

# Machine Learning

## Chapter 5. Credibility

# 5

## Credibility:

Evaluating what's been learned

- ❖ Issues: training, testing, tuning
- ❖ Predicting performance: confidence limits
- ❖ Holdout, cross-validation, bootstrap
- ❖ Comparing schemes: the t-test
- ❖ Predicting probabilities: loss functions
- ❖ Cost-sensitive measures
- ❖ Evaluating numeric prediction
- ❖ The Minimum Description Length principle



# Evaluation: the key to success

- ❖ How predictive is the model we learned?
- ❖ Error on the training data is *not* a good indicator of performance on future data
  - ❑ Otherwise 1-NN would be the optimum classifier!
- ❖ Simple solution that can be used if lots of (labeled) data is available:
  - ❑ Split data into training and test set
- ❖ However: (labeled) data is usually limited
  - ❑ More sophisticated techniques need to be used

# Issues in evaluation

- ❖ Statistical reliability of estimated differences in performance (→ significance tests)
- ❖ Choice of performance measure:
  - ❑ Number of correct classifications
  - ❑ Accuracy of probability estimates
  - ❑ Error in numeric predictions
- ❖ Costs assigned to different types of errors
  - ❑ Many practical applications involve costs

# Training and testing I

- ❖ Natural performance measure for classification problems: *error rate*
  - ❑ *Success*: instance's class is predicted correctly
  - ❑ *Error*: instance's class is predicted incorrectly
  - ❑ Error rate: proportion of errors made over the whole set of instances
- ❖ *Resubstitution error*: error rate obtained from training data
- ❖ Resubstitution error is (hopelessly) optimistic!

# Training and testing II

- ❖ *Test set*: independent instances that have played no part in formation of classifier
  - Assumption: both training data and test data are representative samples of the underlying problem
- ❖ Test and training data may differ in nature
  - Example: classifiers built using customer data from two different towns  $A$  and  $B$ 
    - To estimate performance of classifier from town  $A$  in completely new town, test it on data from  $B$

# Note on parameter tuning

- ❖ It is important that the test data is not used *in any way* to create the classifier
- ❖ Some learning schemes operate in two stages:
  - ❑ Stage 1: build the basic structure
  - ❑ Stage 2: optimize parameter settings
- ❖ The test data can't be used for parameter tuning!
- ❖ Proper procedure uses *three* sets: *training data*, *validation data*, and *test data*
  - ❑ Validation data is used to optimize parameters

# Making the most of the data

- ❖ Once evaluation is complete, *all the data* can be used to build the final classifier
- ❖ Generally, the larger the training data the better the classifier (but returns diminish)
- ❖ The larger the test data the more accurate the error estimate
- ❖ *Holdout* procedure: method of splitting original data into training and test set
  - ❑ Dilemma: ideally both training set *and* test set should be large!





# Predicting performance

- ❖ Assume the estimated error rate is 25%. How close is this to the true error rate?
  - ❑ Depends on the amount of test data
- ❖ Prediction is just like tossing a (biased!) coin
  - ❑ “Head” is a “success”, “tail” is an “error”
- ❖ In statistics, a succession of independent events like this is called a *Bernoulli process*
  - ❑ Statistical theory provides us with confidence intervals for the true underlying proportion

# Confidence intervals

- ❖ We can say:  $p$  lies within a certain specified interval with a certain specified confidence
- ❖ Example:  $S=750$  successes in  $N=1000$  trials
  - ❑ Estimated success rate: 75%
  - ❑ How close is this to true success rate  $p$ ?
    - Answer: with 80% confidence  $p \in [73.2, 76.7]$
- ❖ Another example:  $S=75$  and  $N=100$ 
  - ❑ Estimated success rate: 75%
  - ❑ With 80% confidence  $p \in [69.1, 80.1]$

# Mean and variance

- ❖ Mean and variance for a Bernoulli trial:  
 $p, p(1-p)$
- ❖ Expected success rate  $f=S/N$
- ❖ Mean and variance for  $f$ :  $p, p(1-p)/N$
- ❖ For large enough  $N$ ,  $f$  follows a Normal distribution
- ❖  $c\%$  confidence interval  $[-z \leq X \leq z]$  for random variable with 0 mean is given by:

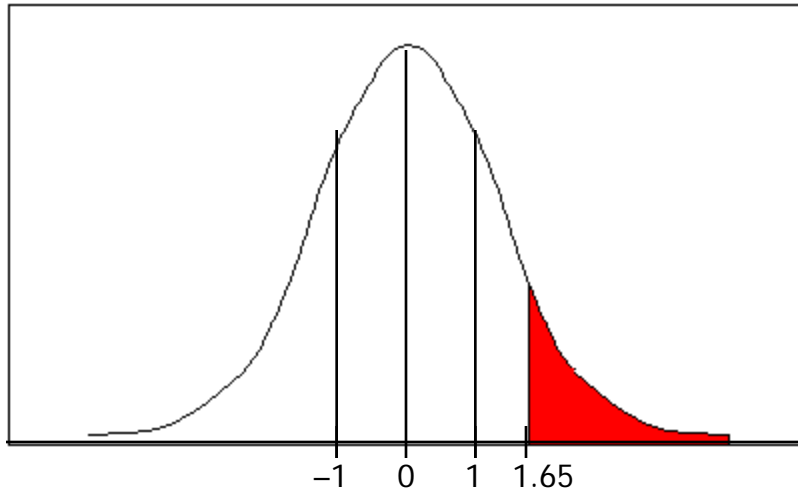
$$\Pr[-z \leq X \leq z] = c$$

- ❖ With a symmetric distribution:

$$\Pr[-z \leq X \leq z] = 1 - 2 \times \Pr[X \geq z]$$

# Confidence limits

- ❖ Confidence limits for the normal distribution with 0 mean and a variance of 1:



