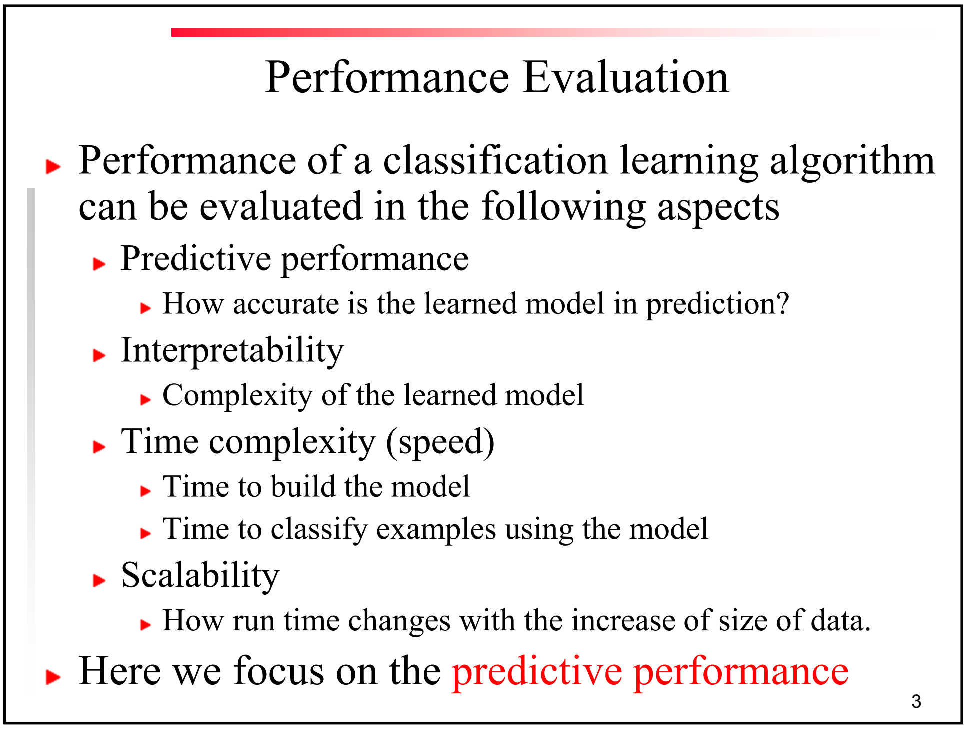
Data Mining
(EECS 6412)

Performance Evaluation of Classification Algorithms

Aijun An
Department of Electrical Engineering and Computer Science
York University

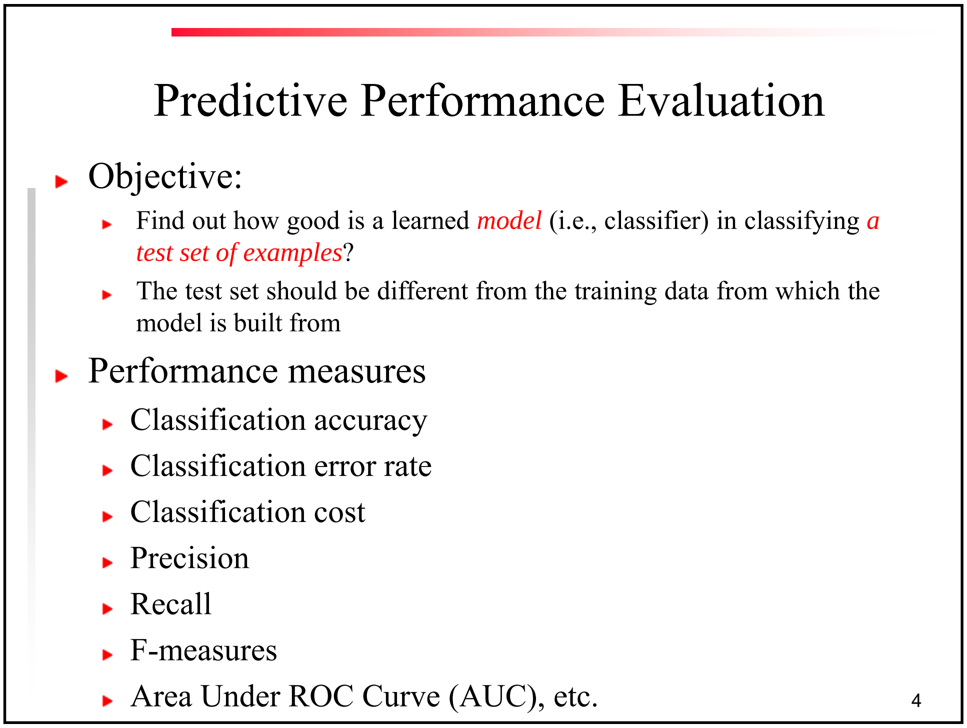
Outline

- Introduction
- Predictive performance Measures
- Performance Evaluation Methods
  - Holdout
  - Repeated holdout
  - Cross-validation
  - Bootstrap
- Methods for model comparisons
  - Significance test



Performance Evaluation

- Performance of a classification learning algorithm can be evaluated in the following aspects
  - Predictive performance
    - How accurate is the learned model in prediction?
  - Interpretability
    - Complexity of the learned model
  - Time complexity (speed)
    - Time to build the model
    - Time to classify examples using the model
  - Scalability
    - How run time changes with the increase of size of data.
- Here we focus on the predictive performance



Predictive Performance Evaluation

- Objective:
  - Find out how good is a learned *model* (i.e., classifier) in classifying *a test set of examples*?
  - The test set should be different from the training data from which the model is built from
- Performance measures
  - Classification accuracy
  - Classification error rate
  - Classification cost
  - Precision
  - Recall
  - F-measures
  - Area Under ROC Curve (AUC), etc.

Performance Measures: Accuracy and Error Rate

- Accuracy of a classifier on a data set:

$$\frac{\text{number of examples classified correctly}}{\text{total number of examples in the data set}}$$

- Error rate of a classifier (= 1 – accuracy)

$$\frac{\text{number of examples classified incorrectly}}{\text{total number of examples in the data set}}$$

- Accuracy or error on the training data is not a good indicator of the classifier’s performance on future data.

