Evaluation basics

Reminders/Comments

- Thought questions due today
- Assignment 2
    - Typo in stochastic gradient descent pseudocode
    - Let’s use linear regression with average error
- Class mini-project updates:
    - Can work in pairs (if you want)
    - You can use packages, such as scikit, for the project
    - Your goal is to formalize your problem; you can ask me about it somewhat, but this is a part of the project

Linear regression

$$\arg\min_{\mathbf{w}} \frac{1}{n}\|\mathbf{X}\mathbf{w}-\mathbf{y}\|_2^2 = \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w}-\mathbf{y}\|_2^2$$

$$\arg\min_{\mathbf{w}} \frac{1}{n}\|\mathbf{X}\mathbf{w}-\mathbf{y}\|_2^2 + \lambda\|\mathbf{w}\|_2^2 \neq \arg\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w}-\mathbf{y}\|_2^2 + \lambda\|\mathbf{w}\|_2^2$$

$$\arg\min_{\mathbf{w}} \frac{1}{n}\|\mathbf{X}\mathbf{w}-\mathbf{y}\|_2^2 + \lambda\|\mathbf{w}\|_2^2$$

$$\mathbf{w} = \left(\frac{1}{n}\mathbf{X}^\top\mathbf{X} + \lambda\right)^{-1}\left(\frac{1}{n}\mathbf{X}^\top\mathbf{y}\right)$$

Stepping back: What is machine learning?

- Central goal: obtain models that give good generalization performance (i.e., predict Y accurately from X)
  - Sometimes just want a good model for one problem
  - Typically want to identify approaches that work well across problems
  - Central to this theme: bias-variance trade-off
- Procedure:
  - Formalize the problem (e.g., as a maximum likelihood problem)
  - Propose a solution methodology (e.g., good optimization algorithm)
  - Evaluate performance (either theoretically or empirically)

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

MOVIE SCIENCE MONTAGE
ACTUAL SCIENCE MONTAGE
Squeak!
...WHIRRRRR...
...WHIRRR...
BING!
PAINT FLECKS FROM THE KILLER'S CLOTHING MATCH AN ANTIMATTER FACTORY IN BELGRADE!
LET'S GO!
OKAY, WE'VE DETERMINED THERE'S NEITHER BARIUM NOR RADIUM IN THIS SAMPLE.
PROBABLY.

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

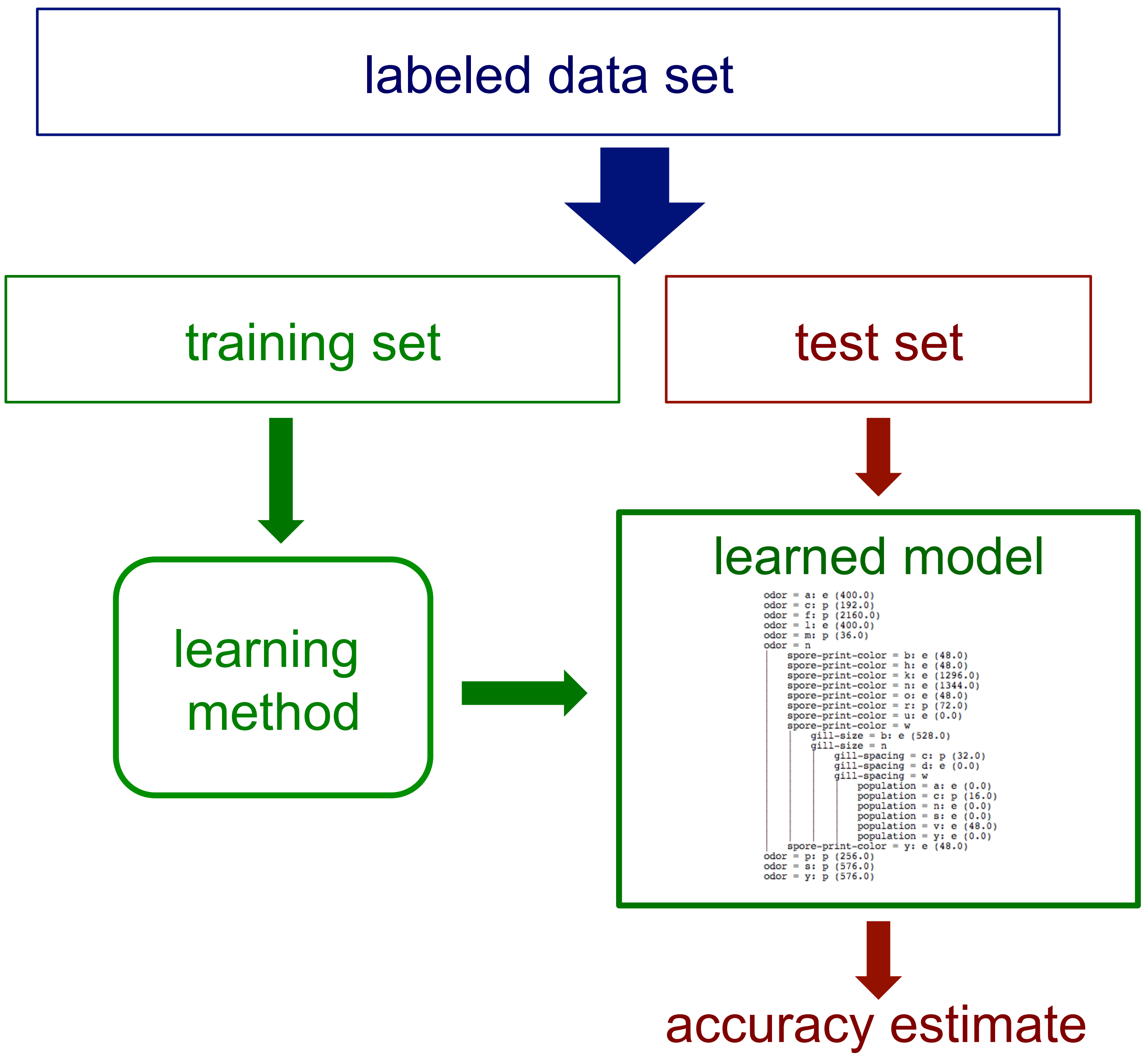
- Best classification accuracy
- Best classification accuracy on “most important” samples
- Robust classification accuracy that ignores outliers
- ROC curve and AUC
- Training time and space complexity
- Testing time and space complexity
- Ease of implementing the algorithm and interpretability
- We’ll 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 neural network
- 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?