$\Pr[X \geq z]$	$z$
0.1%	3.09
0.5%	2.58
1%	2.33
5%	1.65
10%	1.28
20%	0.84
40%	0.25

- ❖ Thus:

$$\Pr[-1.65 \leq X \leq 1.65] = 90\%$$

- ❖ To use this we have to reduce our random variable  $f$  to have 0 mean and unit variance

# Transforming $f$

- ❖ Transformed value for  $f$ :  $\frac{f - p}{\sqrt{p(1-p)/N}}$   
(i.e. subtract the mean and divide by the *standard deviation*)

- ❖ Resulting equation:

$$\Pr\left[-z \leq \frac{f - p}{\sqrt{p(1-p)/N}} \leq z\right] = c$$

- ❖ Solving for  $p$ :

$$p = \left( f + \frac{z^2}{2N} \pm z \sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}} \right) / \left( 1 + \frac{z^2}{N} \right)$$

# Examples

- ❖  $f = 75\%$ ,  $N = 1000$ ,  $c = 80\%$  (so that  $z = 1.28$ ):

$$p \in [0.732, 0.767]$$

- ❖  $f = 75\%$ ,  $N = 100$ ,  $c = 80\%$  (so that  $z = 1.28$ ):

$$p \in [0.691, 0.801]$$

- ❖ Note that normal distribution assumption is only valid for large  $N$  (i.e.  $N > 100$ )

- ❖  $f = 75\%$ ,  $N = 10$ ,  $c = 80\%$  (so that  $z = 1.28$ ):

$$p \in [0.549, 0.881]$$

(should be taken with a grain of salt)



# Holdout estimation

- ❖ What to do if the amount of data is limited?
- ❖ The *holdout* method reserves a certain amount for testing and uses the remainder for training
  - ❑ Usually: one third for testing, the rest for training
- ❖ Problem: the samples might not be representative
  - ❑ Example: class might be missing in the test data
- ❖ Advanced version uses *stratification*
  - ❑ Ensures that each class is represented with approximately equal proportions in both subsets

# Repeated holdout method

- ❖ Holdout estimate can be made more reliable by repeating the process with different subsamples
  - ❑ In each iteration, a certain proportion is randomly selected for training (possibly with stratification)
  - ❑ The error rates on the different iterations are averaged to yield an overall error rate
- ❖ This is called the *repeated holdout* method
- ❖ Still not optimum: the different test sets overlap
  - ❑ Can we prevent overlapping?



# Cross-validation

- ❖ *Cross-validation* avoids overlapping test sets
  - ❑ First step: split data into  $k$  subsets of equal size
  - ❑ Second step: use each subset in turn for testing, the remainder for training
- ❖ Called *k-fold cross-validation*
- ❖ Often the subsets are stratified before the cross-validation is performed
- ❖ The error estimates are averaged to yield an overall error estimate

# More on cross-validation

- ❖ Standard method for evaluation: stratified ten-fold cross-validation
- ❖ Why ten?
  - ❑ Extensive experiments have shown that this is the best choice to get an accurate estimate
  - ❑ There is also some theoretical evidence for this
- ❖ Stratification reduces the estimate's variance
- ❖ Even better: repeated stratified cross-validation
  - ❑ E.g. ten-fold cross-validation is repeated ten times and results are averaged (reduces the variance)

# Leave-One-Out cross-validation

- ❖ Leave-One-Out:  
a particular form of cross-validation:
  - ❑ Set number of folds to number of training instances
  - ❑ I.e., for  $n$  training instances, build classifier  $n$  times
- ❖ Makes best use of the data
- ❖ Involves no random subsampling
- ❖ Very computationally expensive
  - ❑ (exception: NN)

# Leave-One-Out-CV and stratification

- ❖ Disadvantage of Leave-One-Out-CV: stratification is not possible
  - ❑ It *guarantees* a non-stratified sample because there is only one instance in the test set!
- ❖ Extreme example: random dataset split equally into two classes
  - ❑ Best inducer predicts majority class
  - ❑ 50% accuracy on fresh data
  - ❑ Leave-One-Out-CV estimate is 100% error!