Confusion Matrix

- Confusion matrix is often used to calculate all the metrics. For problems with 2 classes:

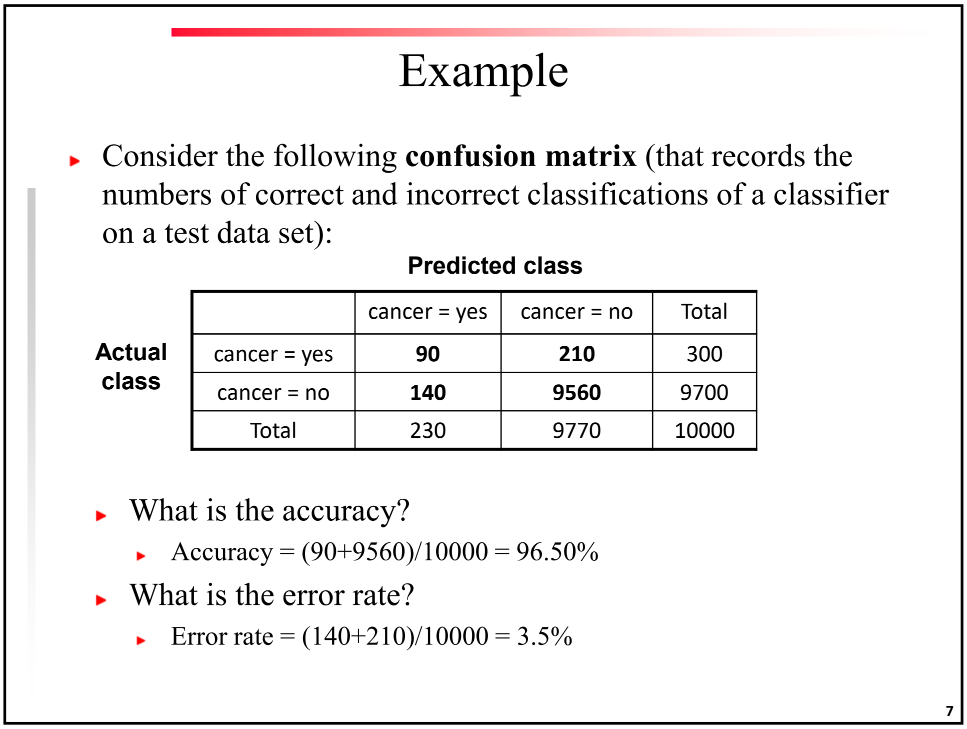
Predicted class

Actual class

	Class = yes	Class = no	Total
Class = yes	**TP**	**FN**	TP+FN
Class = no	**FP**	**TN**	FP+TN
Total	TP+FP	FN+TN	TP+TN+FP+FN

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

- $\text{Error rate} = \dfrac{FP+FN}{TP+TN+FP+FN}$

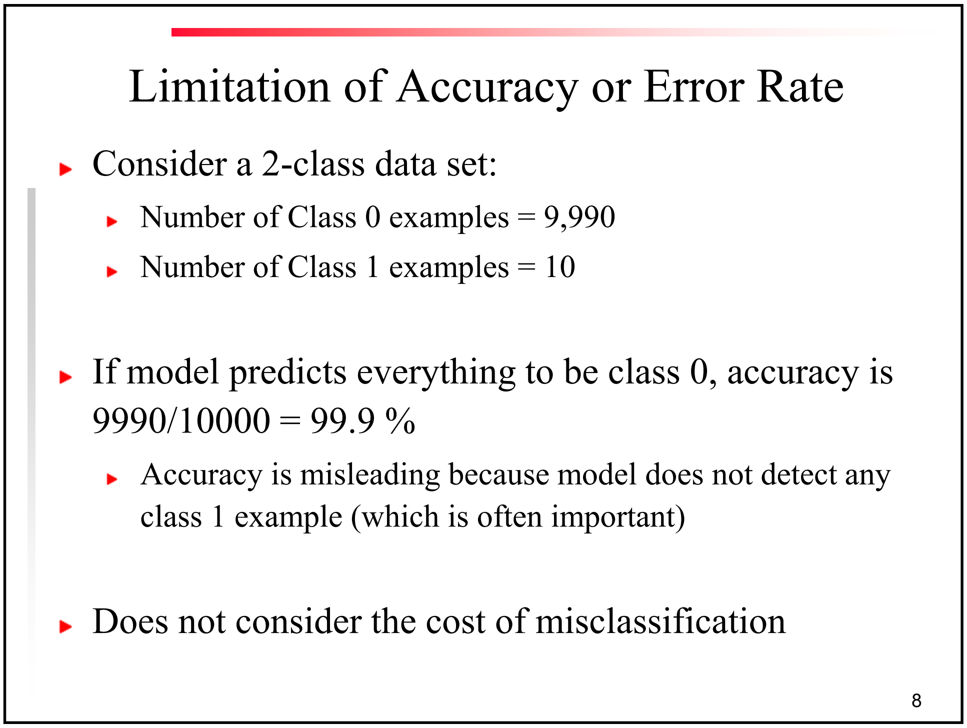


Example

- Consider the following **confusion matrix** (that records the numbers of correct and incorrect classifications of a classifier on a test data set):

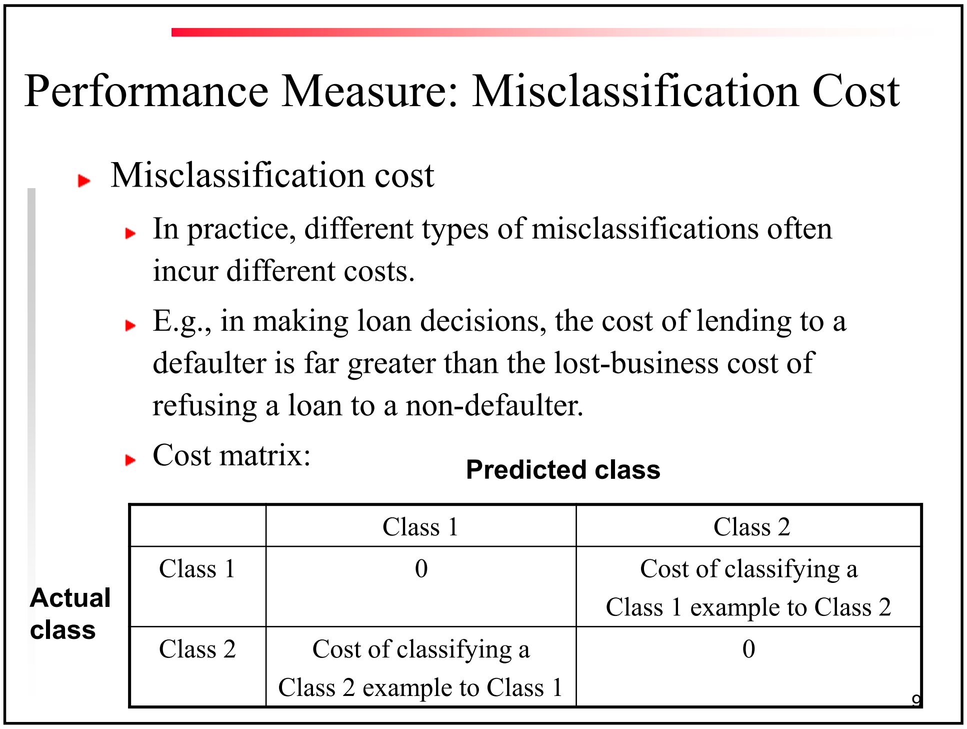
		Predicted class		
		cancer = yes	cancer = no	Total
Actual class	cancer = yes	**90**	**210**	300
	cancer = no	**140**	**9560**	9700
	Total	230	9770	10000

