MPSSDEGB
कौशल से रोजगार, समृद्धि अपार

FITT
Foundation For Innovation And Technology Transfer

SANKALP
Ministry of Skill Development
& Entrepreneurship

DAY 9 - Evaluating Machine Learning Methods

.

Jan, 2024

Contents

- test sets
- learning curves
- validation (tuning) sets
- stratified sampling
- cross validation
- internal cross validation
- confusion matrices
- TP, FP, TN, FN
- ROC curves
- confidence intervals for error
- pairwise *t*-tests for comparing learning systems
- scatter plots for comparing learning systems
- lesion studies

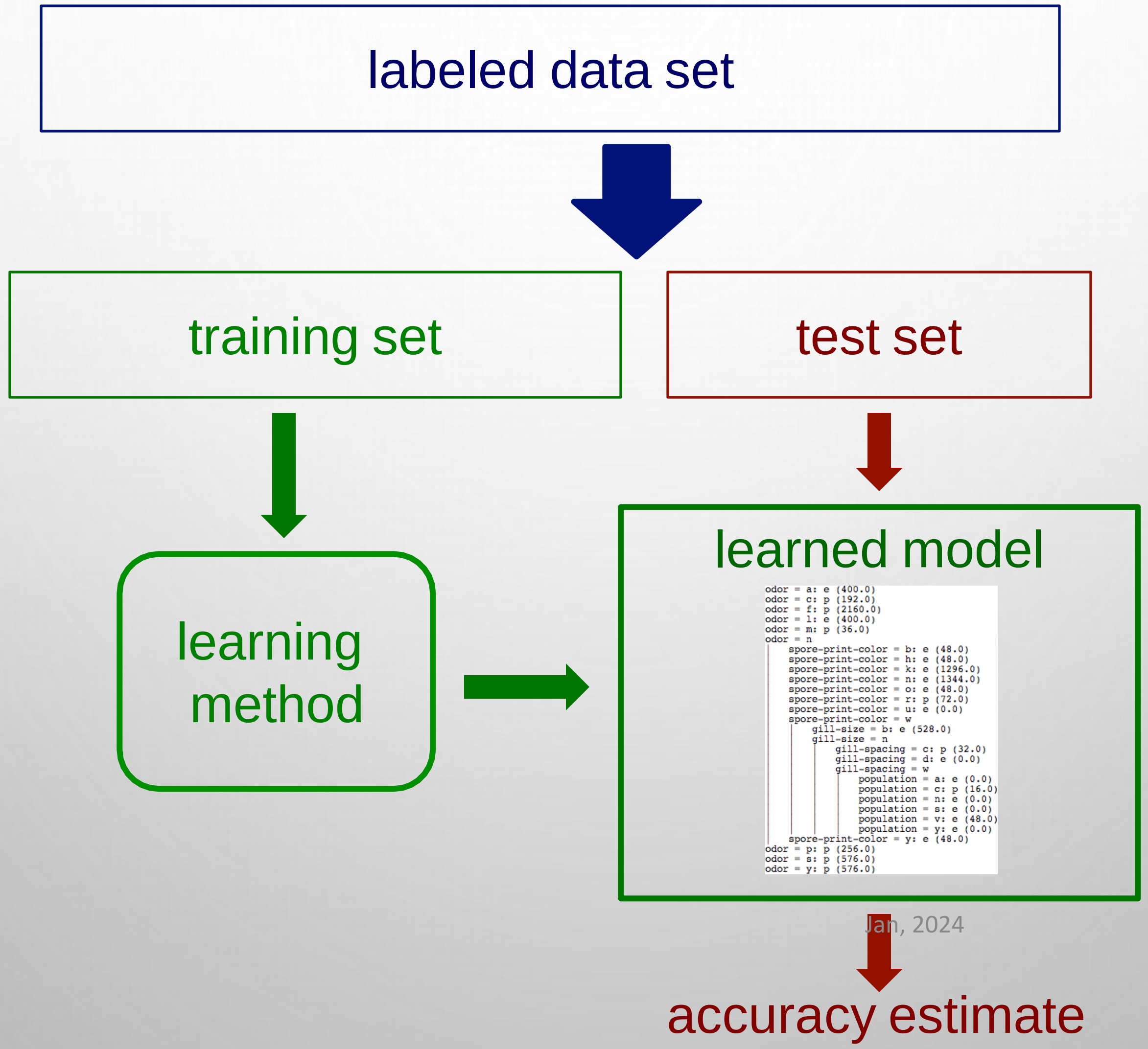
Contents

- recall/sensitivity/true positive rate (TPR)
- precision/positive predictive value (PPV)
- specificity and false positive rate (FPR or 1-specificity)
- precision-recall (PR) curves

