



Evaluation

Data Mining and Text Mining (UIC 583 @ Politecnico di Milano)

- What the issues?
- What evaluation metrics?
- What evaluation of performance?
- How to compare performance among models?
- Which model should we select?

What the issues?

- ❑ Which measures?
 - ▶ Classification accuracy
 - ▶ Total cost/benefit
when different errors involve different costs
 - ▶ Lift and ROC curves
 - ▶ Error in numeric predictions

- ❑ How reliable are the predicted results ?

- ❑ How much should we believe in what was learned?
- ❑ How predictive is the model we learned?

- ❑ Error on the training data is not a good indicator of performance on future data

- ❑ Question: Why?
- ❑ Answer:
 - ▶ Since the classifier has been learned from the very same training data, any estimate based on that data will be optimistic.
 - ▶ In addition, new data will probably not be exactly the same as the training data!

- ❑ Overfitting: fitting the training data too precisely usually leads to poor results on new data

What metrics?

- ❑ Metrics for Performance Evaluation
 - ▶ How to evaluate the performance of a model?

- ❑ Methods for Performance Evaluation
 - ▶ How to obtain reliable estimates?

- ❑ Methods for Model Comparison
 - ▶ How to compare the relative performance among competing models?

- ❑ Model selection
 - ▶ Which model should we prefer?

- ❑ Focus on the predictive capability of a model
 - ▶ Rather than how fast it takes to classify or build models, scalability, etc.
- ❑ Confusion Matrix:

	PREDICTED CLASS		
	Yes	No	
ACTUAL CLASS	Yes	a	b
	No	c	d

a: TP (true positive) c: FP (false positive)

b: FN (false negative) d: TN (true negative)

	PREDICTED CLASS		
	Yes	No	
ACTUAL CLASS	Yes	a (TP)	b (FN)
	No	c (FP)	d (TN)

□ Most widely-used metric:

$$Accuracy = \frac{a + d}{a + b + c + d} = \frac{TP + TN}{TP + TN + FP + FN}$$

- ❑ Consider a 2-class problem
 - ▶ Number of Class 0 examples = 9990
 - ▶ 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

	PREDICTED CLASS		
	$C(i j)$	Yes	No
ACTUAL CLASS	Yes	$C(\text{Yes} \text{Yes})$	$C(\text{No} \text{Yes})$
	No	$C(\text{Yes} \text{No})$	$C(\text{No} \text{No})$

$C(i|j)$: Cost of misclassifying class j example as class i

Cost Matrix	PREDICTED CLASS		
	C(i j)	+	-
ACTUAL CLASS	+	-1	100
	-	1	0

Model M ₁	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	150	40
	-	60	250

Model M ₂	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	250	45
	-	5	200

Accuracy = 80%

Cost = 3910

Accuracy = 90%

Cost = 4255

Count	PREDICTED CLASS		
		Yes	No
ACTUAL CLASS	Yes	a	b
	No	c	d

Accuracy is proportional to cost if

1. $C(\text{Yes}|\text{No})=C(\text{No}|\text{Yes}) = q$
2. $C(\text{Yes}|\text{Yes})=C(\text{No}|\text{No}) = p$

$$N = a + b + c + d$$

$$\text{Accuracy} = (a + d)/N$$

Cost	PREDICTED CLASS		
		Yes	No
ACTUAL CLASS	Yes	p	q
	No	q	p

$$\begin{aligned} \text{Cost} &= p(a + d) + q(b + c) \\ &= p(a + d) + q(N - a - d) \\ &= qN - (q - p)(a + d) \\ &= N[q - (q - p) \times \text{Accuracy}] \end{aligned}$$

$$Precision(p) = \frac{TP}{TP + FP} = \frac{a}{a + c}$$

The higher the precision, the lower the FPs

$$Recall(r) = \frac{TP}{TP + FN} = \frac{a}{a + b}$$

The higher the precision, the lower the FNs

$$F1 - measure = \frac{2rp}{r + p} = \frac{2a}{2a + b + c}$$

The higher the F1, the lower the FPs & FNs

- ❑ Precision is biased towards C(Yes|Yes) & C(Yes|No),
- ❑ Recall is biased towards C(Yes|Yes) & C(No|Yes)
- ❑ F1-measure is biased towards all except C(No|No), it is high when both precision and recall are reasonably high

$$WeightedAccuracy = \frac{w_a \times a + w_d \times d}{w_a \times a + w_b \times b + w_c \times c + w_d \times d}$$

How to evaluate performance?