  - What is the accuracy?
    - Accuracy = (90+9560)/10000 = 96.50%
  - What is the error rate?
    - Error rate = (140+210)/10000 = 3.5%



Limitation of Accuracy or Error Rate

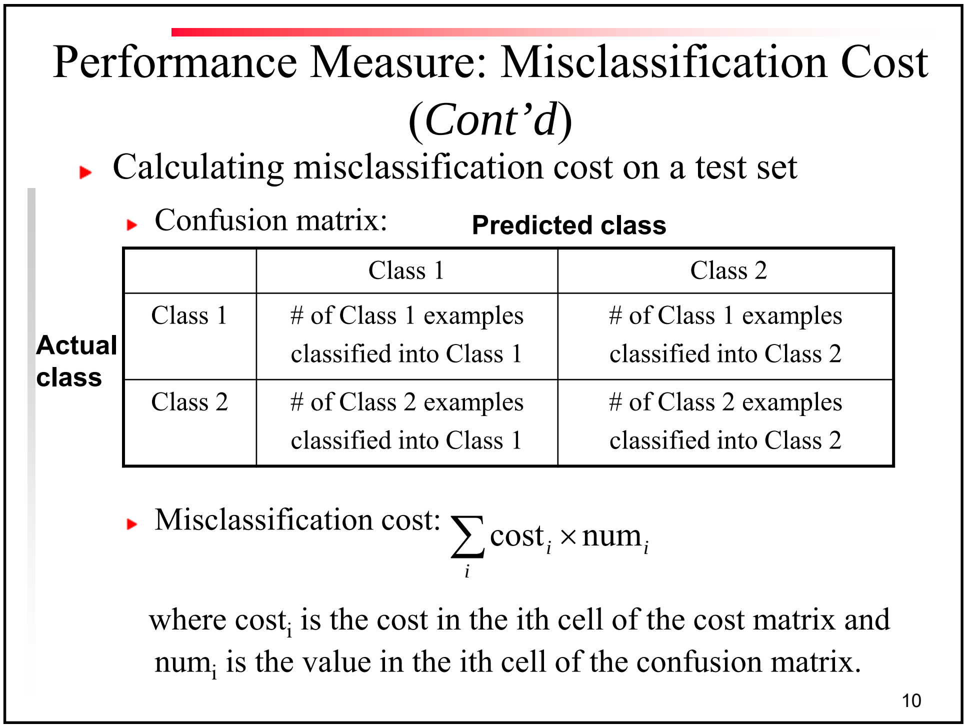
- Consider a 2-class data set:
  - Number of Class 0 examples = 9,990
  - Number of Class 1 examples = 10
- If model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %
  - Accuracy is misleading because model does not detect any class 1 example (which is often important)
- Does not consider the cost of misclassification



Performance Measure: Misclassification Cost

- Misclassification cost
  - In practice, different types of misclassifications often incur different costs.
  - E.g., in making loan decisions, the cost of lending to a defaulter is far greater than the lost-business cost of refusing a loan to a non-defaulter.
  - Cost matrix:

		Predicted class	
		Class 1	Class 2
Actual class	Class 1	0	Cost of classifying a Class 1 example to Class 2
	Class 2	Cost of classifying a Class 2 example to Class 1	0



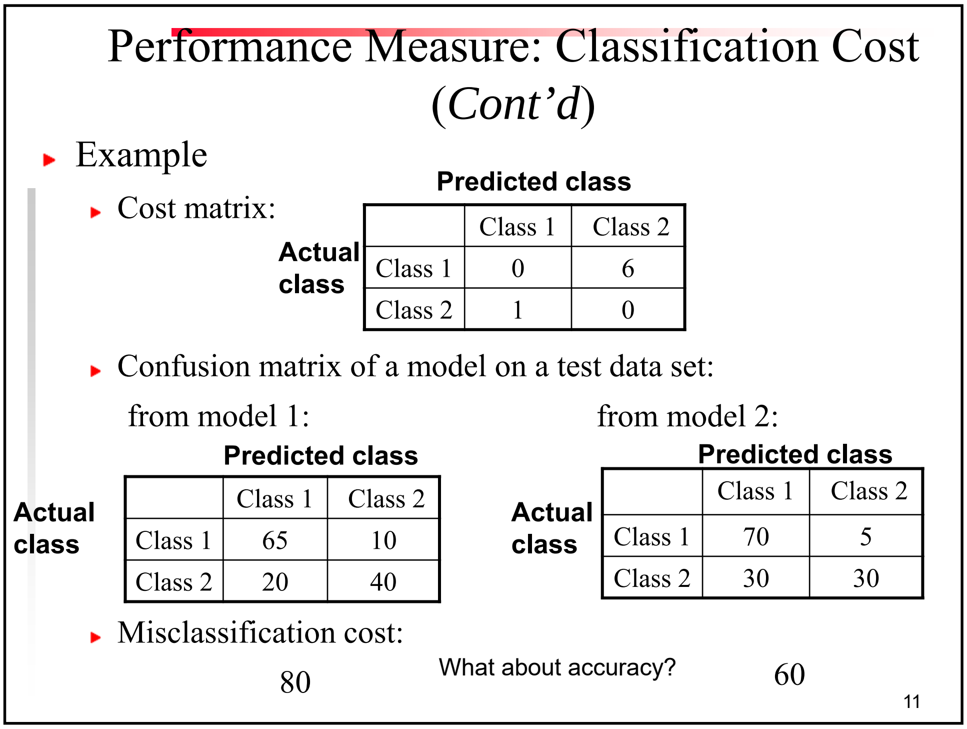
Performance Measure: Misclassification Cost (*Cont'd*)

- Calculating misclassification cost on a test set
  - Confusion matrix:

		Predicted class	
		Class 1	Class 2
Actual class	Class 1	# of Class 1 examples classified into Class 1	# of Class 1 examples classified into Class 2
	Class 2	# of Class 2 examples classified into Class 1	# of Class 2 examples classified into Class 2

  - Misclassification cost: $\sum_i \text{cost}_i \times \text{num}_i$

where cost_i is the cost in the ith cell of the cost matrix and num_i is the value in the ith cell of the confusion matrix.