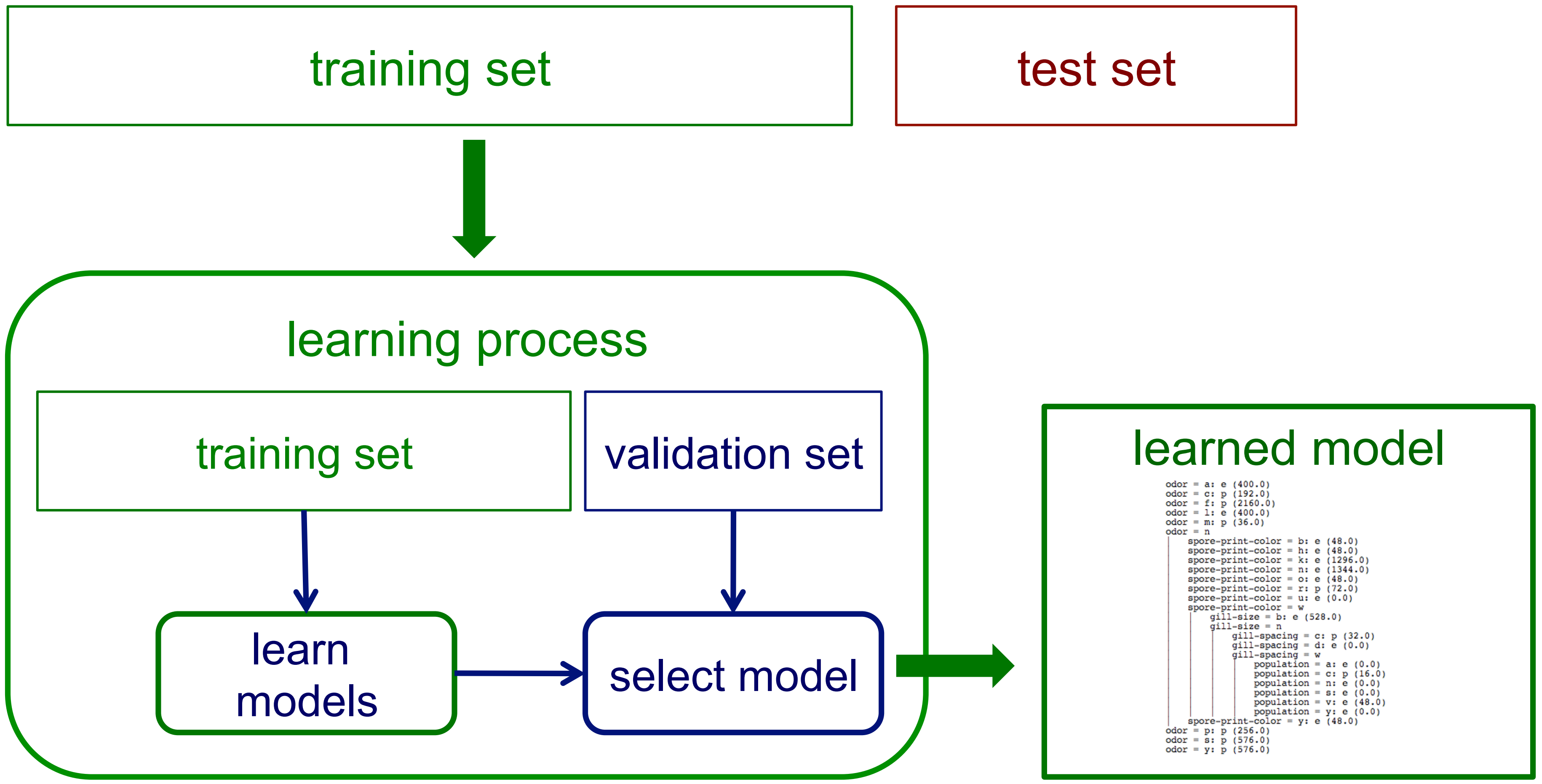
```
odor = a: e (400.0)
odor = c: p (192.0)
odor = f: p (2160.0)
odor = l: e (400.0)
odor = m: p (36.0)
odor = n
|   spore-print-color = b: e (48.0)
|   spore-print-color = h: e (48.0)
|   spore-print-color = k: e (1296.0)
|   spore-print-color = n: e (1344.0)
|   spore-print-color = o: e (48.0)
|   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)
```

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”)
- If the test-set labels influence the learned model in any way, accuracy estimates will be biased

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)?



```
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)
```

