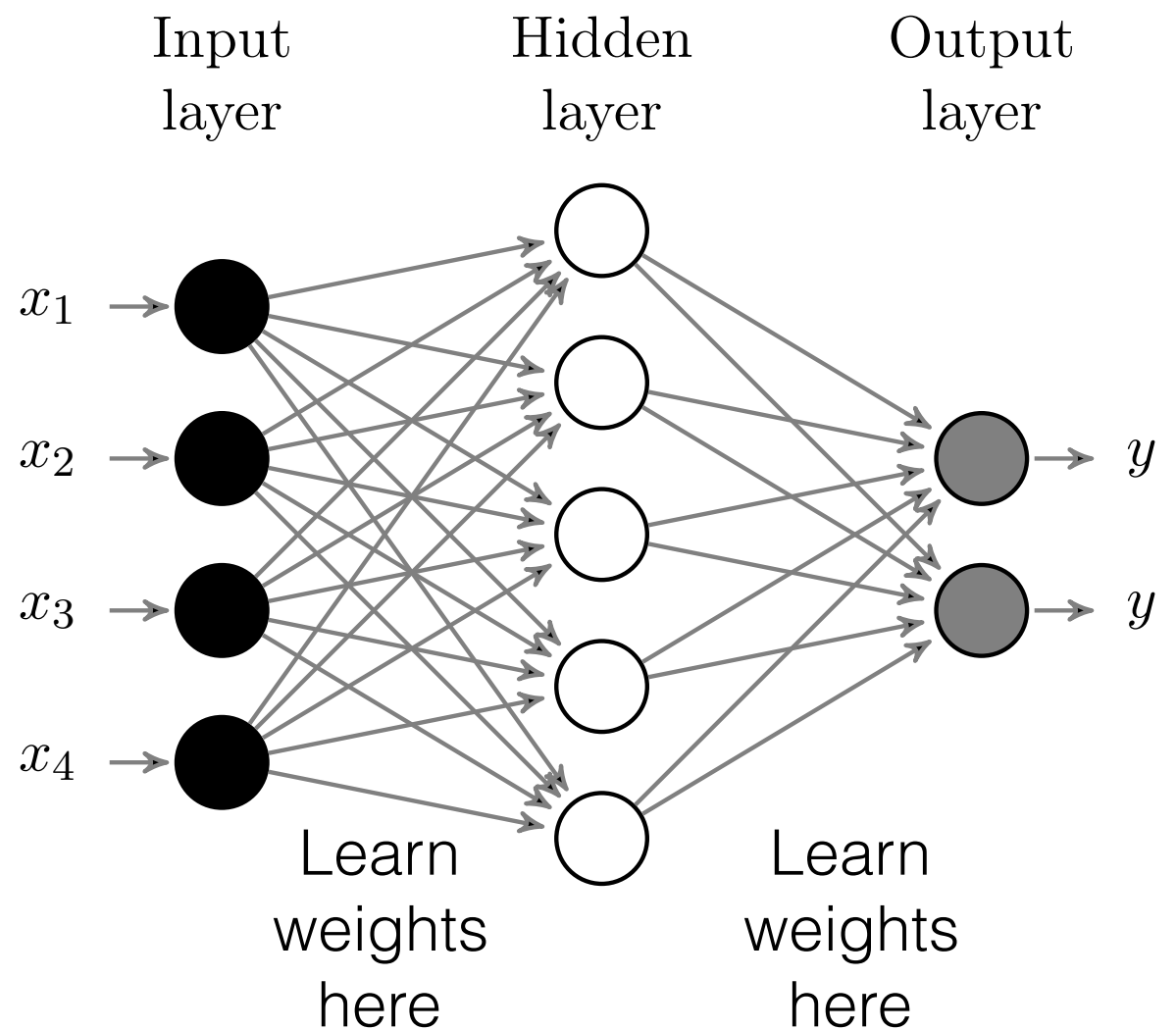
- Define “better” and your hypothesis upfront, to direct experiment to answer that hypothesis
- Be cognizant of your choices
  - e.g., hyperparameters, optimizers
  - e.g., number of folds
- Do not cheat by looking at test data
  - then you’ll just do poorly on some new data, outside the test data
- To avoid overfitting hyperparameter selection on the given training data, systematically sweep parameters rather than using human guessing and testing
  - this could bias you to training set, and cause bad performance on test



# GLM with fixed representation vs. neural network



GLM with  
augmented fix representation



Two-layer neural network



# Whiteboard

- Examples of backpropagation with other transfers