# The bootstrap

- ❖ CV uses sampling *without replacement*
  - ❑ The same instance, once selected, can not be selected again for a particular training/test set
- ❖ The *bootstrap* uses sampling *with replacement* to form the training set
  - ❑ Sample a dataset of  $n$  instances  $n$  times *with replacement* to form a new dataset of  $n$  instances
  - ❑ Use this data as the training set
  - ❑ Use the instances from the original dataset that don't occur in the new training set for testing



# The 0.632 bootstrap

## ❖ Also called the *0.632 bootstrap*

- ❑ A particular instance has a probability of  $1-1/n$  of *not* being picked
- ❑ Thus its probability of ending up in the test data is:

$$\left(1 - \frac{1}{n}\right)^n \approx e^{-1} = 0.368$$

- ❑ This means the training data will contain approximately 63.2% of the instances

# Estimating error with the bootstrap

- ❖ The error estimate on the test data will be very pessimistic

- Trained on just ~63% of the instances

- ❖ Therefore, combine it with the resubstitution error:

$$err = 0.632 \cdot e_{\text{test instances}} + 0.368 \cdot e_{\text{training instances}}$$

- ❖ The resubstitution error gets less weight than the error on the test data
- ❖ Repeat process several times with different replacement samples; average the results

# More on the bootstrap

- ❖ Probably the best way of estimating performance for very small datasets
- ❖ However, it has some problems
  - ❑ Consider the random dataset from above
  - ❑ A perfect memorizer will achieve 0% resubstitution error and ~50% error on test data
  - ❑ Bootstrap estimate for this classifier:
$$err = 0.632 \cdot 50\% + 0.368 \cdot 0\% = 31.6\%$$
  - ❑ True expected error: 50%





# Comparing data mining schemes

- ❖ Frequent question: which of two learning schemes performs better?
- ❖ Note: this is domain dependent!
- ❖ Obvious way: compare 10-fold CV estimates
- ❖ Problem: variance in estimate
- ❖ Variance can be reduced using repeated CV
- ❖ However, we still don't know whether the results are reliable

# Significance tests

- ❖ Significance tests tell us how confident we can be that there really is a difference
- ❖ *Null hypothesis*: there is no “real” difference
- ❖ *Alternative hypothesis*: there is a difference
- ❖ A significance test measures how much evidence there is in favor of rejecting the null hypothesis
- ❖ Let's say we are using 10-fold CV
- ❖ Question: do the two means of the 10 CV estimates differ significantly?

# Paired t-test

- ❖ *Student's t-test* tells whether the means of two samples are significantly different
- ❖ Take individual samples using cross-validation
- ❖ Use a *paired* t-test because the individual samples are paired
  - ❑ The same CV is applied twice

## William Gosset

**Born:** 1876 in Canterbury; **Died:** 1937 in Beaconsfield, England  
**Obtained a post as a chemist in the Guinness brewery in Dublin in 1899. Invented the t-test to handle small samples for quality control in brewing. Wrote under the name "Student".**



# Distribution of the means

- ❖  $x_1 x_2 \dots x_k$  and  $y_1 y_2 \dots y_k$  are the  $2k$  samples for a  $k$ -fold CV
- ❖  $m_x$  and  $m_y$  are the means
- ❖ With enough samples, the mean of a set of independent samples is normally distributed
- ❖ Estimated variances of the means are  $\sigma_x^2/k$  and  $\sigma_y^2/k$
- ❖ If  $\mu_x$  and  $\mu_y$  are the true means then  $\frac{m_x - \mu_x}{\sqrt{\sigma_x^2/k}}$   $\frac{m_y - \mu_y}{\sqrt{\sigma_y^2/k}}$  are *approximately* normally distributed with mean 0, variance 1

# Student's distribution

- ❖ With small samples ( $k < 100$ ) the mean follows *Student's distribution with  $k-1$  degrees of freedom*

- ❖ Confidence limits:

**9 degrees of freedom**

$\Pr[X \geq z]$	$z$
0.1%	4.30
0.5%	3.25
1%	2.82
5%	1.83
10%	1.38
20%	0.88

**normal distribution**

$\Pr[X \geq z]$	$z$
0.1%	3.09
0.5%	2.58
1%	2.33
5%	1.65
10%	1.28
20%	0.84

# Distribution of the differences

- ❖ Let  $m_d = m_x - m_y$
- ❖ The difference of the means ( $m_d$ ) also has a Student's distribution with  $k-1$  degrees of freedom
- ❖ Let  $\sigma_d^2$  be the variance of the difference
- ❖ The standardized version of  $m_d$  is called the  $t$ -statistic:

$$t = \frac{m_d}{\sqrt{\sigma_d^2 / k}}$$

- ❖ We use  $t$  to perform the  $t$ -test

# Performing the test

- Fix a significance level  $\alpha$ 
  - If a difference is significant at the  $\alpha\%$  level, there is a  $(100-\alpha)\%$  chance that there really is a difference
- Divide the significance level by two because the test is two-tailed
  - I.e. the true difference can be +ve or -ve
- Look up the value for  $z$  that corresponds to  $\alpha/2$
- If  $t \leq -z$  or  $t \geq z$  then the difference is significant
  - I.e. the null hypothesis can be rejected