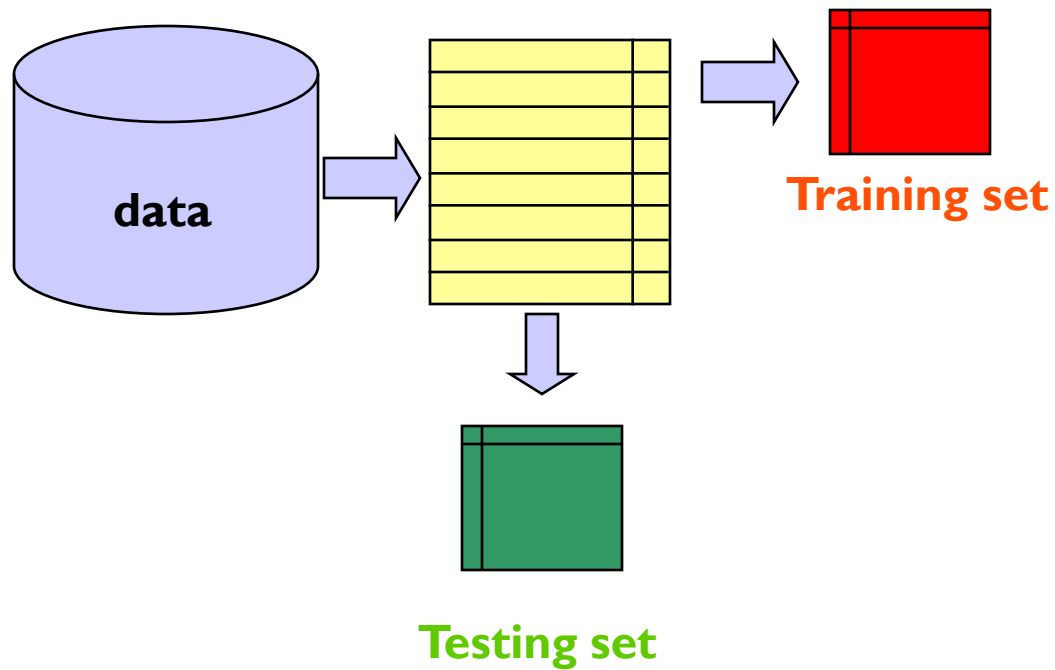
- ❑ Metrics for Performance Evaluation
 - ▶ How to evaluate the performance of a model?

- ❑ **Methods for Performance Evaluation**
 - ▶ How to obtain reliable estimates?

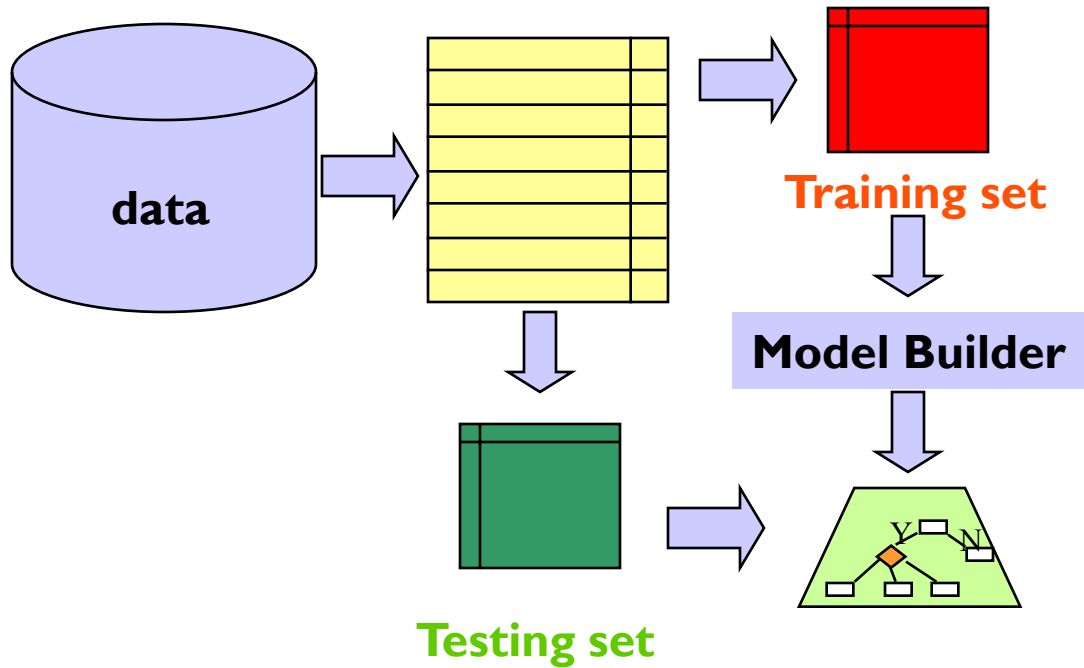
- ❑ Methods for Model Comparison
 - ▶ How to compare the relative performance among competing models?