Jan, 2024

Test sets revisited

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



labeled data set

training set

test set

learning method

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)
```

accuracy estimate

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

* In some applications it is reasonable to assume that you have access to the feature vector (i.e. x) but not the y part of each test instance.

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*

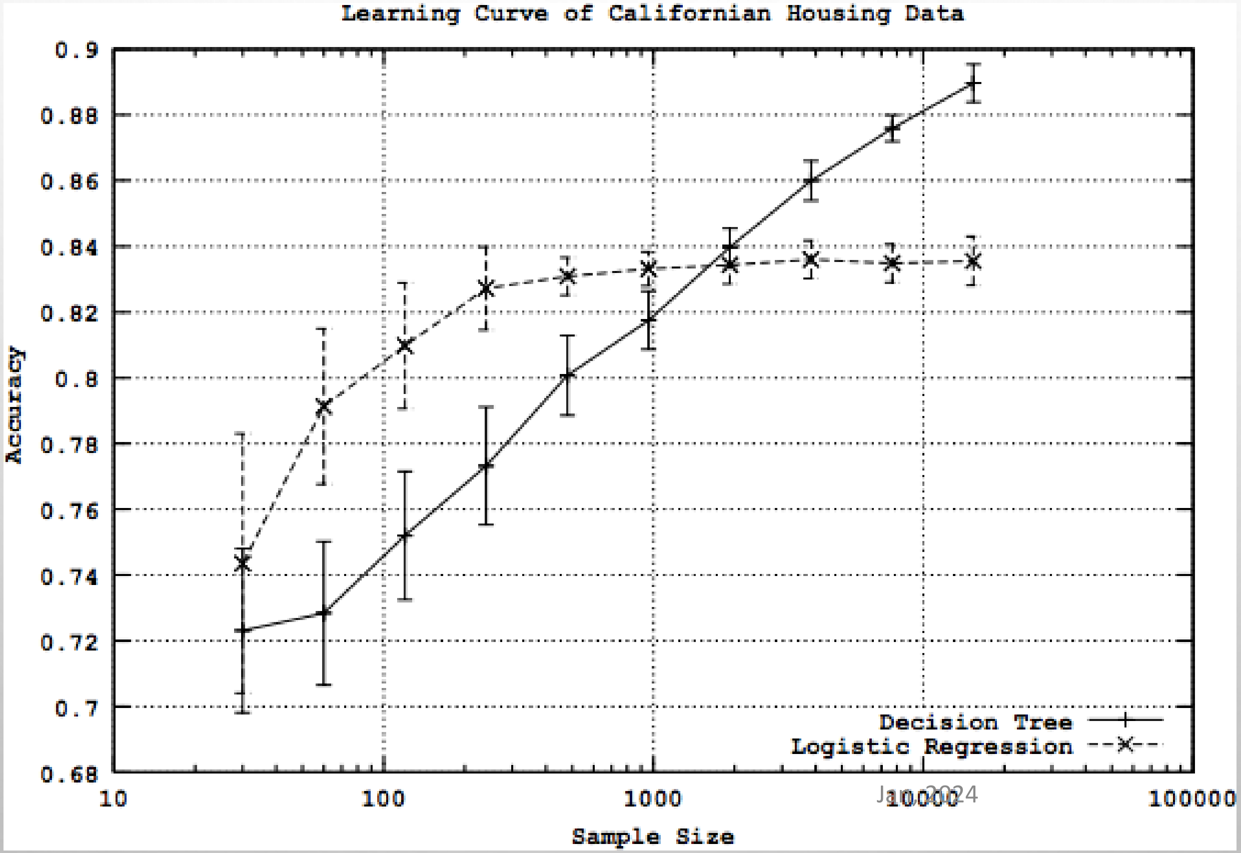
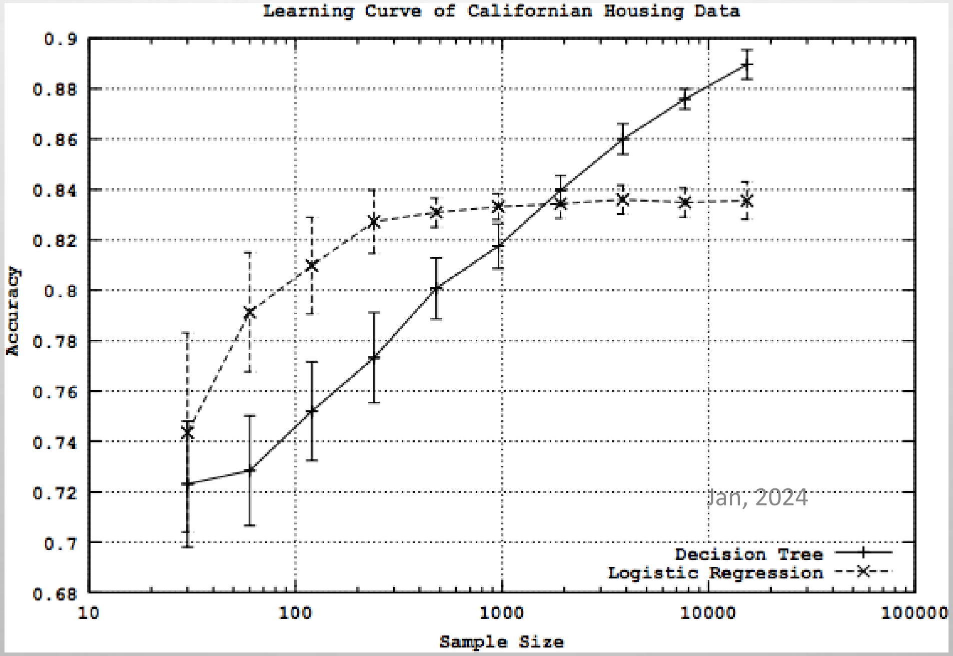


Figure from Perlich et al. *Journal of Machine Learning Research*, 2003

Learning curves

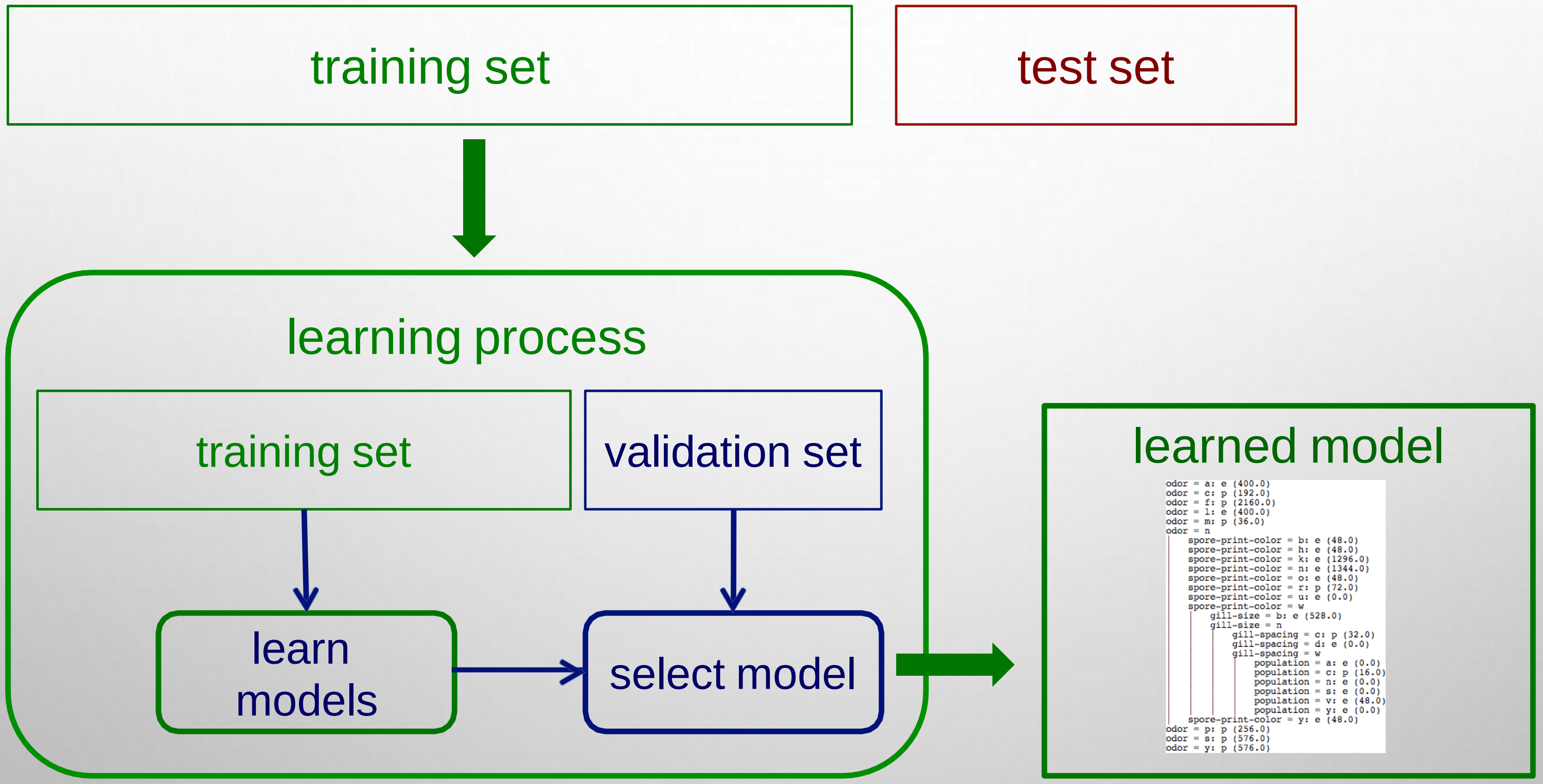
given training/test set partition

- for each sample size s on learning curve
  - (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)



Validation (tuning) sets revisited

Suppose we want unbiased estimates of accuracy during the learning process (e.g. to choose the best level of decision-tree pruning)?



```
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

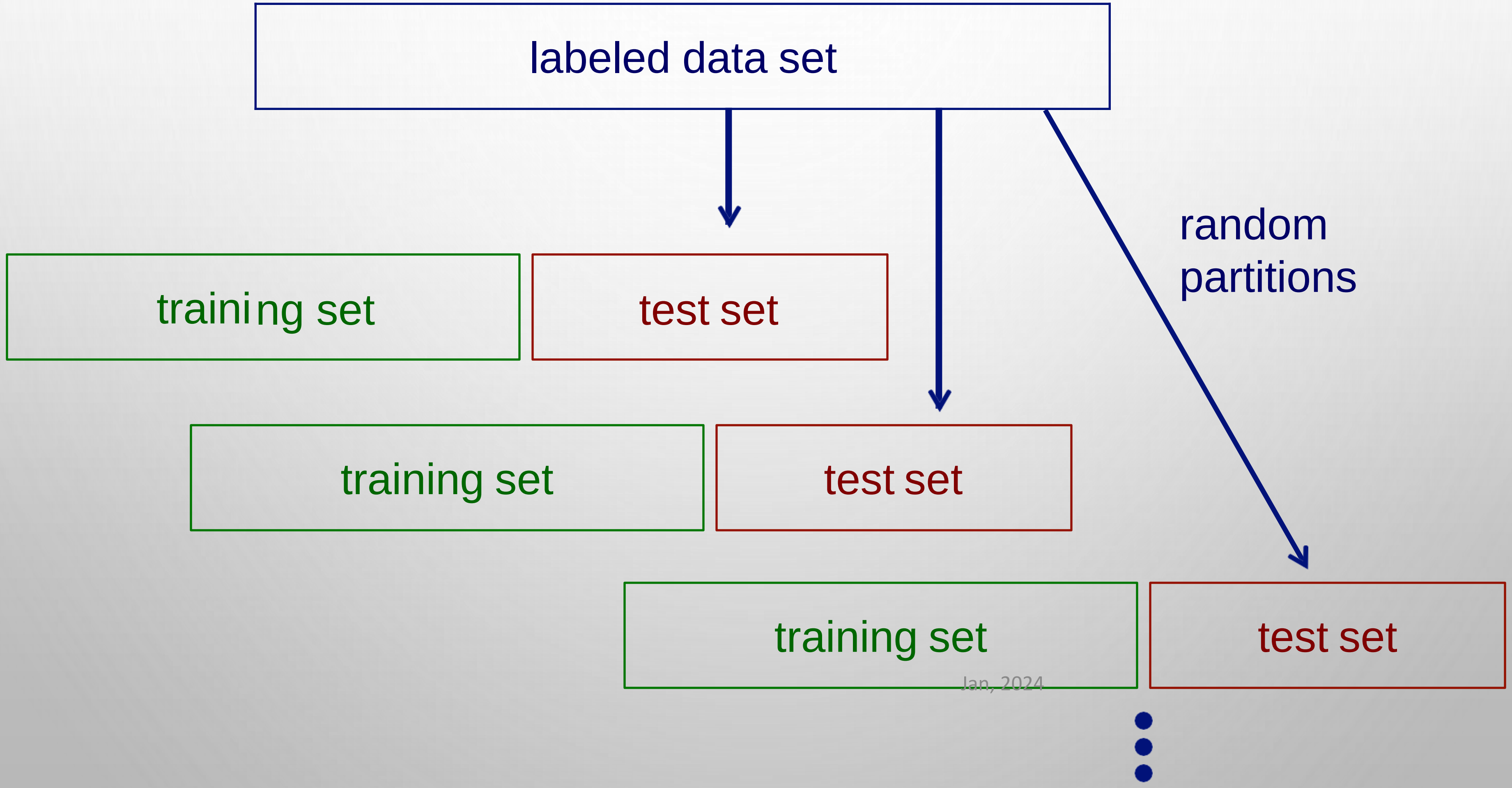
Limitations of using a single training/test partition

- 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.