Partition training data into separate training/validation sets

Exercise: l1 regression models

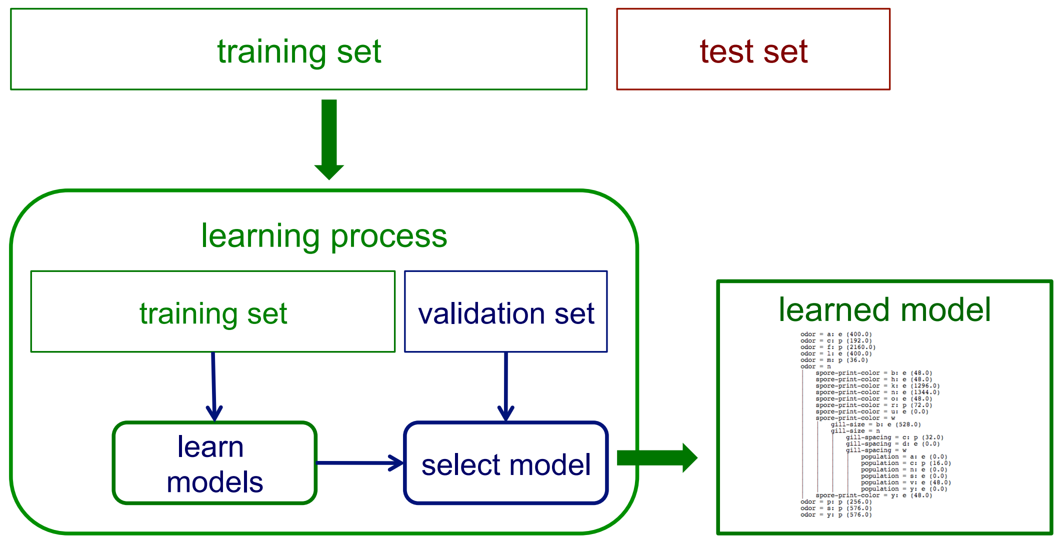
- Imagine you have a dataset with 5 observations (inputs)

- You expand up your inputs into a 9-th order polynomial
  - For d = 5 (number of inputs), k = 9 (the order), the total number of terms is (d+k) choose k. For d = 5, k = 9, this is 2000

- Now you are going to run l1 regression, to subselect features
  - why?

$$\min_{\mathbf{w}\in\mathbb{R}^d} \|\mathbf{X}\mathbf{w}-\mathbf{y}\|_2^2 + \lambda\|\mathbf{w}\|_1$$

- How do you pick the regularization parameter, lambda?



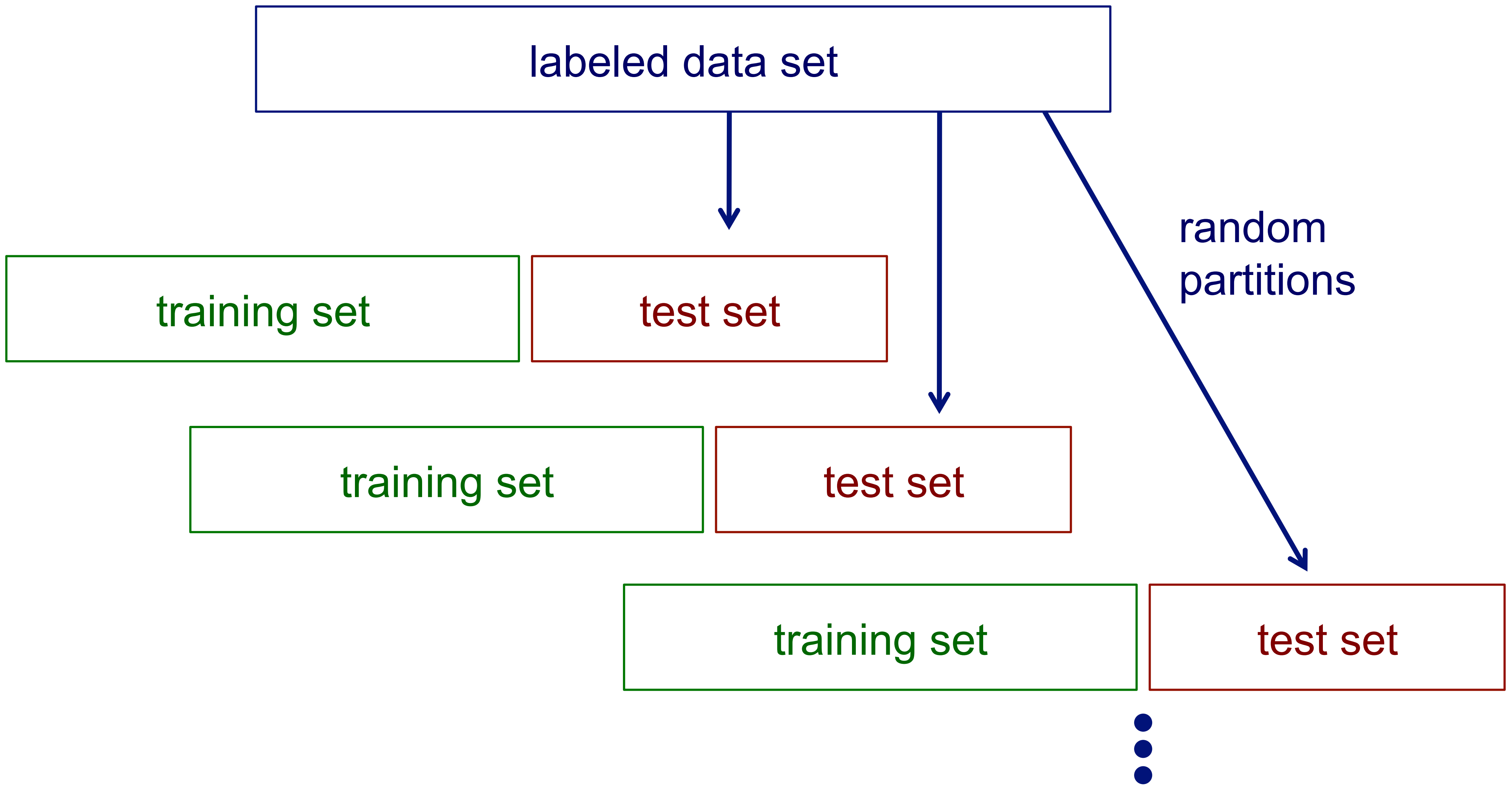
Partition training data into separate training/validation sets