Performance Measure: Classification Cost (*Cont'd*)

- Example
  - Cost matrix:

Predicted class

Actual class		Class 1	Class 2
	Class 1	0	6
	Class 2	1	0

  - Confusion matrix of a model on a test data set:

from model 1:

Predicted class

Actual class		Class 1	Class 2
	Class 1	65	10
	Class 2	20	40

from model 2:

Predicted class

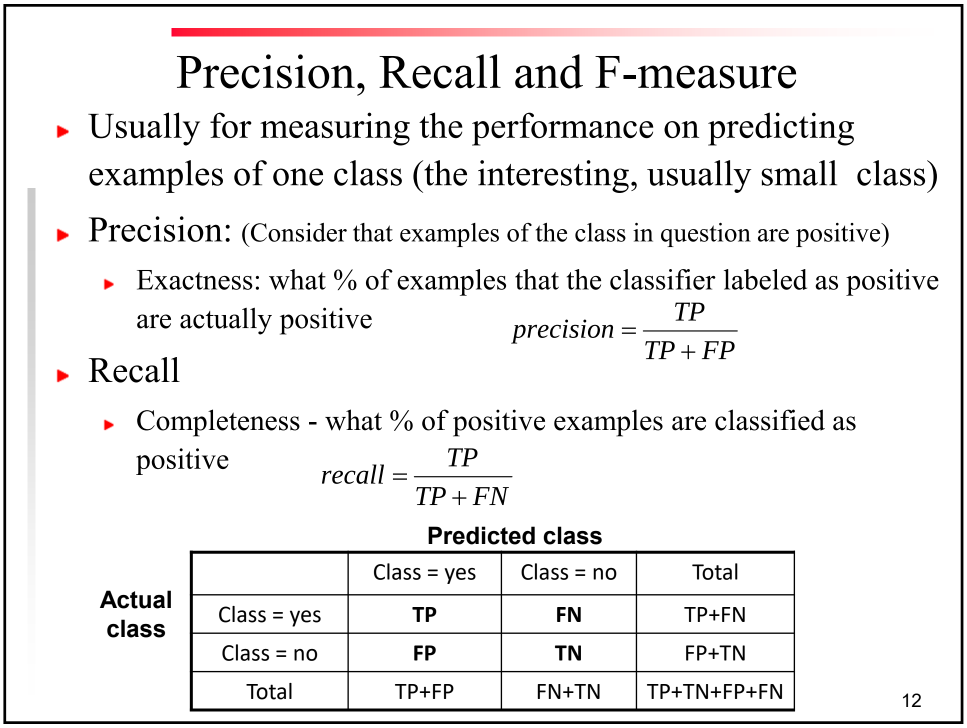
Actual class		Class 1	Class 2
	Class 1	70	5
	Class 2	30	30

  - Misclassification cost:

80

What about accuracy?

60



Precision, Recall and F-measure

- Usually for measuring the performance on predicting examples of one class (the interesting, usually small class)
- Precision: (Consider that examples of the class in question are positive)
  - Exactness: what % of examples that the classifier labeled as positive are actually positive

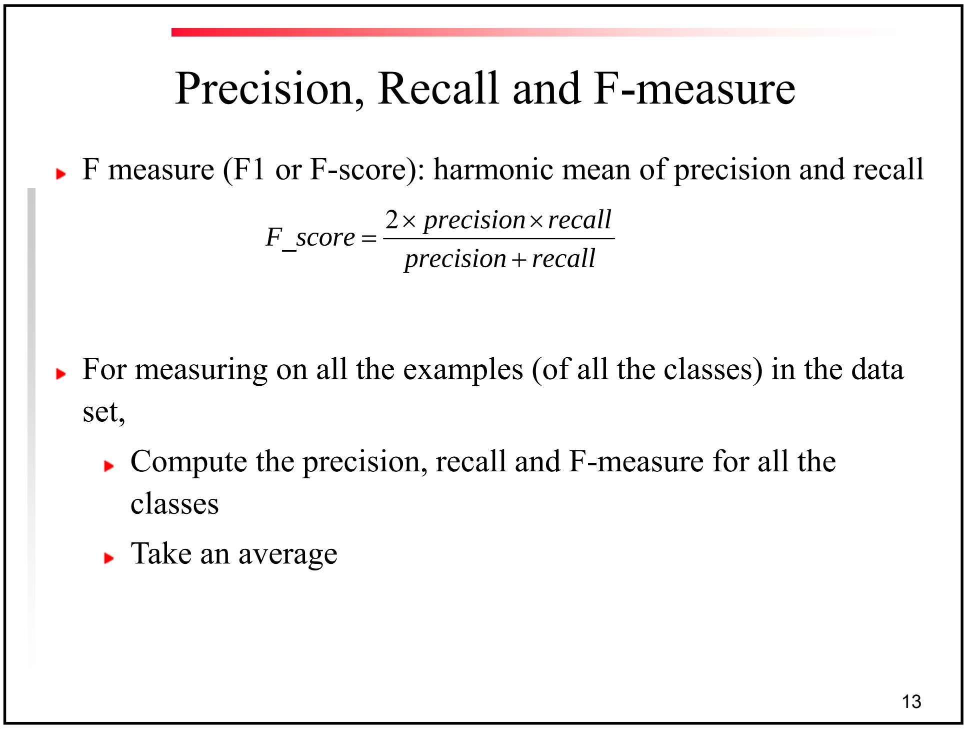
$$precision = \frac{TP}{TP + FP}$$

- Recall
  - Completeness - what % of positive examples are classified as positive

$$recall = \frac{TP}{TP + FN}$$

Predicted class

Actual class		Class = yes	Class = no	Total
	Class = yes	**TP**	**FN**	TP+FN
	Class = no	**FP**	**TN**	FP+TN
	Total	TP+FP	FN+TN	TP+TN+FP+FN