# Unpaired observations

- ❖ If the CV estimates are from different randomizations, they are no longer paired
- ❖ (or maybe we used  $k$ -fold CV for one scheme, and  $j$ -fold CV for the other one)
- ❖ Then we have to use an *un* paired t-test with  $\min(k, j) - 1$  degrees of freedom
- ❖ The  $t$ -statistic becomes:

$$t = \frac{m_d}{\sqrt{\sigma_d^2 / k}} \quad \Rightarrow \quad t = \frac{m_x - m_y}{\sqrt{\frac{\sigma_x^2}{k} + \frac{\sigma_y^2}{j}}}$$



# Interpreting the result

- ❖ All our cross-validation estimates are based on the same dataset
- ❖ Samples are not independent
- ❖ Should really use a different dataset sample for each of the  $k$  estimates used in the test to judge performance across different training sets
- ❖ Or, use heuristic test, e.g. *corrected resampled t-test*



# Predicting probabilities

- ❖ Performance measure so far: success rate
- ❖ Also called *0-1 loss function*:

$$\sum_i \begin{cases} 0 & \text{if prediction is correct} \\ 1 & \text{if prediction is incorrect} \end{cases}$$

- ❖ Most classifiers produces class probabilities
- ❖ Depending on the application, we might want to check the accuracy of the probability estimates
- ❖ 0-1 loss is not the right thing to use in those cases

# Quadratic loss function

❖  $p_1 \dots p_k$  are probability estimates for an instance

❖  $c$  is the index of the instance's actual class

❖  $a_1 \dots a_k = 0$ , except for  $a_c$  which is 1

❖ *Quadratic loss* is: 
$$\sum_j (p_j - a_j)^2 = \sum_{j \neq c} p_j^2 + (1 - p_c)^2$$

❖ Want to minimize 
$$E \left[ \sum_j (p_j - a_j)^2 \right]$$

❖ Can show that this is minimized when  $p_j = p_j^*$ , the true probabilities

# Informational loss function

- ❖ The informational loss function is  $-\log(p_c)$ , where  $c$  is the index of the instance's actual class
- ❖ Number of bits required to communicate the actual class
- ❖ Let  $p_1^* \dots p_k^*$  be the true class probabilities
- ❖ Then the expected value for the loss function is:

$$-p_1^* \log_2 p_1 - \dots - p_k^* \log_2 p_k$$

- ❖ Justification: minimized when  $p_j = p_j^*$
- ❖ Difficulty: *zero-frequency problem*

# Discussion

## ❖ Which loss function to choose?

- ❑ Both encourage honesty
- ❑ Quadratic loss function takes into account all class probability estimates for an instance
- ❑ Informational loss focuses only on the probability estimate for the actual class
- ❑ Quadratic loss is bounded:  
*it can never exceed 2*  $1 + \sum_j p_j^2$
- ❑ Informational loss can be infinite

## ❖ Informational loss is related to *MDL principle* [later]



# Counting the cost

- ❖ In practice, different types of classification errors often incur different costs
- ❖ Examples:
  - ☐ Terrorist profiling
    - “Not a terrorist” correct 99.99% of the time
  - ☐ Loan decisions
  - ☐ Oil-slick detection
  - ☐ Fault diagnosis
  - ☐ Promotional mailing

# Counting the cost

❖ The *confusion matrix*:

		Predicted class	
		Yes	No
Actual class	Yes	True positive	False negative
	No	False positive	True negative

❖ There many other types of cost!

- ❑ E.g.: cost of collecting training data

# Lift charts

- ❖ In practice, costs are rarely known
- ❖ Decisions are usually made by comparing possible scenarios
- ❖ Example: promotional mailout to 1,000,000 households
  - Mail to all; 0.1% respond (1000)
  - Data mining tool identifies subset of 100,000 most promising, 0.4% of these respond (400)  
*40% of responses for 10% of cost may pay off*
  - Identify subset of 400,000 most promising, 0.2% respond (800)
- ❖ A *lift chart* allows a visual comparison