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

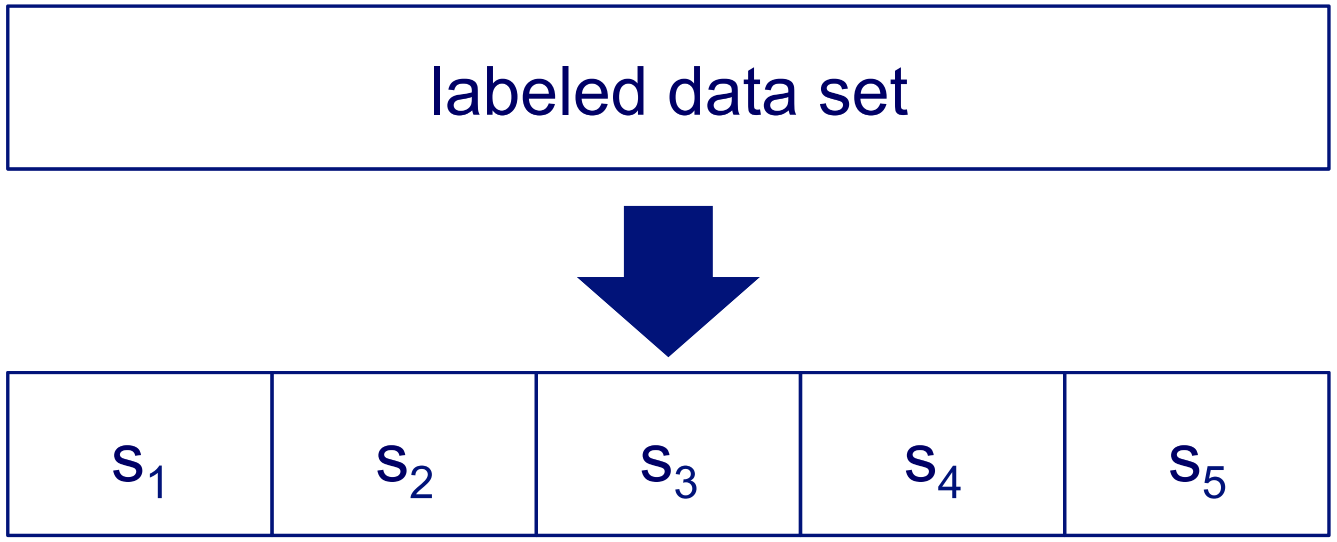
Random resampling

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



Cross validation

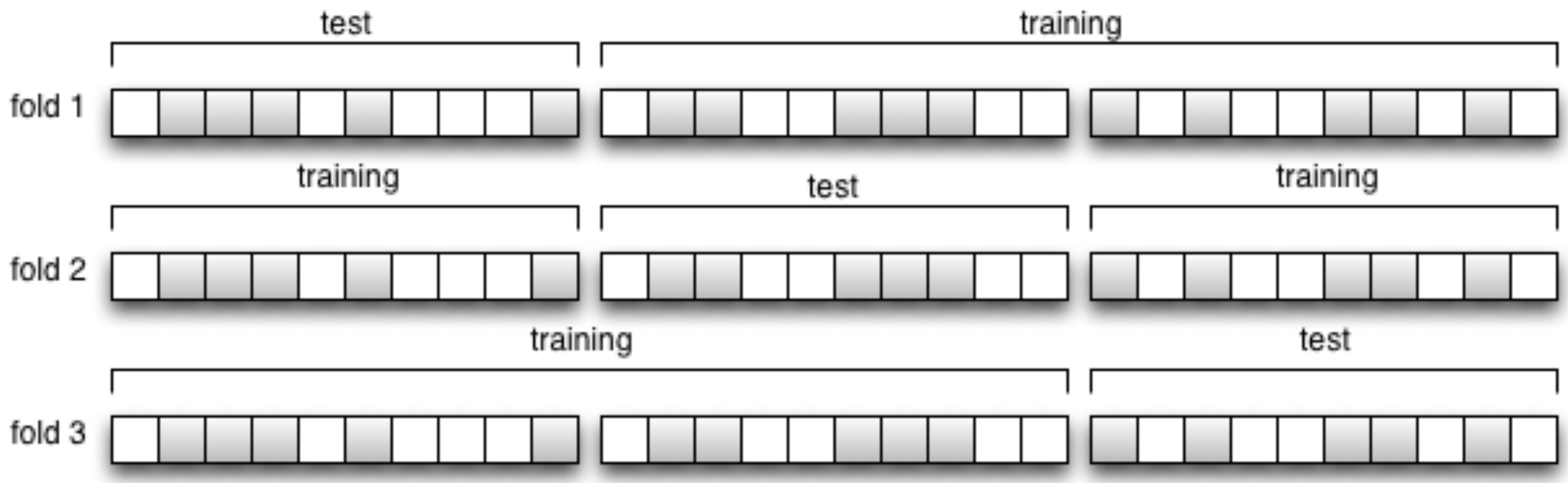
partition data
into *n* subsamples



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

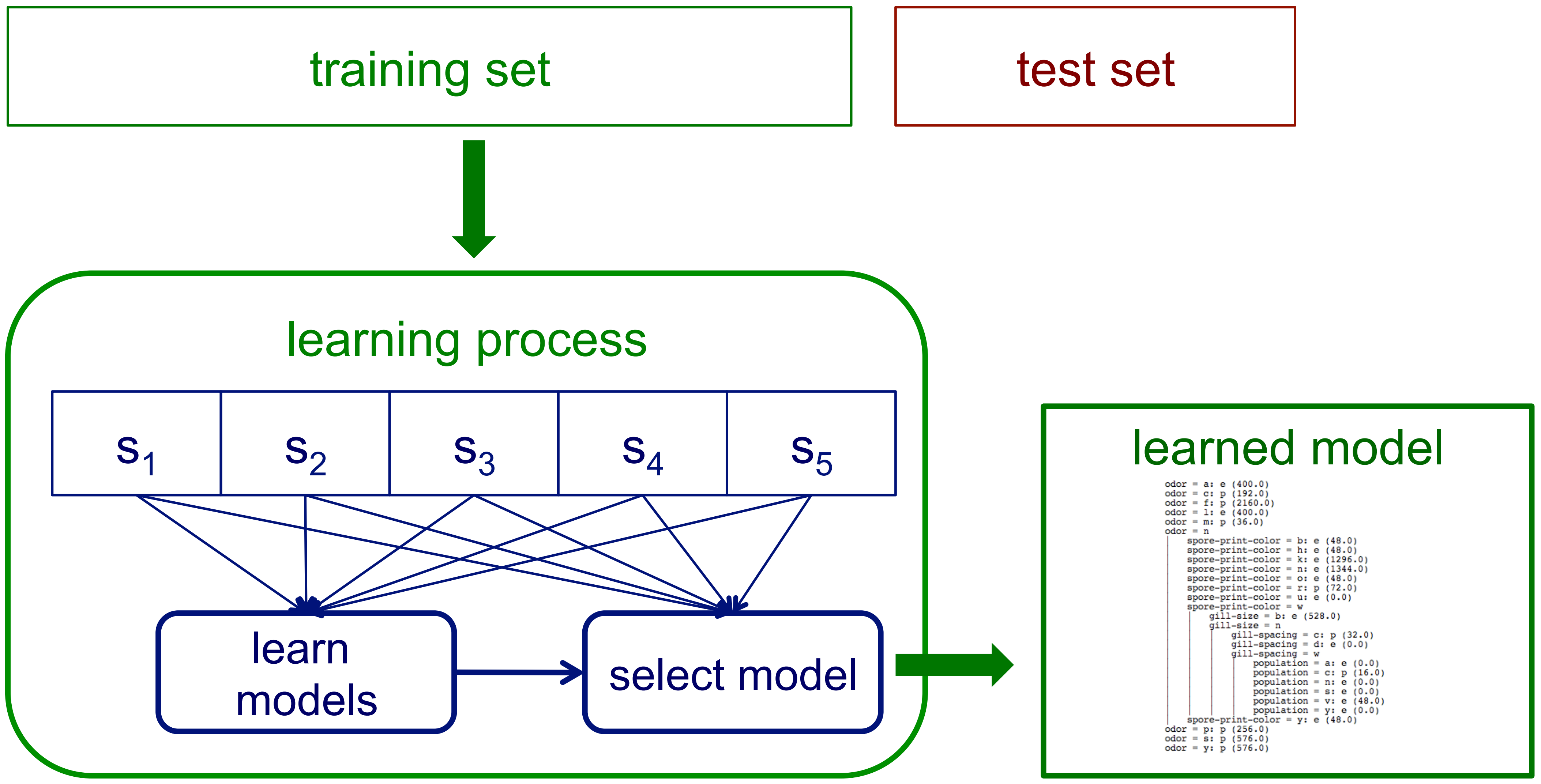
accuracy = 73/100 = 73%

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
- 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 model</u>

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)



```
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)
```

Example: selecting the regularization parameter with CV

- Given a training set

```
1. partition the training set into n folds, s1, …, sn
2. for each value of lambda considered
     • for i = 1 to n
         • learn regression model using all folds but si
         • evaluate accuracy on si
3. select lambda that resulted in the best accuracy for s1, …, sn
4. learn model using entire training set and selected lambda
```

- This is typically run separately for each training set
  - What would it mean to 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 approximates how a learned model performs, when training on that much data
    - e.g., multiple training/test splits approximates performance of a learning method, with given hyperparameters
    - 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

Practical questions

- What are possible hyperparameters that you may have to deal with in machine learning?
- What does it mean if we get 10 measures of accuracy, and they are quite different from each other?
- What if we have more than one dataset?
- What might an experiment look like, that tests learning speeds?
    - measuring speed is straightforward. Can't we just test the learning speed on the training set, for our different algorithms?

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

- Confidence interval around expected error Xbar 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)