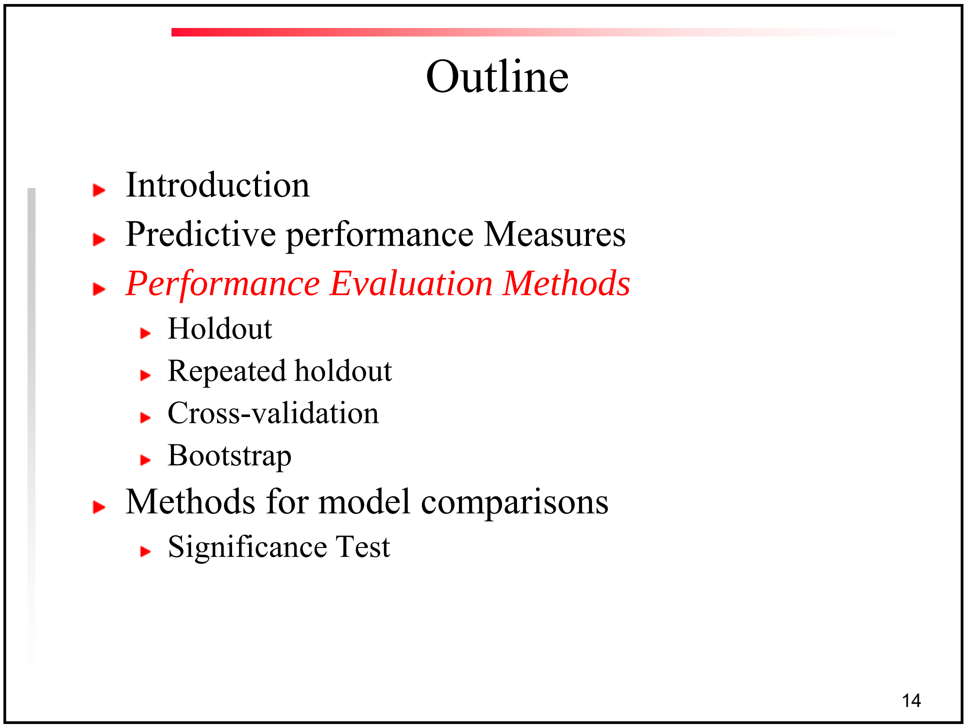


Precision, Recall and F-measure

- F measure (F1 or F-score): harmonic mean of precision and recall

$$F\_score = \frac{2 \times precision \times recall}{precision + recall}$$

- For measuring on all the examples (of all the classes) in the data set,
  - Compute the precision, recall and F-measure for all the classes
  - Take an average



Outline

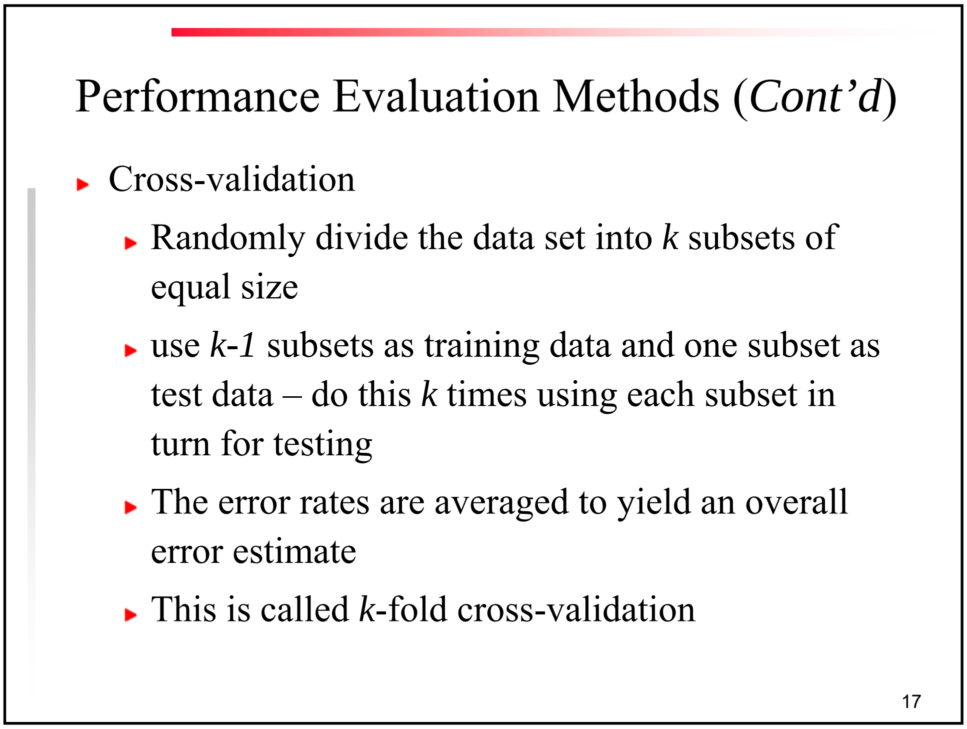
- Introduction
- Predictive performance Measures
- *Performance Evaluation Methods*
  - Holdout
  - Repeated holdout
  - Cross-validation
  - Bootstrap
- Methods for model comparisons
  - Significance Test

Performance Evaluation Methods

- Problem:
  - Given a set *S* of data and a classification learning algorithm *A*, how do we evaluate the predictive performance of *A* on *S*?
  - We cannot learn a model from *S* and evaluate the model on *S* again because
    - Error on the training data is not a good indicator of a classifier's performance on future data.
- Evaluation methods
  - Hold-out estimation
  - Repeated hold-out estimation
  - Cross validation
  - Bootstrap