labeled data set

random partitions

training set

test set

training set

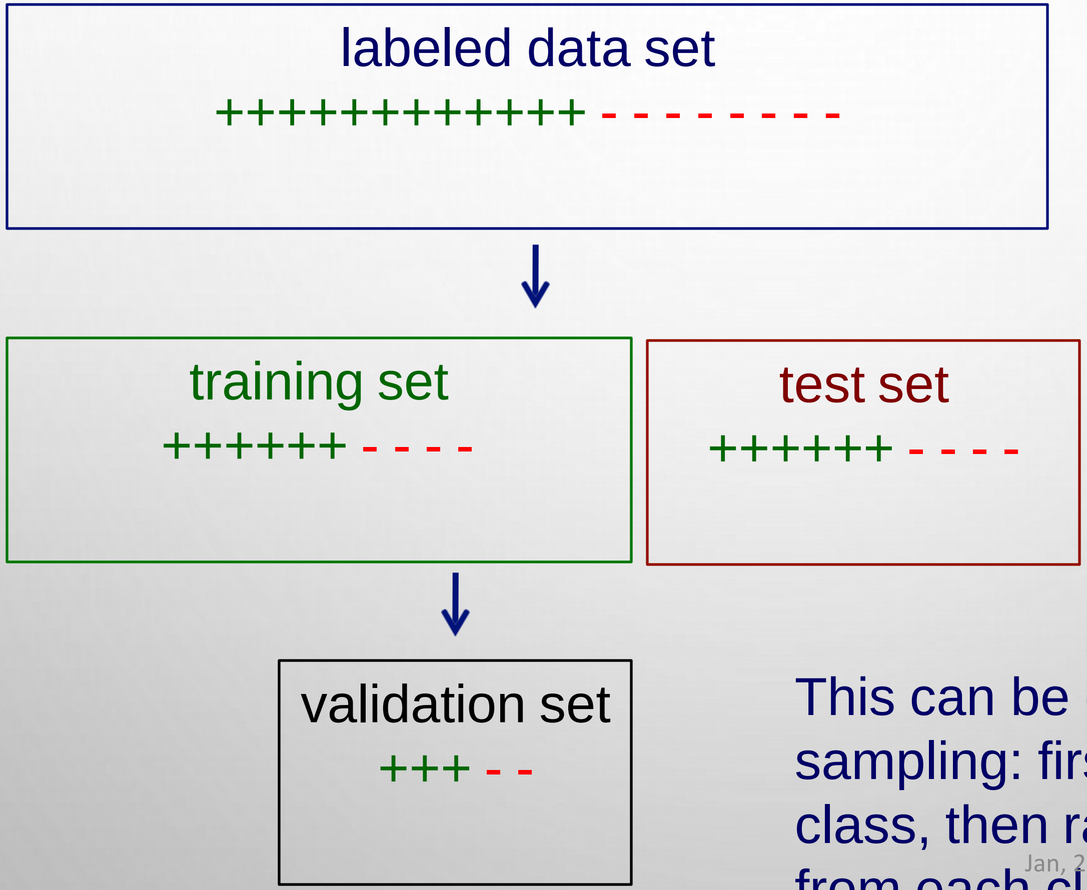
test set

training set

test set

Stratified sampling

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



labeled data set

++++++++++++ - - - - - - - -

training set

++++++ - - - -

test set

++++++ - - - -

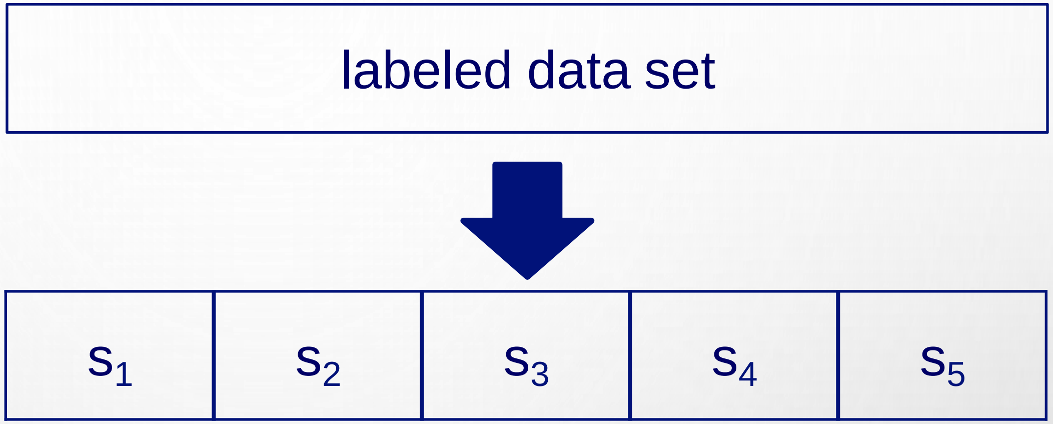
validation set

+++ - -

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



labeled data set

s_1 s_2 s_3 s_4 s_5

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

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

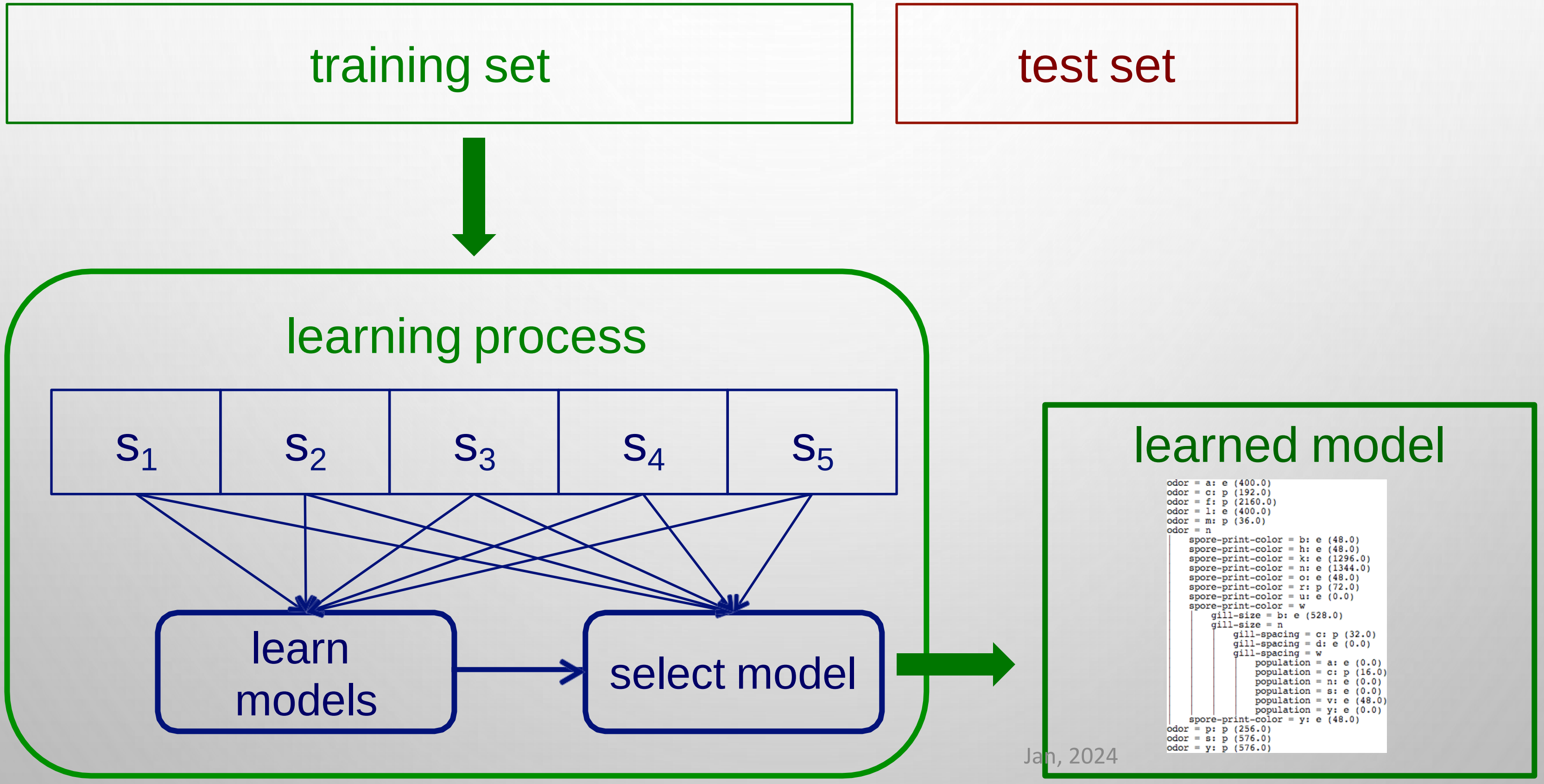
accuracy = 73/100 = 73%

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 level of decision-tree pruning)



```
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: using internal cross validation to select *k* in *k*-NN

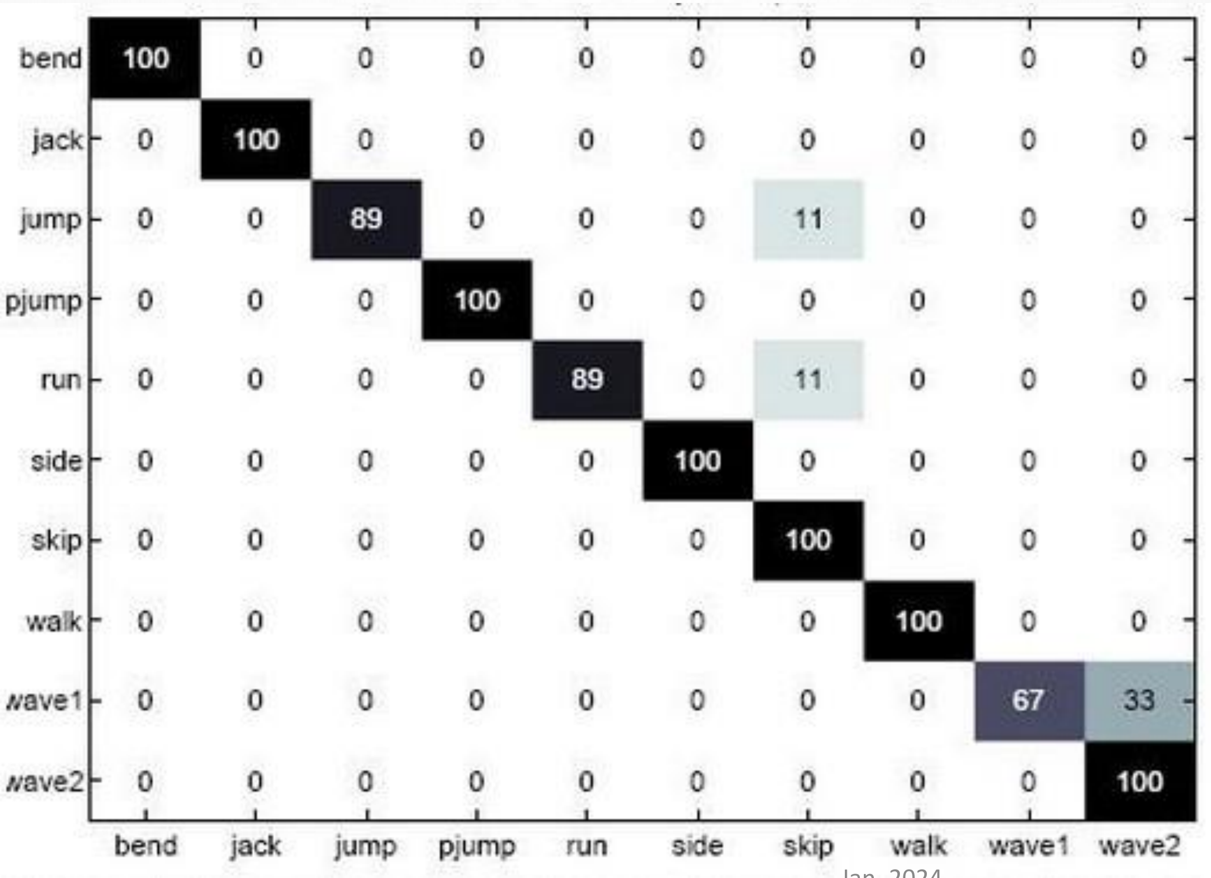
```
given a training set
    1.  partition training set into $n$ folds, $s_1 \dots s_n$
    2.      for each value of $k$
            considered for $i$ = 1 to $n$
                learn $k$-NN model using all folds but $s_i$
                evaluate accuracy on $s_i$
    3.  select $k$ that resulted in best accuracy for $s_1 \dots s_n$
    4.  learn model using entire training set and selected $k$
```

the steps inside the box are run independently for each training set (i.e. if we're using 10-fold CV to measure the overall accuracy of our *k*-NN approach, then the box would be executed 10 times)

Confusion matrices

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

activity recognition from video



	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

actual class

predicted class

figure from vision.jhu.edu

Confusion matrix for 2-class problems

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

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

Is accuracy an adequate measure of predictive performance?