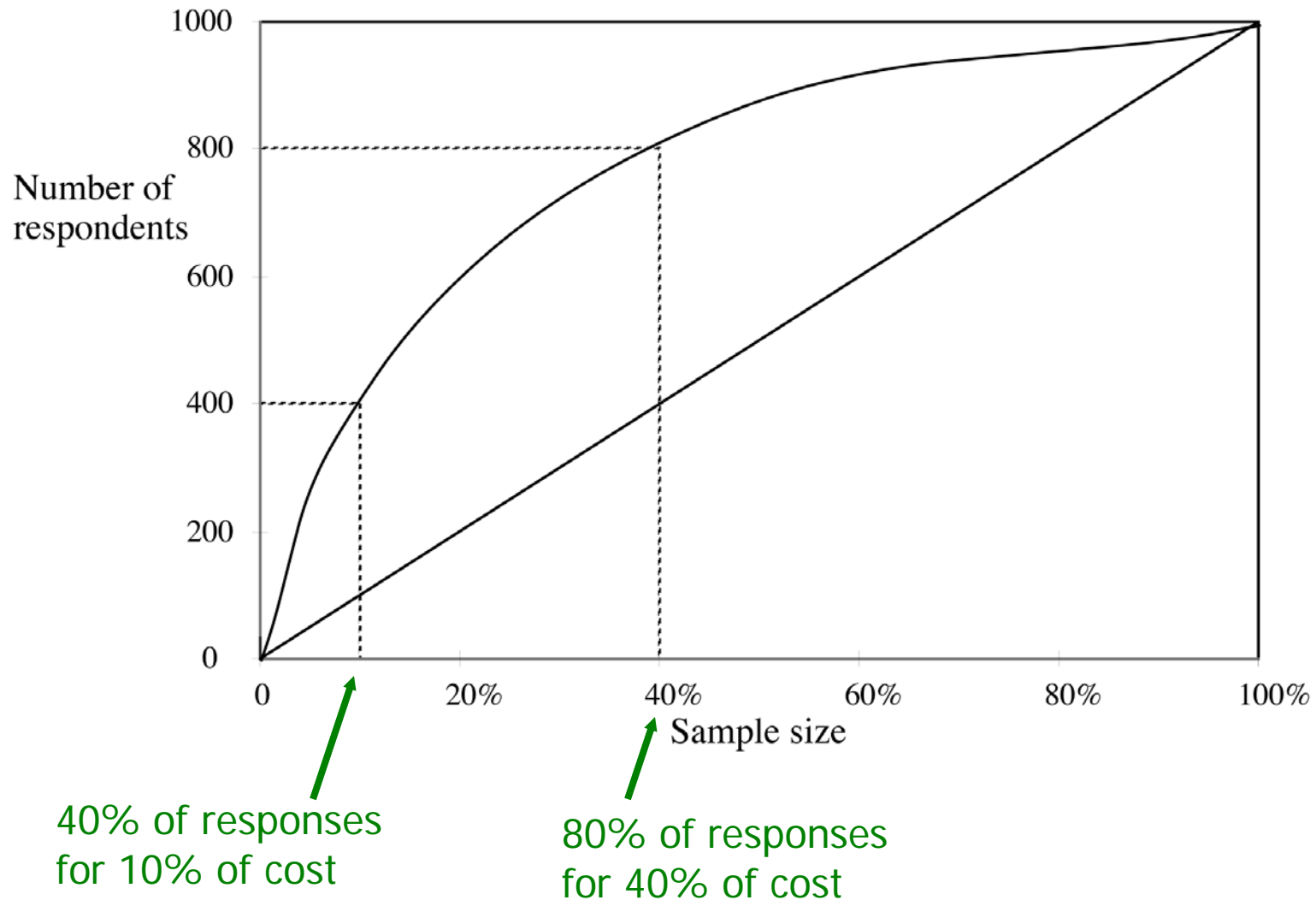
# Generating a lift chart

- ❖ Sort instances according to predicted probability of being positive:

	<b>Predicted probability</b>	<b>Actual class</b>
1	0.95	Yes
2	0.93	Yes
3	0.93	No
4	0.88	Yes
...	...	...