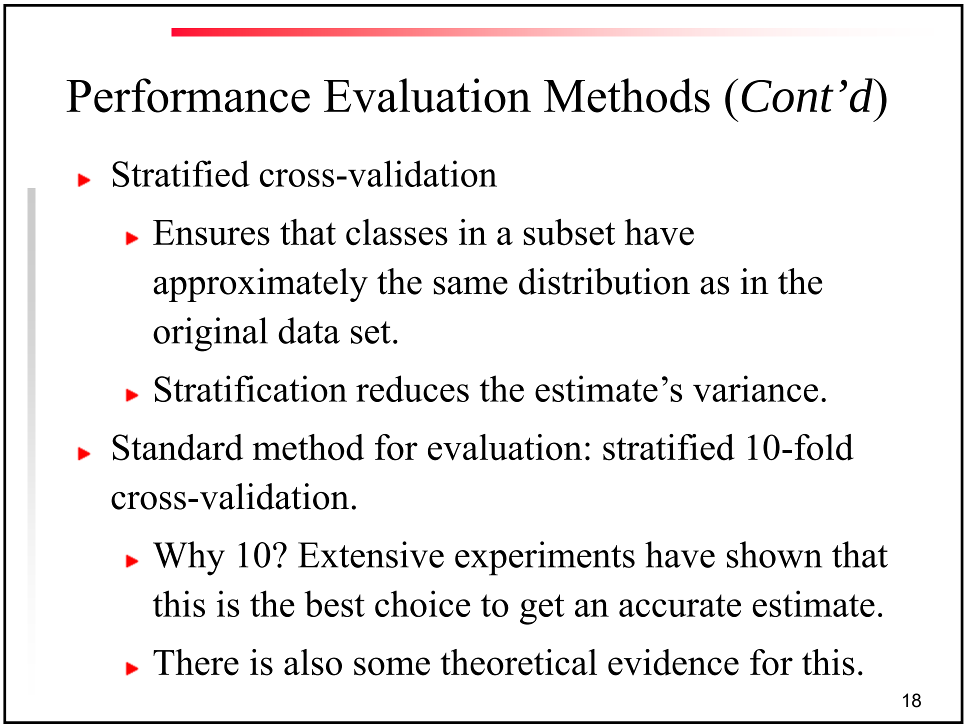
Performance Evaluation Methods

- Hold-out estimation
  - Randomly split the data set into a training set and a test set, e.g., training set (2/3), test set(1/3)
  - Build a model from the training set and estimate the error (or another measure, e.g., cost) on the test set
- Repeated hold-out estimation
  - Holdout estimate can be made more reliable by repeating the process with different random splits
  - For each split, an error rate is collected.
  - An overall error rate is obtained by averaging the error rates on the different splits



Performance Evaluation Methods (*Cont'd*)

- Cross-validation
  - Randomly divide the data set into *k* subsets of equal size
  - use *k-1* subsets as training data and one subset as test data – do this *k* times using each subset in turn for testing
  - The error rates are averaged to yield an overall error estimate
  - This is called *k*-fold cross-validation



Performance Evaluation Methods (*Cont'd*)

- Stratified cross-validation
  - Ensures that classes in a subset have approximately the same distribution as in the original data set.
  - Stratification reduces the estimate's variance.
- Standard method for evaluation: stratified 10-fold cross-validation.
  - Why 10? Extensive experiments have shown that this is the best choice to get an accurate estimate.
  - There is also some theoretical evidence for this.

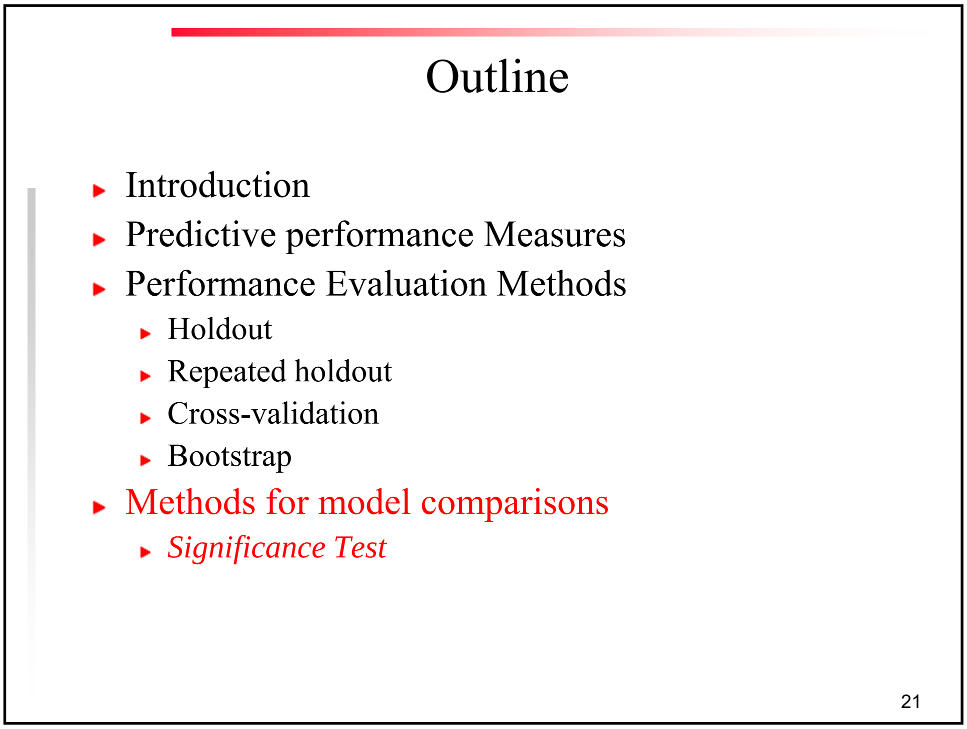
Performance Evaluation Methods (*Cont'd*)

- Leave-one-out
  - A special case of *k*-fold cross-validation method when *k* equals to the number *N* of examples in the data set
  - In each iteration, the test data set contains only one example and the training data set contains the rest *N*-1 examples.
  - Advantages:
    - The greatest possible amount of data is used for training in each iteration.
    - Deterministic: no random sampling is involved
  - Disadvantages:
    - Long running time
    - Non-stratified partitions of the data set

Evaluating Classifier Accuracy: Bootstrap

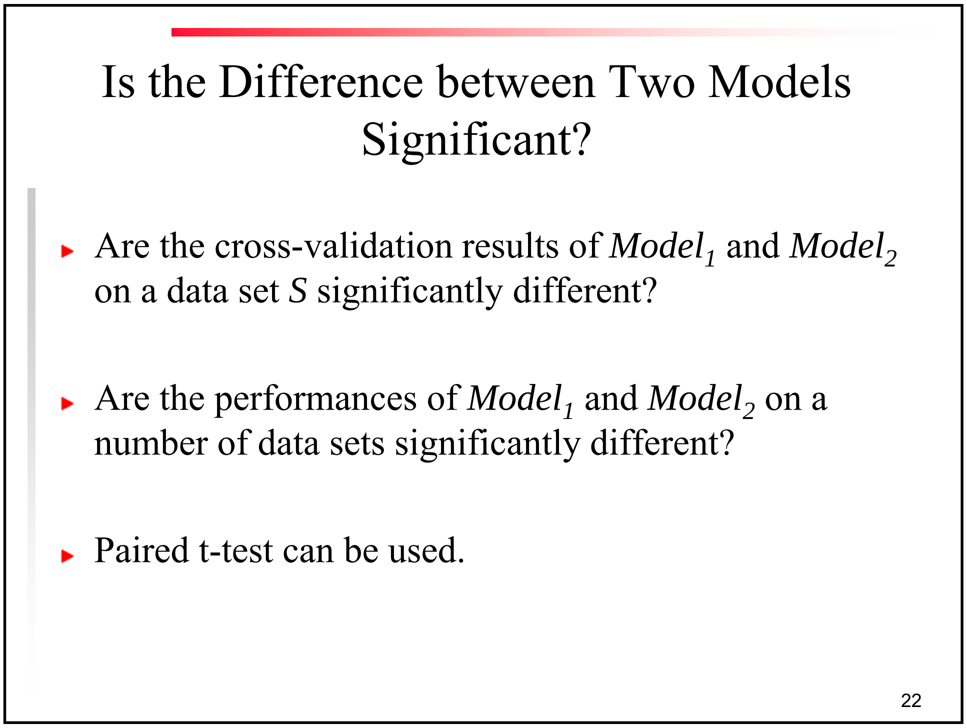
- Bootstrap
  - Works well with small data sets
  - Samples the given training examples randomly *with replacement*
    - i.e., each time an example is selected, it is equally likely to be selected again and re-added to the training set
- Several bootstrap methods, and a common one is **.632 boostrap**
  - A data set with *d* examples is sampled *d* times, with replacement, resulting in a training set of *d* examples. The examples that did not make it into the training set end up forming the test set. About 63.2% of the original data end up in the bootstrap, and the remaining 36.8% form the test set (since $(1 – 1/d)^d \approx e^{-1} = 0.368$)
  - Repeat the sampling procedure *k* times, overall accuracy of the model:

$$Acc(M) = \frac{1}{k}\sum_{i=1}^{k}(0.632 \times Acc(M_i)_{test\_set} + 0.368 \times Acc(M_i)_{train\_set})$$



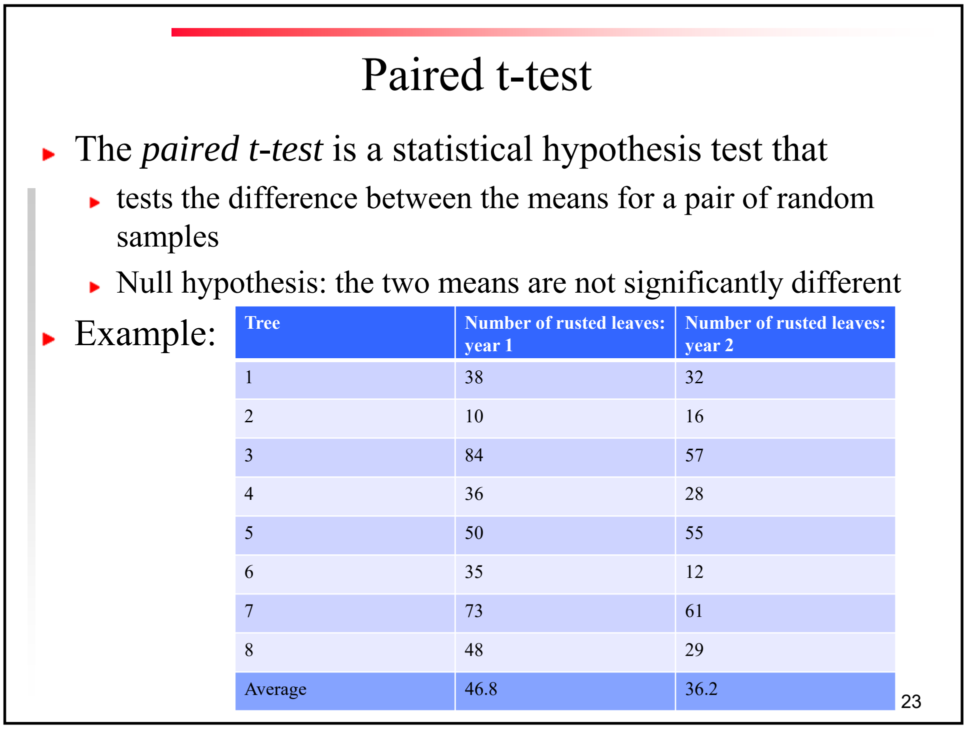
Outline

- Introduction
- Predictive performance Measures
- Performance Evaluation Methods
  - Holdout
  - Repeated holdout
  - Cross-validation
  - Bootstrap
- Methods for model comparisons
  - *Significance Test*



Is the Difference between Two Models Significant?

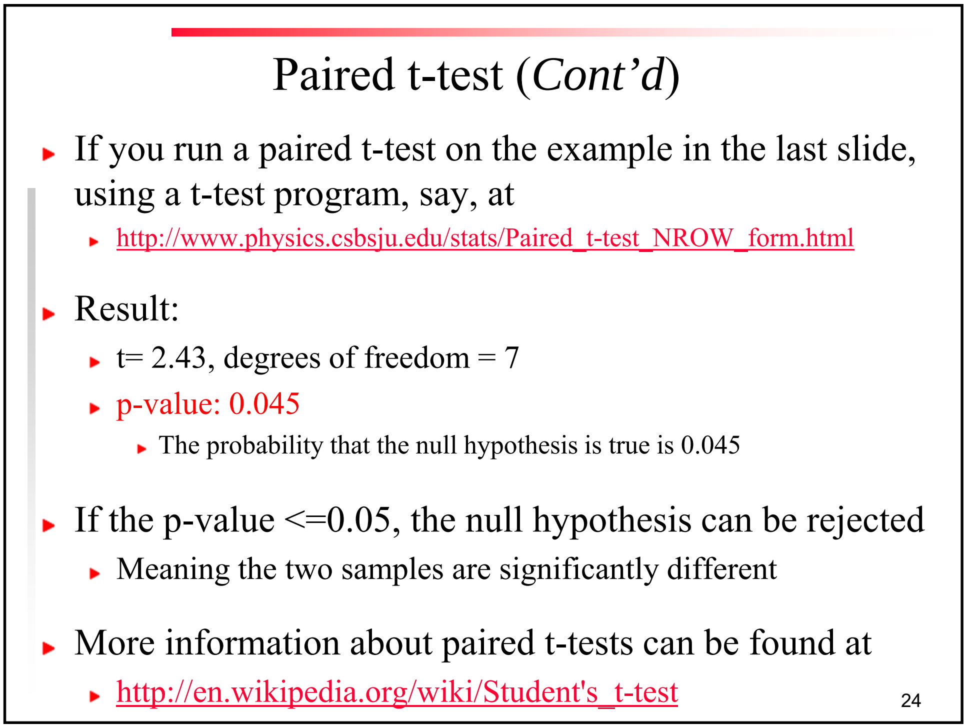
- Are the cross-validation results of $Model_1$ and $Model_2$ on a data set S significantly different?
- Are the performances of $Model_1$ and $Model_2$ on a number of data sets significantly different?
- Paired t-test can be used.