- accuracy may not be useful measure in cases where
  - there is a large class skew
    - Is 98% accuracy good if 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 accuracy metrics

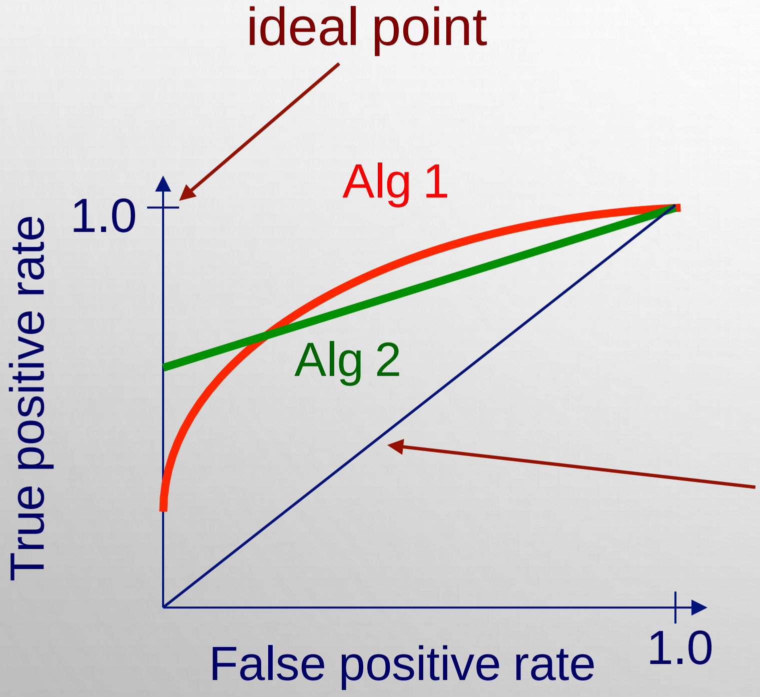
		actual class	
		positive	negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

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

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

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



expected curve for random guessing

Different methods can work better in different parts of ROC space. This depends on cost of false + vs. false -

ROC curve example

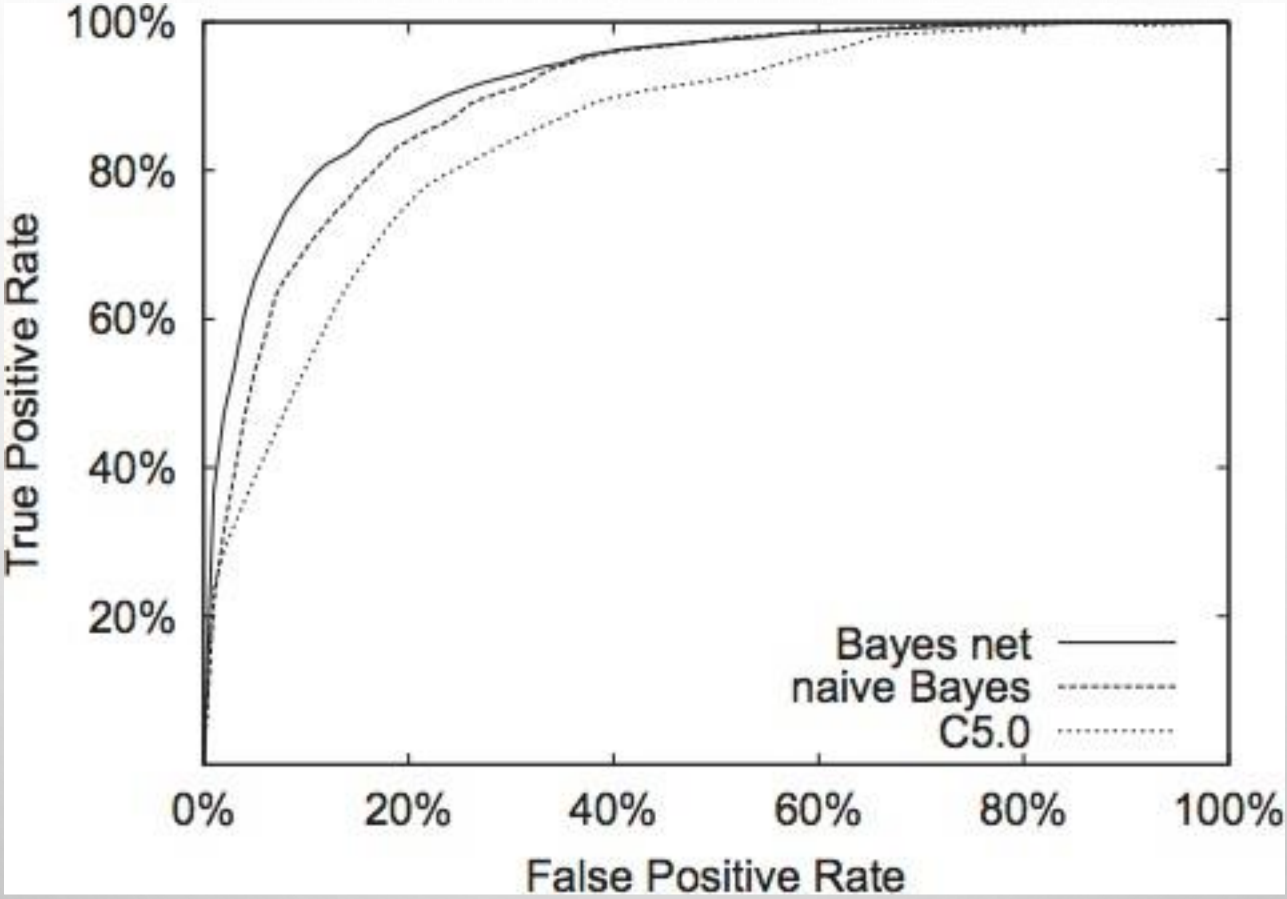
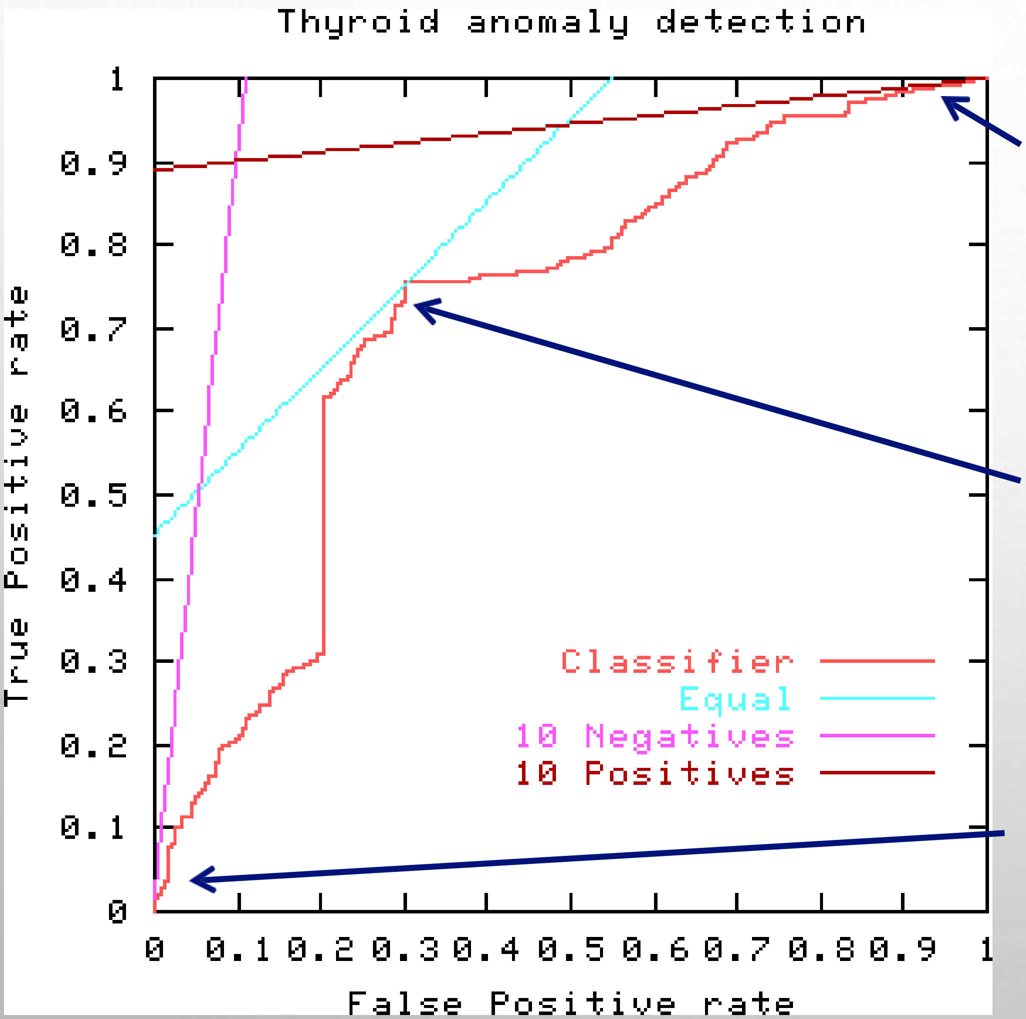


figure from Bockhorst et al., *Bioinformatics* 2003

ROC curves and misclassification costs



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

Algorithm for creating an ROC curve

1. sort test-set predictions according to confidence that each instance is positive

2. step through sorted list from high to low confidence

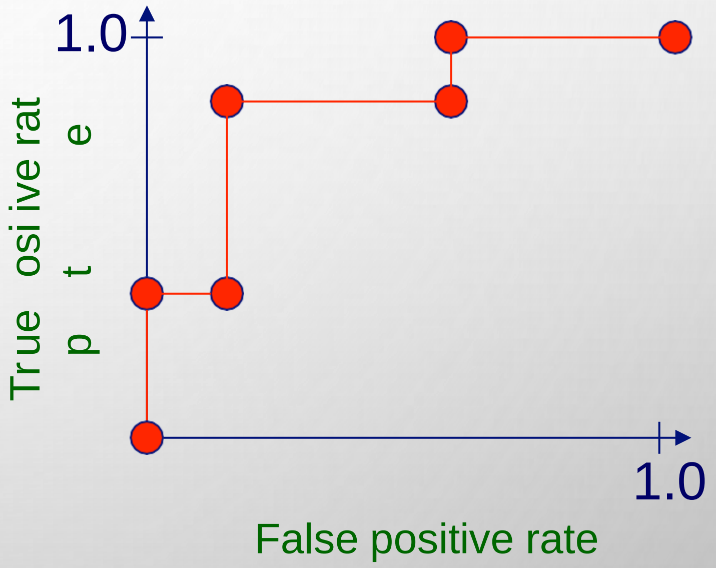
   i. locate a *threshold* between
      instances with opposite classes (keeping instances with the same confidence value on the same side of threshold)

   ii. compute TPR, FPR for instances above threshold

   iii. output (FPR, TPR) coordinate

Plotting an ROC curve

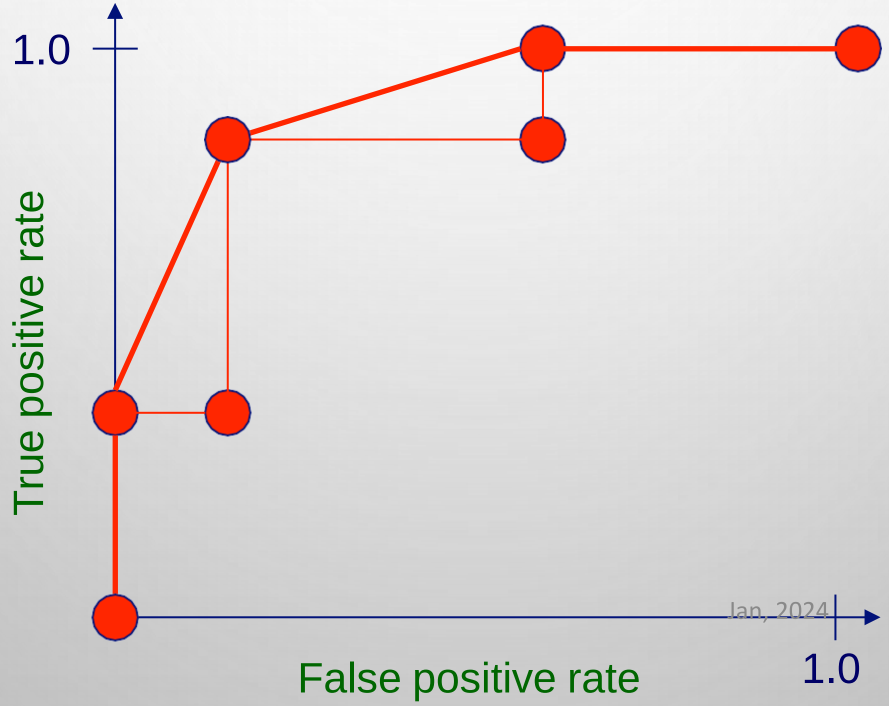
instance	confidence positive		correct class
Ex 9	.99		+
Ex 7	.98	TPR= 2/5, FPR= 0/5	+
Ex 1	.72	TPR= 2/5, FPR= 1/5	-
Ex 2	.70		+
Ex 6	.65	TPR= 4/5, FPR= 1/5	+
Ex 10	.51		-
Ex 3	.39	TPR= 4/5, FPR= 3/5	-
Ex 5	.24	TPR= 5/5, FPR= 3/5	+
Ex 4	.11		-
Ex 8	.01	TPR= 5/5, FPR= 5/5	-



Plotting an ROC curve

can interpolate between points to get *convex hull*

- convex hull: repeatedly, while possible, perform interpolations that skip one data point and discard any point that lies below a line