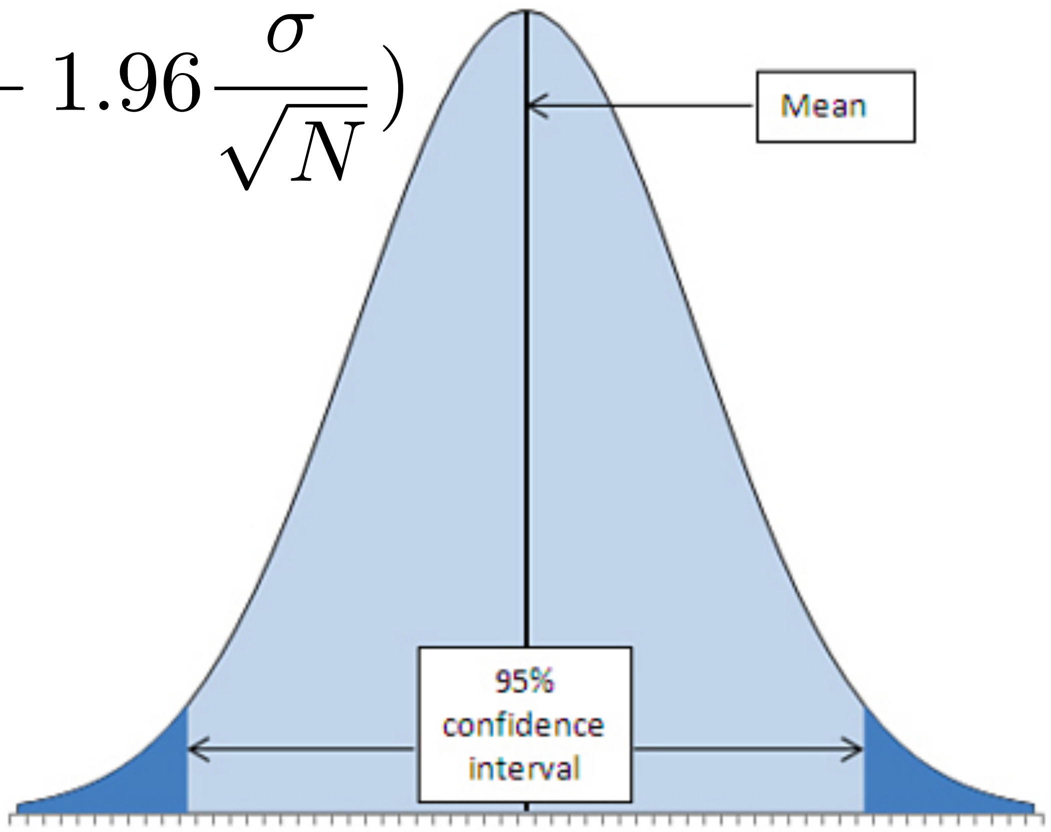
- If the two confidence intervals do overlap, need to use a statistical significance test
  - see http://www.cs.iastate.edu/~honavar/dietterich98approximate.pdf for a comparison of the performance of several statistical tests

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

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



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

	Accuracies on test sets				
System 1:	80%	50	75	…	99
System 2:	79	49	74	…	98
δ :	+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

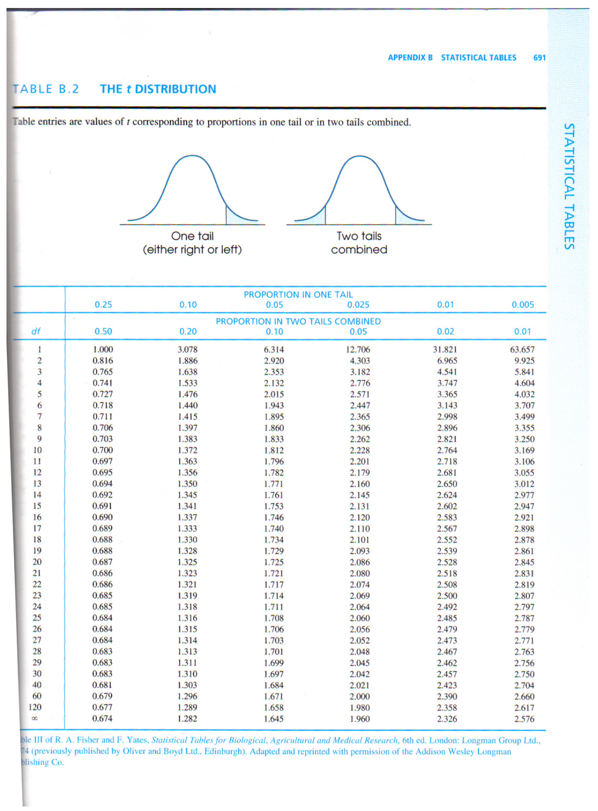


TABLE B.2 THE *t* DISTRIBUTION

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

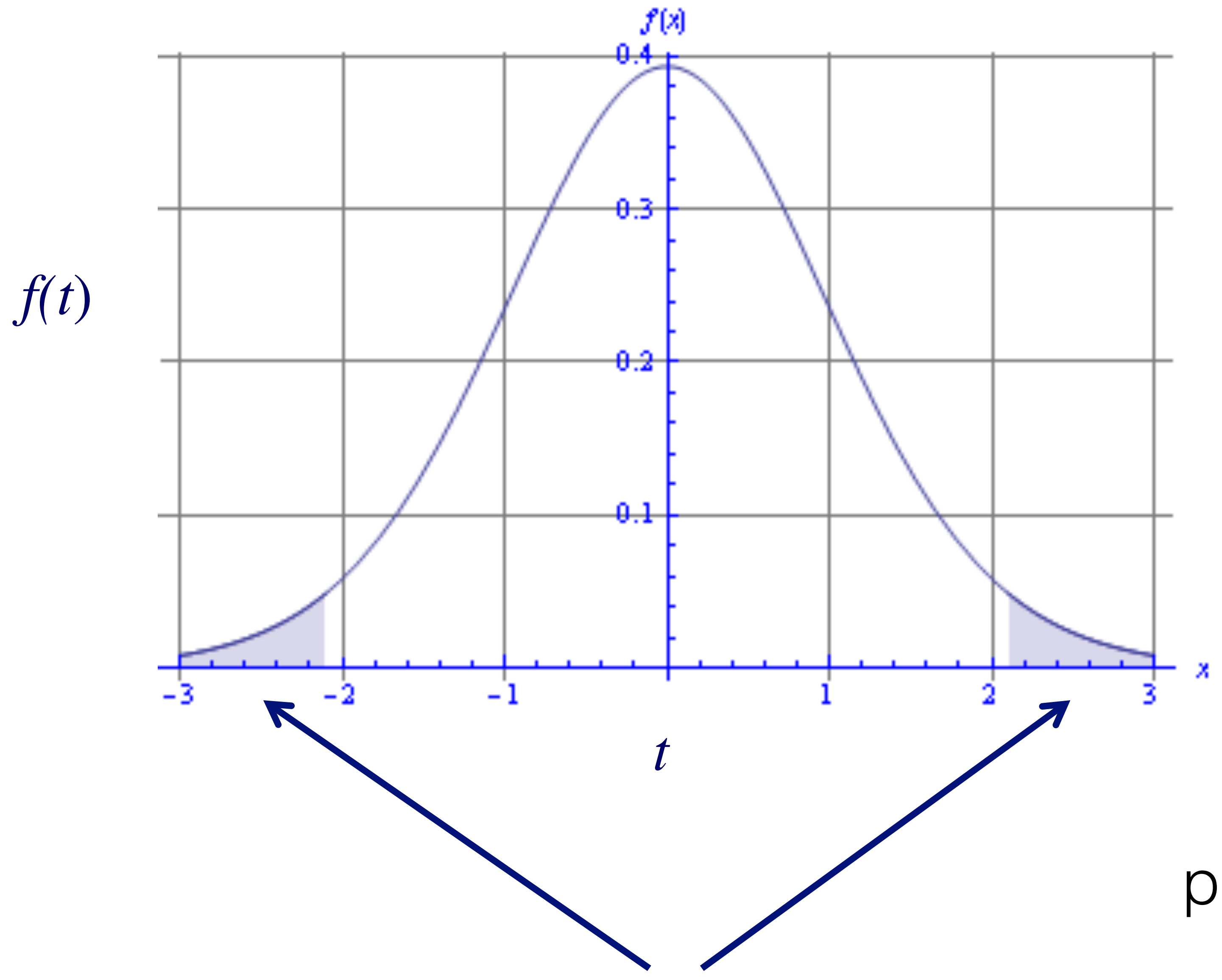
One tail
(either right or left)
Two tails
combined