Paired t-test

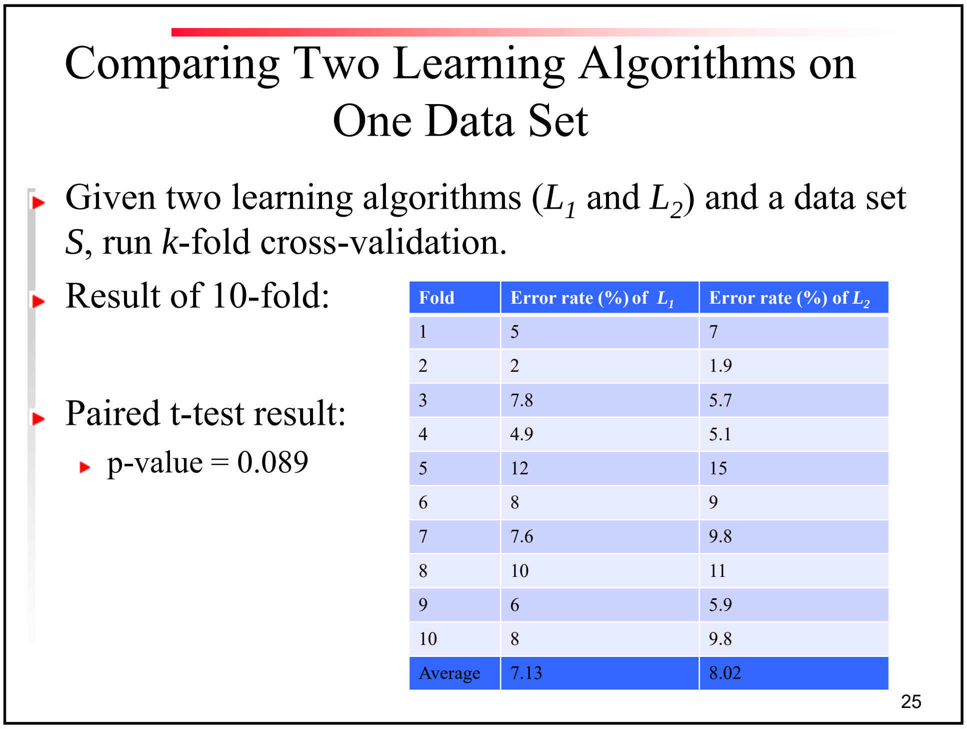
- The *paired t-test* is a statistical hypothesis test that
  - tests the difference between the means for a pair of random samples
  - Null hypothesis: the two means are not significantly different
- Example:

Tree	Number of rusted leaves: year 1	Number of rusted leaves: year 2
1	38	32
2	10	16
3	84	57
4	36	28
5	50	55
6	35	12
7	73	61
8	48	29
Average	46.8	36.2



Paired t-test (*Cont'd*)

- If you run a paired t-test on the example in the last slide, using a t-test program, say, at
  - http://www.physics.csbsju.edu/stats/Paired_t-test_NROW_form.html
- Result:
  - t= 2.43, degrees of freedom = 7
  - p-value: 0.045
    - The probability that the null hypothesis is true is 0.045
- If the p-value <=0.05, the null hypothesis can be rejected
  - Meaning the two samples are significantly different
- More information about paired t-tests can be found at
  - http://en.wikipedia.org/wiki/Student's_t-test

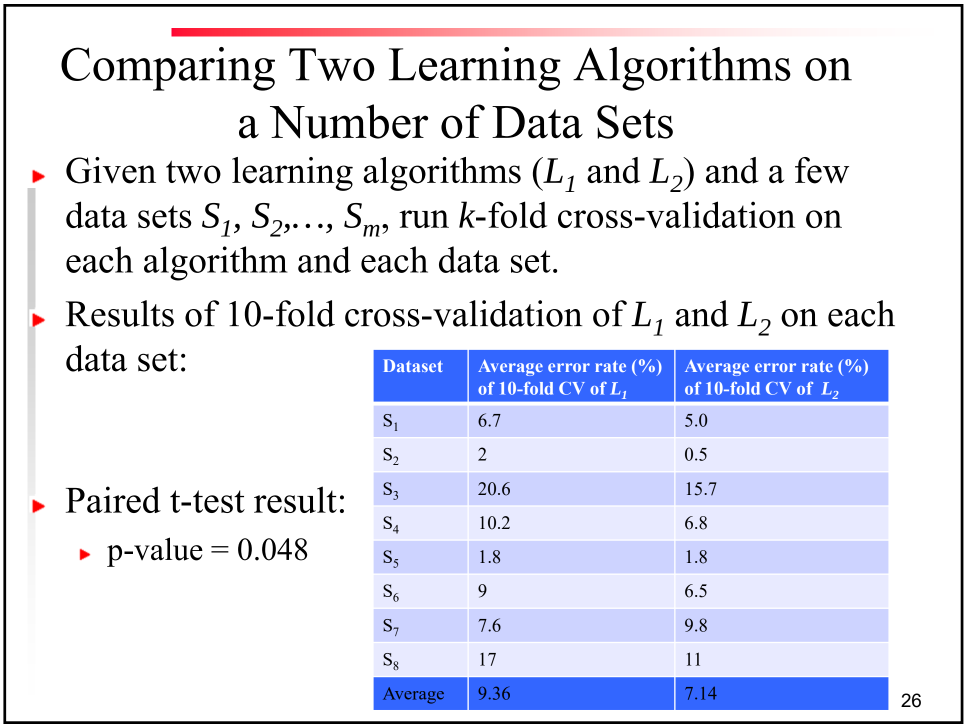


Comparing Two Learning Algorithms on One Data Set

- Given two learning algorithms (L_1 and L_2) and a data set S, run k-fold cross-validation.
- Result of 10-fold:

Fold	Error rate (%) of L_1	Error rate (%) of L_2
1	5	7
2	2	1.9
3	7.8	5.7
4	4.9	5.1
5	12	15
6	8	9
7	7.6	9.8
8	10	11
9	6	5.9
10	8	9.8
Average	7.13	8.02

- Paired t-test result:
  - p-value = 0.089



Comparing Two Learning Algorithms on a Number of Data Sets

- Given two learning algorithms (L_1 and L_2) and a few data sets S_1, S_2,…, S_m, run k-fold cross-validation on each algorithm and each data set.
- Results of 10-fold cross-validation of L_1 and L_2 on each data set:

Dataset	Average error rate (%) of 10-fold CV of L_1	Average error rate (%) of 10-fold CV of L_2
S_1	6.7	5.0
S_2	2	0.5
S_3	20.6	15.7
S_4	10.2	6.8
S_5	1.8	1.8
S_6	9	6.5
S_7	7.6	9.8
S_8	17	11
Average	9.36	7.14

- Paired t-test result:
  - p-value = 0.048