- interpolated points are achievable in theory: can flip weighted coin to choose between classifiers represented by plotted points



ROC curves

Does a low false-positive rate indicate that most positive predictions (i.e. predictions with confidence > some threshold) are correct?

suppose our TPR is 0.9, and FPR is 0.01

fraction of instances that are positive	fraction of p ositive predictions that are correct
0.5	0.989
0.1	0.909
0.01	0.476
0.001	0.083

Other accuracy metrics

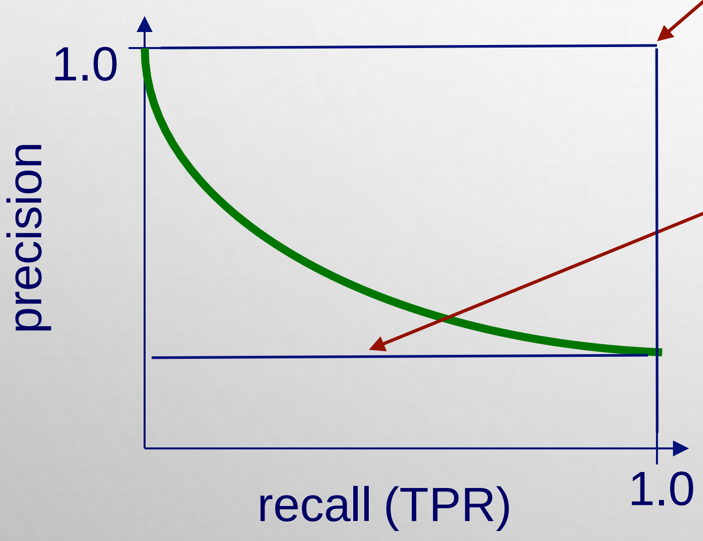
		actual class	
		positive	negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

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

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision/recall curves

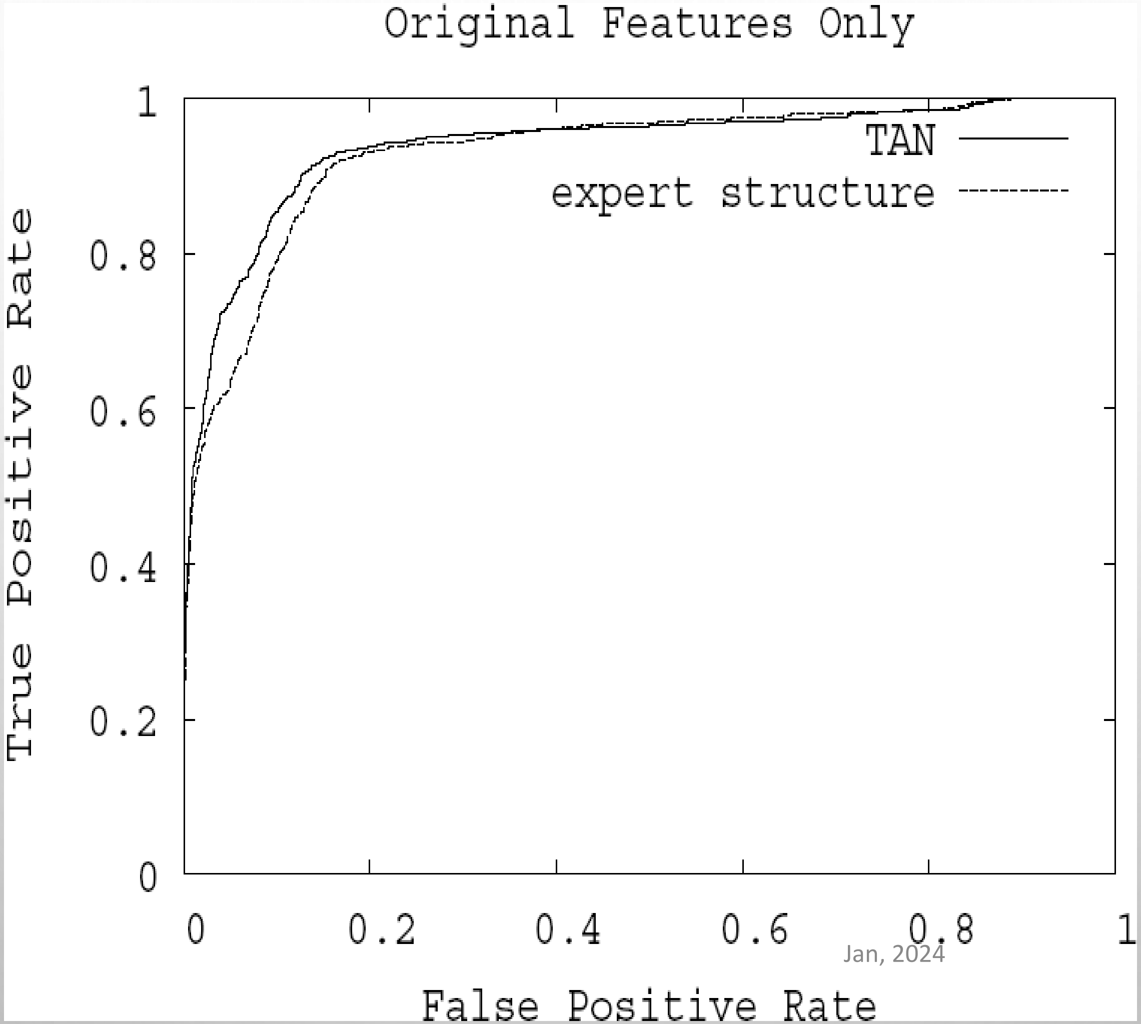
A *precision/recall curve* plots the precision vs. recall (TP-rate) as a threshold on the confidence of an instance being positive is varied



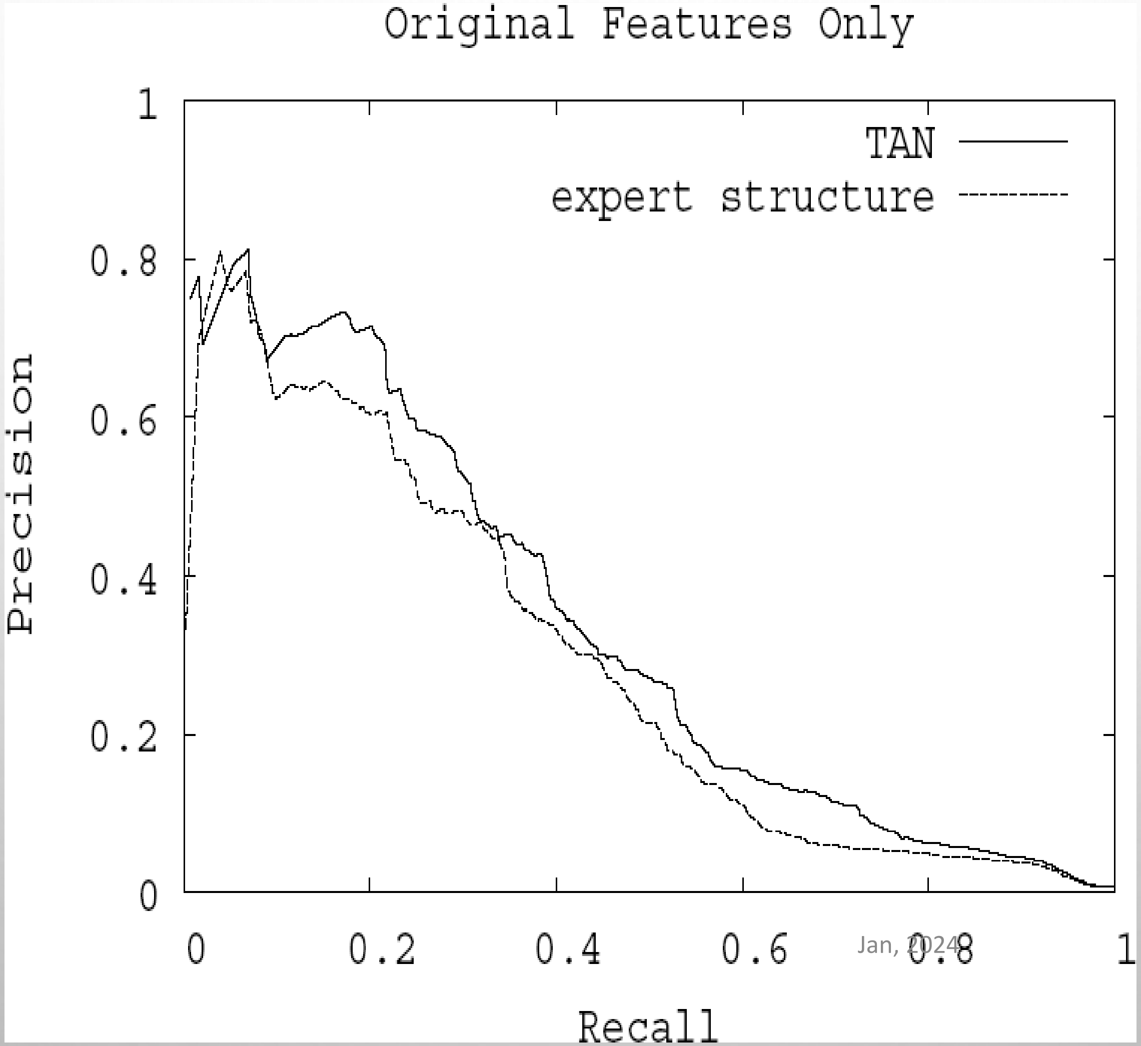
ideal point

default precision determined by the fraction of instances that are positive

Mammography Example: ROC



Mammography Example: PR



How do we get one ROC/PR curve when we do cross validation?

Approach 1

- make assumption that confidence values are comparable across folds
- pool predictions from all test sets
- plot the curve from the pooled predictions

Approach 2 (for ROC curves)

- plot individual curves for all test sets
- view each curve as a function
- plot the average curve for this set of functions

Comments on ROC and 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 fraction of predictions that are false positives
- well suited for tasks with lots of negative instances

To Avoid Cross-Validation Pitfalls, Ask:

- 1. Is my held-aside test data really representative of going out to collect new data?
  - Even if your methodology is fine, someone may have collected features for positive examples differently than for negatives – should be randomized