- ❖  $x$  axis is sample size  
 $y$  axis is number of true positives

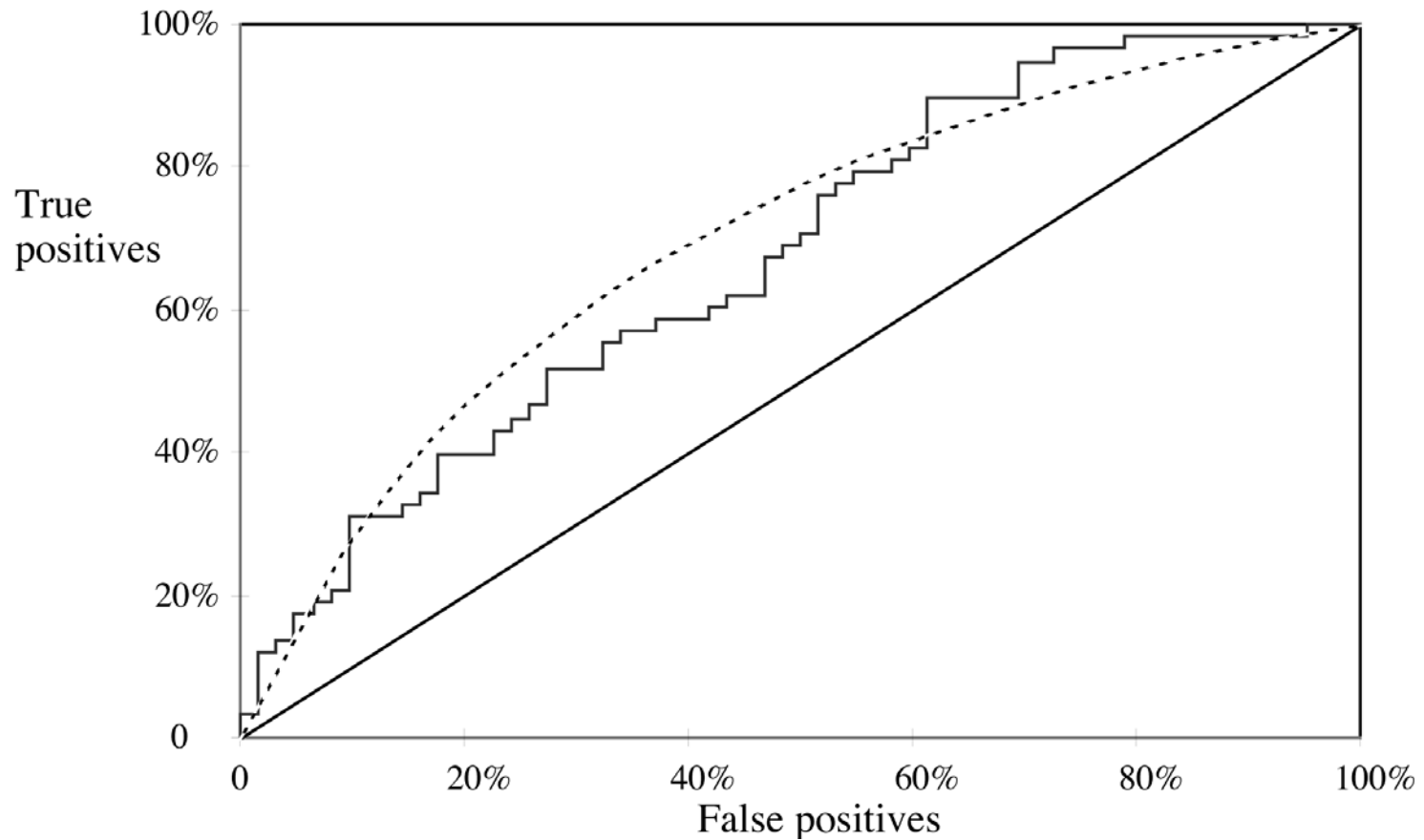
# A hypothetical lift chart



# ROC curves

- ❖ *ROC curves* are similar to lift charts
  - ❑ Stands for “receiver operating characteristic”
  - ❑ Used in signal detection to show tradeoff between hit rate and false alarm rate over noisy channel
- ❖ Differences to lift chart:
  - ❑ *y* axis shows percentage of true positives in sample *rather than absolute number*
  - ❑ *x* axis shows percentage of false positives in sample *rather than sample size*