- ❑ Model selection
 - ▶ Which model should be preferred?

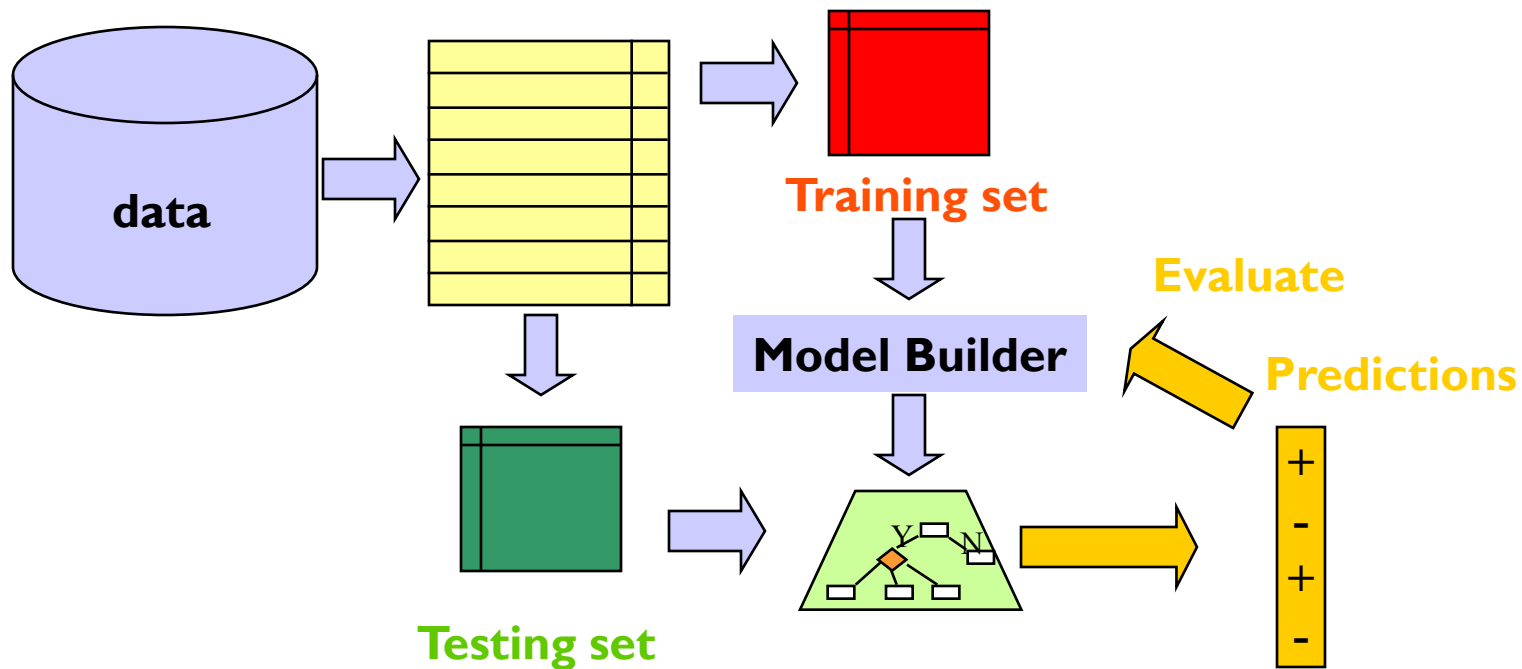
Classification Step 1: Split data into train and test sets