  - Example: samples from cancer processed by different people or on different days than samples for normal controls

To Avoid Pitfalls, Ask:

- 2. Did I repeat my entire data processing procedure on every fold of cross-validation, using only the training data for that fold?
  - On each fold of cross-validation, did I ever access in any way the label of a test case?
  - Any preprocessing done over entire data set (feature selection, parameter tuning, threshold selection) must not use labels

To Avoid Pitfalls, Ask:

- 3. Have I modified my algorithm so many times, or tried so many approaches, on this same data set that I (the human) am overfitting it?
  - Have I continually modified my preprocessing or learning algorithm until I got some improvement on this data set?
  - If so, I really need to get some additional data now to at least test on

Confidence intervals on error

Given the observed error (accuracy) of a model over a limited sample of data, how well does this error characterize its accuracy over additional instances?

Suppose we have

- a learned model h
- a test set S containing n instances drawn independently of one another and independent of h
- $n \geq 30$
- h makes r errors over the n instances

our best estimate of the error of h is

$$error_S(h) = \frac{r}{n}$$

With approximately N% probability, the true error lies in the interval

Confidence intervals on error

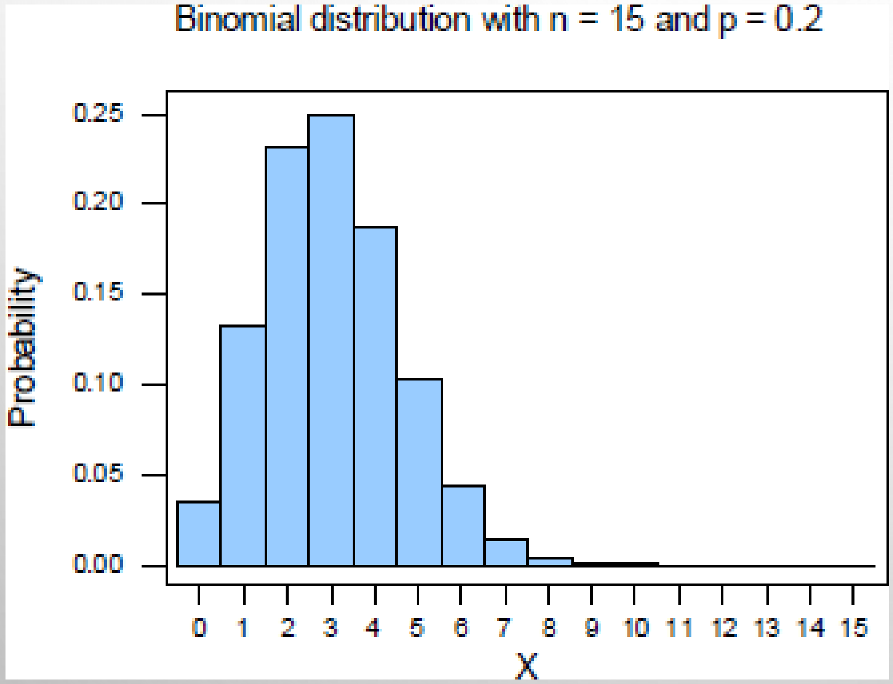
$$error_S(h) = z_N\sqrt{\frac{error_S(h)(1 - error_S(h))}{n}}$$

where z_N is a constant that depends on N (e.g. for 95% confidence, z_N =1.96)

Confidence intervals on error

How did we get this?

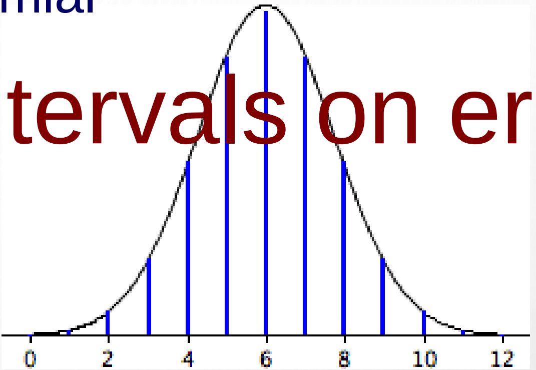
1. Our estimate of the error follows a binomial distribution given by n and p (the true error rate over the data distribution)



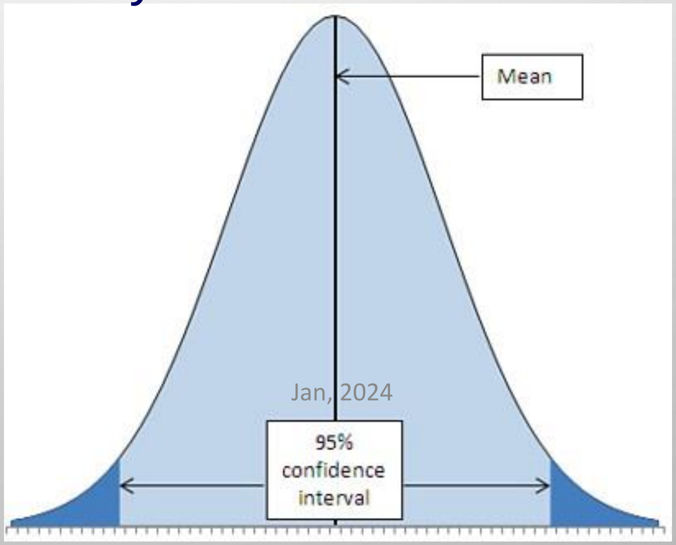
2. Simplest (and most common) way to determine a binomial confidence interval is to use the *normal approximation*