# A sample ROC curve

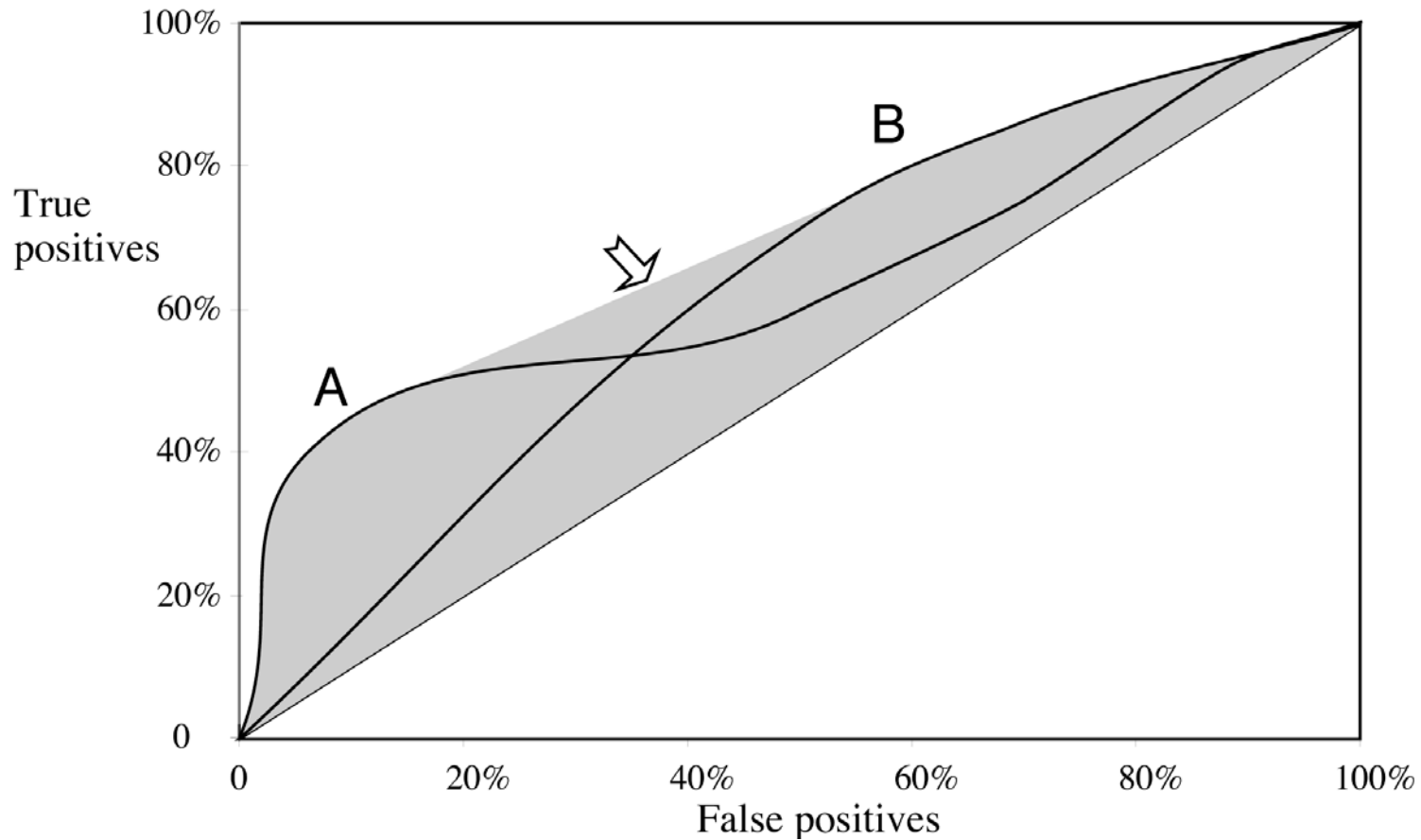


- ❖ Jagged curve—one set of test data
- ❖ Smooth curve—use cross-validation

# Cross-validation and ROC curves

- ❖ Simple method of getting a ROC curve using cross-validation:
  - ❑ Collect probabilities for instances in test folds
  - ❑ Sort instances according to probabilities
- ❖ This method is implemented in WEKA
- ❖ However, this is just one possibility
  - ❑ The method described in the book generates an ROC curve for each fold and averages them

# ROC curves for two schemes



- ❖ For a small, focused sample, use method A
- ❖ For a larger one, use method B
- ❖ In between, choose between A and B with appropriate probabilities

# The convex hull

- ❖ Given two learning schemes we can achieve any point on the convex hull!
- ❖ TP and FP rates for scheme 1:  $t_1$  and  $f_1$
- ❖ TP and FP rates for scheme 2:  $t_2$  and  $f_2$
- ❖ If scheme 1 is used to predict  $100 \times q$  % of the cases and scheme 2 for the rest, then
  - TP rate for combined scheme:  
 $q \times t_1 + (1 - q) \times t_2$
  - FP rate for combined scheme:  
 $q \times f_1 + (1 - q) \times f_2$

# Cost-sensitive learning

- ❖ Most learning schemes do not perform cost-sensitive learning
  - ❑ They generate the same classifier no matter what costs are assigned to the different classes
  - ❑ Example: standard decision tree learner
- ❖ Simple methods for cost-sensitive learning:
  - ❑ Resampling of instances according to costs
  - ❑ Weighting of instances according to costs
- ❖ Some schemes can take costs into account by varying a parameter, e.g. naïve Bayes



# Measures in information retrieval

- ❖ Percentage of retrieved documents that are relevant:  $precision = TP / (TP + FP)$
- ❖ Percentage of relevant documents that are returned:  $recall = TP / (TP + FN)$
- ❖ Precision/recall curves have hyperbolic shape
- ❖ Summary measures: average precision at 20%, 50% and 80% recall (*three-point average recall*)
- ❖  $F\text{-measure} = (2 \times recall \times precision) / (recall + precision)$

# Summary of measures

	Domain	Plot	Explanation
Lift chart	Marketing	TP Subset size	TP $(TP + FP) / (TP + FP + TN + FN)$
ROC curve	Communications	TP rate FP rate	$TP / (TP + FN)$ $FP / (FP + TN)$
Recall- precision curve	Information retrieval	Recall Precision	$TP / (TP + FN)$ $TP / (TP + FP)$



# Evaluating numeric prediction

- ❖ Same strategies: independent test set, cross-validation, significance tests, etc.
- ❖ Difference: error measures
- ❖ Actual target values:  $a_1 a_2 \dots a_n$
- ❖ Predicted target values:  $p_1 p_2 \dots p_n$
- ❖ Most popular measure: *mean-squared error*

$$\frac{(p_1 - a_1)^2 + \dots + (p_n - a_n)^2}{n}$$

- ❑ Easy to manipulate mathematically

# Other measures

- ❖ The *root mean-squared error* :

$$\sqrt{\frac{(p_1 - a_1)^2 + \dots + (p_n - a_n)^2}{n}}$$

- ❖ The *mean absolute error* is less sensitive to outliers than the mean-squared error:

$$\frac{|p_1 - a_1| + \dots + |p_n - a_n|}{n}$$

- ❖ Sometimes *relative* error values are more appropriate (e.g. 10% for an error of 50 when predicting 500)

# Improvement on the mean

❖ How much does the scheme improve on simply predicting the average?

❖ The *relative squared error* is ( $\bar{a}$  is the average):

$$\frac{(p_1 - a_1)^2 + \dots + (p_n - a_n)^2}{(\bar{a} - a_1)^2 + \dots + (\bar{a} - a_n)^2}$$

❖ The *relative absolute error* is:

$$\frac{|p_1 - a_1| + \dots + |p_n - a_n|}{|\bar{a} - a_1| + \dots + |\bar{a} - a_n|}$$

# Correlation coefficient

- ❖ Measures the *statistical correlation* between the predicted values and the actual values

$$\frac{S_{PA}}{\sqrt{S_P S_A}}$$

$$S_{PA} = \frac{\sum_i (p_i - \bar{p})(a_i - \bar{a})}{n-1} \quad S_P = \frac{\sum_i (p_i - \bar{p})^2}{n-1} \quad S_A = \frac{\sum_i (a_i - \bar{a})^2}{n-1}$$

- ❖ Scale independent, between -1 and +1
- ❖ Good performance leads to large values!