	PROPORTION IN ONE TAIL					
	0.25	0.10	0.05	0.025	0.01	0.005
	PROPORTION IN TWO TAILS COMBINED					
df	0.50	0.20	0.10	0.05	0.02	0.01
1	1.000	3.078	6.314	12.706	31.821	63.657
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	1.383	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
∞	0.674	1.282	1.645	1.960	2.326	2.576

ble III of R. A. Fisher and F. Yates, *Statistical Tables for Biological, Agricultural and Medical Research*, 6th ed. London: Longman Group Ltd., 74 (previously published by Oliver and Boyd Ltd., Edinburgh). Adapted and reprinted with permission of the Addison Wesley Longman blishing Co.

STATISTICAL TABLES

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

p is the probability of seeing Xbar, under our null hypothesis

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
  - Random sampling approaches are more robust to this cheating
- 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

Why might random sampling be more robust?

- Imagine a benchmark dataset
  - e.g., MNIST is a character recognition dataset, split into one training and test set
- One paper reports that a specific neural network did really well, with a narrow regularization range
- Now you build on this, adding say a few more nodes, increase the number of regularization parameters in that range and train and report test accuracy
- Lo-and-behold, you do better than the previous model!
- Is there an issue?

Exercise: Generating learning curves

- Imagine you want to test the sample efficiency of logistic regression and decision trees
  - An algorithm is sample efficient if it can get good generalization performance using a small number of samples
  - For example, Algorithm 1 needs at least 1000 samples whereas Algorithm 2 only needs about 100 samples, to reach a similar level of performance on test data
  - Which models might be more or less sample efficient?
- How would you do this?