Confidence intervals on error

2. When $n \geq 30$, and p is not too extreme, the normal distribution is a good approximation to the binomial



3. We can determine the N% confidence interval by determining what bounds contain N% of the probability mass under the normal



Empirical Confidence Bounds

- Bootstrapping: Given n examples in data set, randomly, uniformly, independently (with replacement) draw n examples – bootstrap sample
- Repeat 1000 (or 10,000) times:
  - Draw bootstrap sample
  - Repeat entire cross-validation process
- Lower (upper) bound is result such that 2.5% of runs yield lower (higher)

Comparing learning systems

How can we determine if one learning system provides better performance than another

- for a particular task?
- across a set of tasks / data sets?

Motivating example

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

- 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

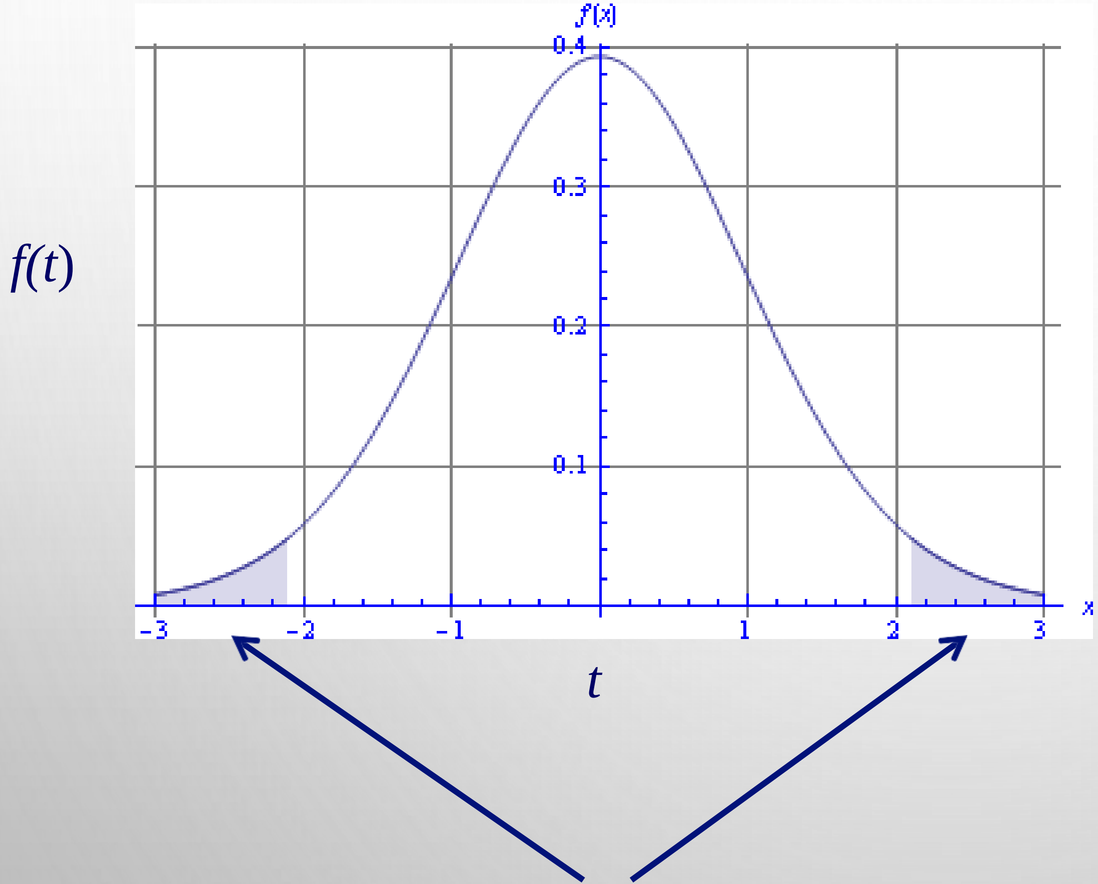
Comparing systems using a paired t test

- consider δ's as observed values of a set of i.i.d. random variables

- *null hypothesis*: the 2 learning systems have the same accuracy
- *alternative hypothesis*: one of the systems is more accurate than the other

- hypothesis test:
  - use paired t-test to determine probability p that mean of δ's would arise from null hypothesis
  - if p is sufficiently small (typically < 0.05) then reject the null hypothesis

Comparing systems using a paired *t* test



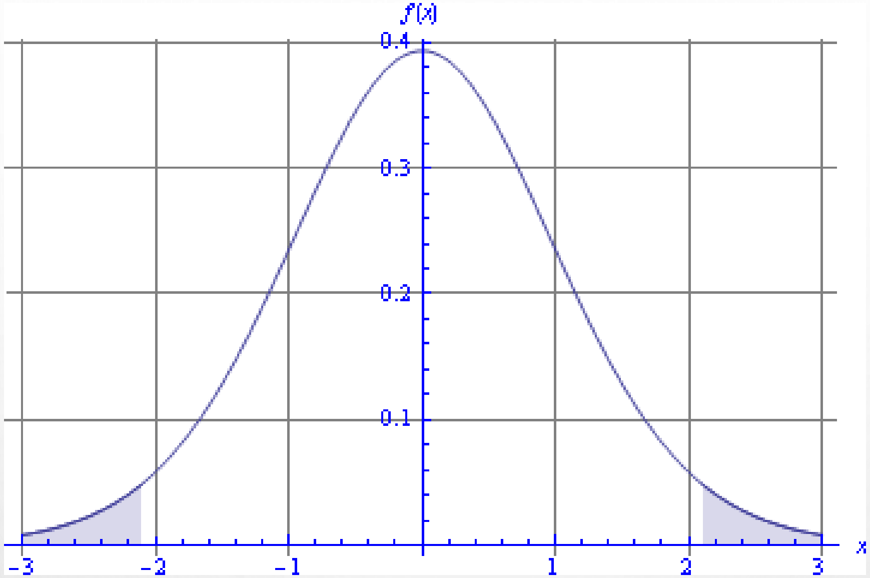
The null distribution of our *t* statistic looks like this

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

Why do we use a two-tailed test?



- a two-tailed test asks the question: is the accuracy of the two systems different
- a one-tailed test asks the question: is system A better than system B
- a priori, we don’t know which learning system will be more accurate (if there is a difference) – we want to allow that either one might be