# Which measure?

- ❖ Best to look at all of them
- ❖ Often it doesn't matter
- ❖ Example:

	A	B	C	D
Root mean-squared error	67.8	91.7	63.3	57.4
Mean absolute error	41.3	38.5	33.4	29.2
Root rel squared error	42.2%	57.2%	39.4%	35.8%
Relative absolute error	43.1%	40.1%	34.8%	30.4%
Correlation coefficient	0.88	0.88	0.89	0.91

- ❖ D best
- ❖ C second-best
- ❖ A, B arguable



# The MDL principle

- ❖ MDL stands for *minimum description length*
- ❖ The description length is defined as:  
$$\begin{aligned} & \text{space required to describe a theory} \\ & + \\ & \text{space required to describe the theory's mistakes} \end{aligned}$$
- ❖ In our case the theory is the classifier and the mistakes are the errors on the training data
- ❖ Aim: we seek a classifier with minimal DL
- ❖ MDL principle is a *model selection criterion*



# Model selection criteria

- ❖ Model selection criteria attempt to find a good compromise between:
  - The complexity of a model
  - Its prediction accuracy on the training data
- ❖ Reasoning: a good model is a simple model that achieves high accuracy on the given data
- ❖ Also known as *Occam's Razor* : the best theory is the smallest one that describes all the facts

**William of Ockham, born in the village of Ockham in Surrey (England) about 1285, was the most influential philosopher of the 14th century and a controversial theologian.**



# Elegance vs. errors

- ❖ Theory 1: very simple, elegant theory that explains the data almost perfectly
- ❖ Theory 2: significantly more complex theory that reproduces the data without mistakes
- ❖ Theory 1 is probably preferable
- ❖ Classical example: Kepler's three laws on planetary motion
  - ❑ Less accurate than Copernicus's latest refinement of the Ptolemaic theory of epicycles

# MDL and compression

- ❖ MDL principle relates to data compression:
  - ❑ The best theory is the one that compresses the data the most
  - ❑ I.e. to compress a dataset we generate a model and then store the model and its mistakes
- ❖ We need to compute
  - (a) size of the model, and
  - (b) space needed to encode the errors
- ❖ (b) easy: use the informational loss function
- ❖ (a) need a method to encode the model

# MDL and Bayes's theorem

- ❖  $L[T]$  = "length" of the theory
- ❖  $L[E|T]$  = training set encoded wrt the theory
- ❖ Description length =  $L[T] + L[E|T]$
- ❖ Bayes's theorem gives *a posteriori* probability of a theory given the data:

$$\Pr[T | E] = \frac{\Pr[E | T] \Pr[T]}{\Pr[E]}$$

- ❖ Equivalent to:

$$-\log \Pr[T | E] = -\log \Pr[E | T] - \log \Pr[T] + \underbrace{\log \Pr[E]}_{\text{constant}}$$

# MDL and MAP

- ❖ MAP stands for *maximum a posteriori probability*
- ❖ Finding the MAP theory corresponds to finding the MDL theory
- ❖ Difficult bit in applying the MAP principle: determining the prior probability  $\Pr[T]$  of the theory
- ❖ Corresponds to difficult part in applying the MDL principle: coding scheme for the theory
- ❖ I.e. if we know a priori that a particular theory is more likely we need less bits to encode it

# Discussion of MDL principle

- ❖ Advantage: makes full use of the training data when selecting a model
- ❖ Disadvantage 1: appropriate coding scheme/prior probabilities for theories are crucial
- ❖ Disadvantage 2: no guarantee that the MDL theory is the one which minimizes the expected error
- ❖ Note: Occam's Razor is an axiom!
- ❖ Epicurus's *principle of multiple explanations*: keep all theories that are consistent with the data

# Bayesian model averaging

- ❖ Reflects Epicurus's principle: all theories are used for prediction weighted according to  $P[T|E]$
- ❖ Let  $I$  be a new instance whose class we must predict
- ❖ Let  $C$  be the random variable denoting the class
- ❖ Then BMA gives the probability of  $C$  given
  - $I$
  - training data  $E$
  - possible theories  $T_j$

$$\Pr[C | I, E] = \sum_j \Pr[C | I, T_j] \Pr[T_j | E]$$

# MDL and clustering

- ❖ Description length of theory:  
bits needed to encode the clusters
  - e.g. cluster centers
- ❖ Description length of data given theory:  
encode cluster membership and position relative to cluster
  - e.g. distance to cluster center
- ❖ Works if coding scheme uses less code space for small numbers than for large ones
- ❖ With nominal attributes, must communicate probability distributions for each cluster