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*

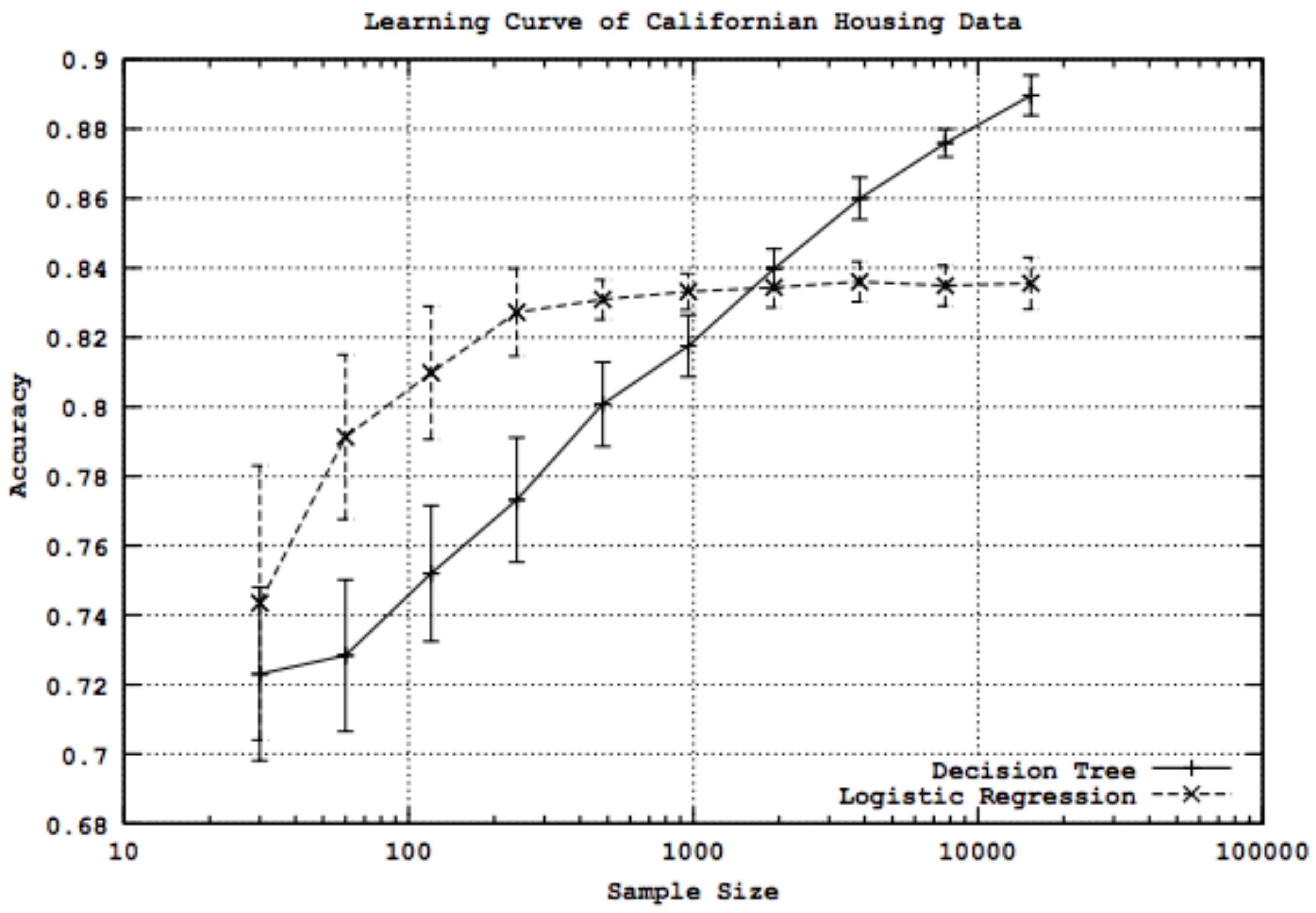
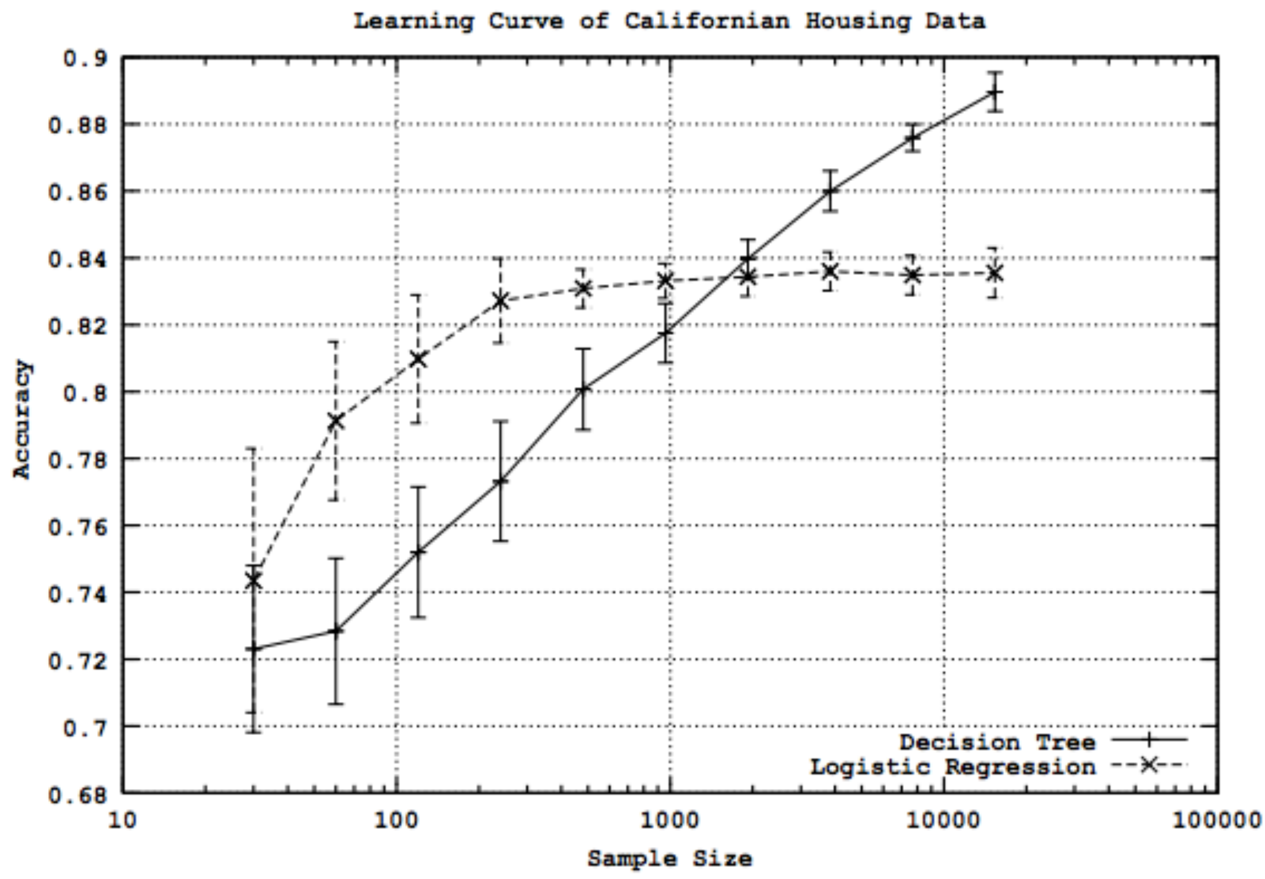


Figure from Perlich et al. *Journal of Machine Learning Research*, 2003

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)



Exercise: Generating learning curves (cont.)

- What if you want to measure sample complexity
  - This is how many samples are needed, to reach the optimal function in your function class
- How would you generate these learning curves? What is the metric?