Sign Test

- If less than 300 examples, we won’t have 30 test examples per fold
- Prefer leave-one-out cross-validation
- Count “wins” for Algorithm A and B over the N test examples on which they disagree
- Let M be the larger of these counts
- What is probability under b(N,0.5) that either A or B would win *at least* M times

Scatter plots for pairwise method comparison

We can compare the performance of two methods *A* and *B* by plotting (*A performance*, *B performance*) across numerous data sets

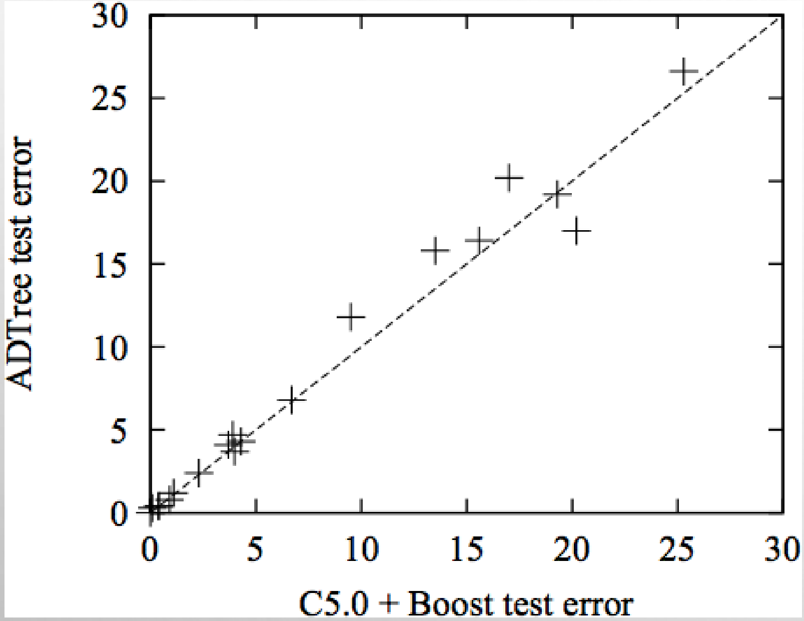


figure from Freund & Mason, *ICML* 1999

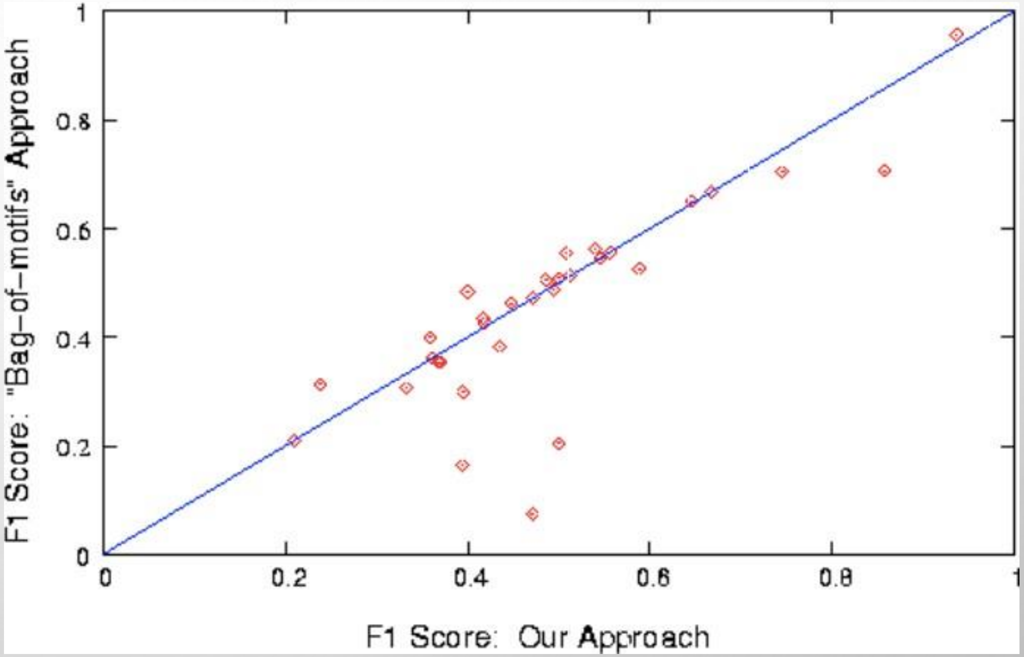
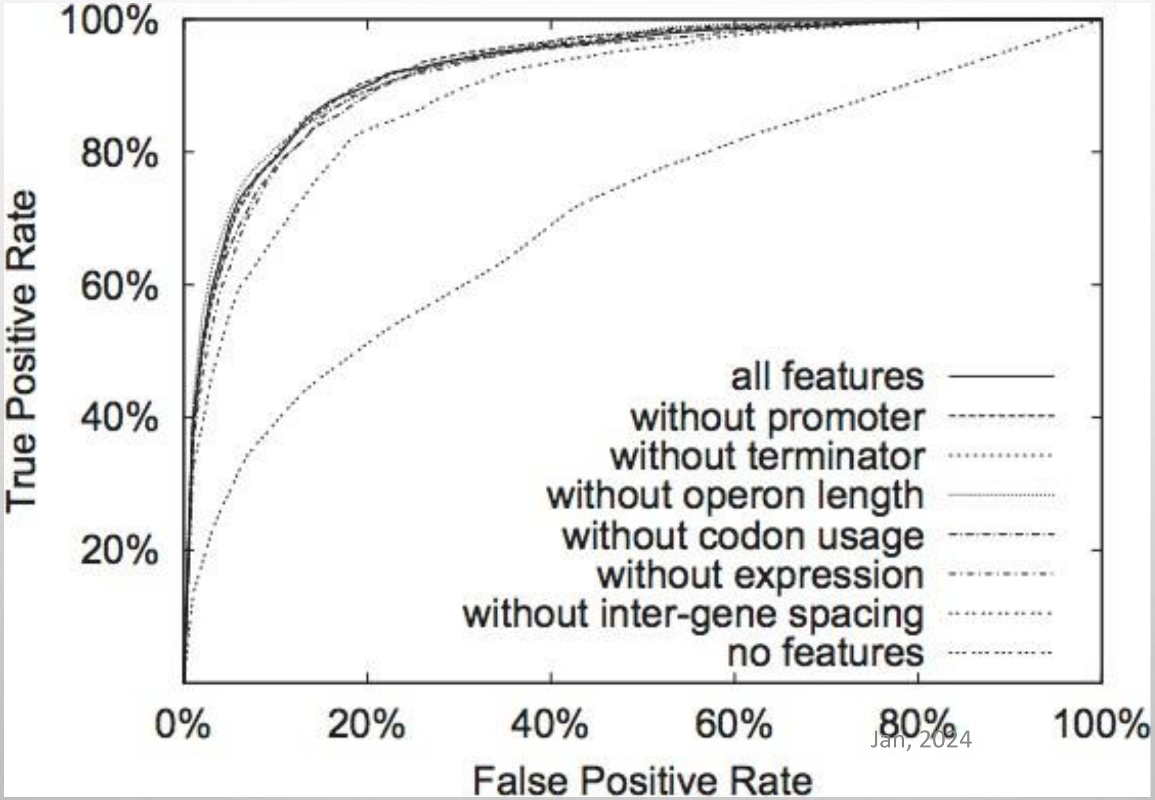


figure from Noto & Craven, *BMC Bioinformatics* 2006

Lesion studies

We can gain insight into what contributes to a learning system’s performance by removing (lesioning) components of it

The ROC curves here show how performance is affected when various feature types are removed from the learning representation



Jan, 2024

figure from Bockhorst et al., *Bioinformatics* 2003

References

1. Bockhorst, Joseph, et al. "A Bayesian network approach to operon prediction." Bioinformatics 19.10 (2003): 1227-1235.
2. Noto, Keith, and Mark Craven. "A specialized learner for inferring structured cis-regulatory modules." BMC bioinformatics 7 (2006): 1-10.
3. Freund, Yoav, and Llew Mason. "The alternating decision tree learning algorithm." icml. Vol. 99. 1999.
4. Perlich, Claudia, et al. "Machine learning for targeted display advertising: Transfer learning in action." Machine learning 95.1 (2014): 103-127.
5. Zhou, Jianlong, et al. "Evaluating the quality of machine learning explanations: A survey on methods and metrics." Electronics 10.5 (2021): 593.
6. Dalianis, Hercules, and Hercules Dalianis. "Evaluation metrics and evaluation." Clinical Text Mining: secondary use of electronic patient records (2018): 45-53.

THANKS