Classification Step 2: Build a model on a training set



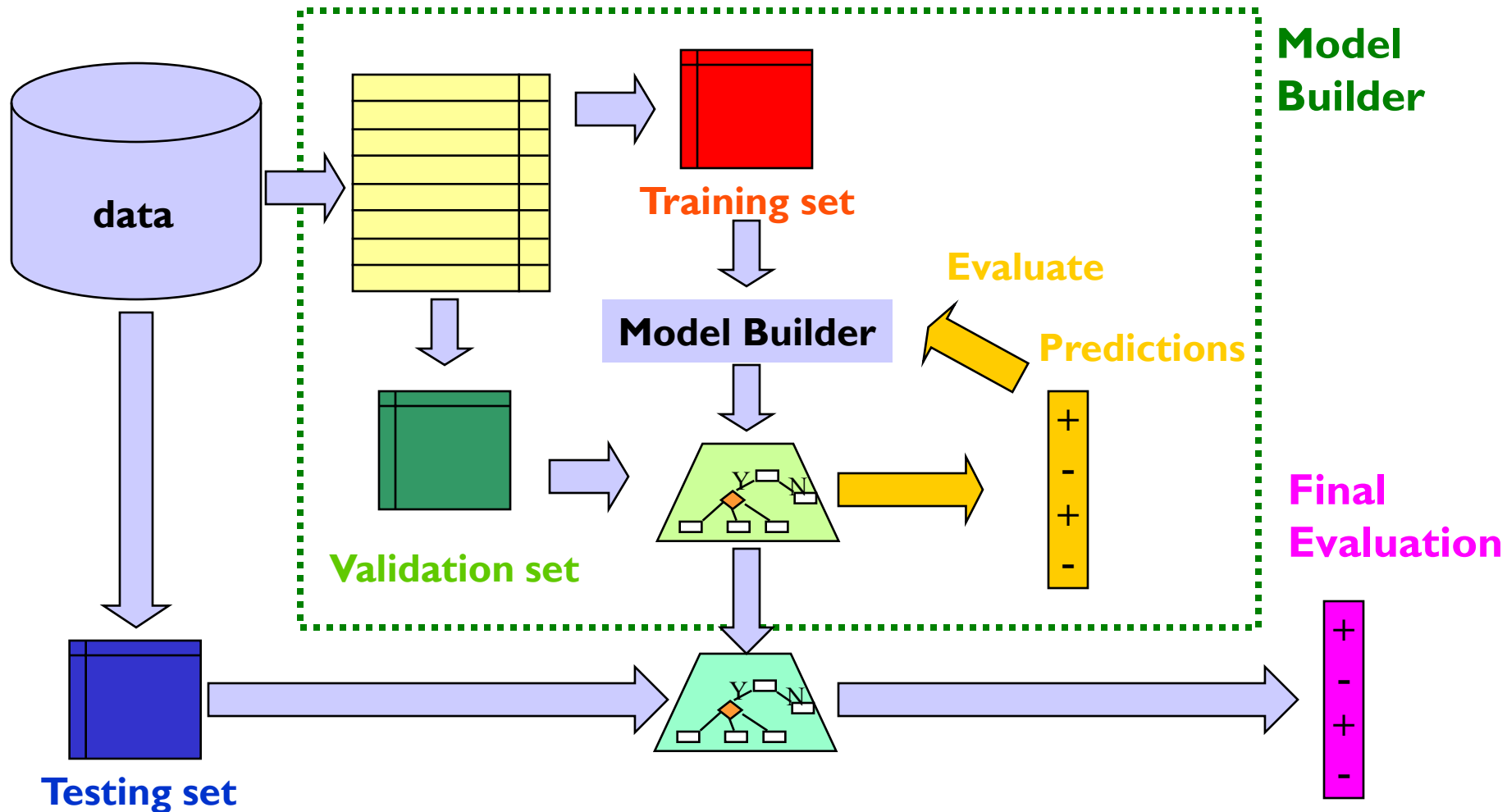
Classification Step 3: Evaluate on test set (Re-train?)



- ❑ 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: builds the basic structure**
 - ▶ **Stage 2: optimizes 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

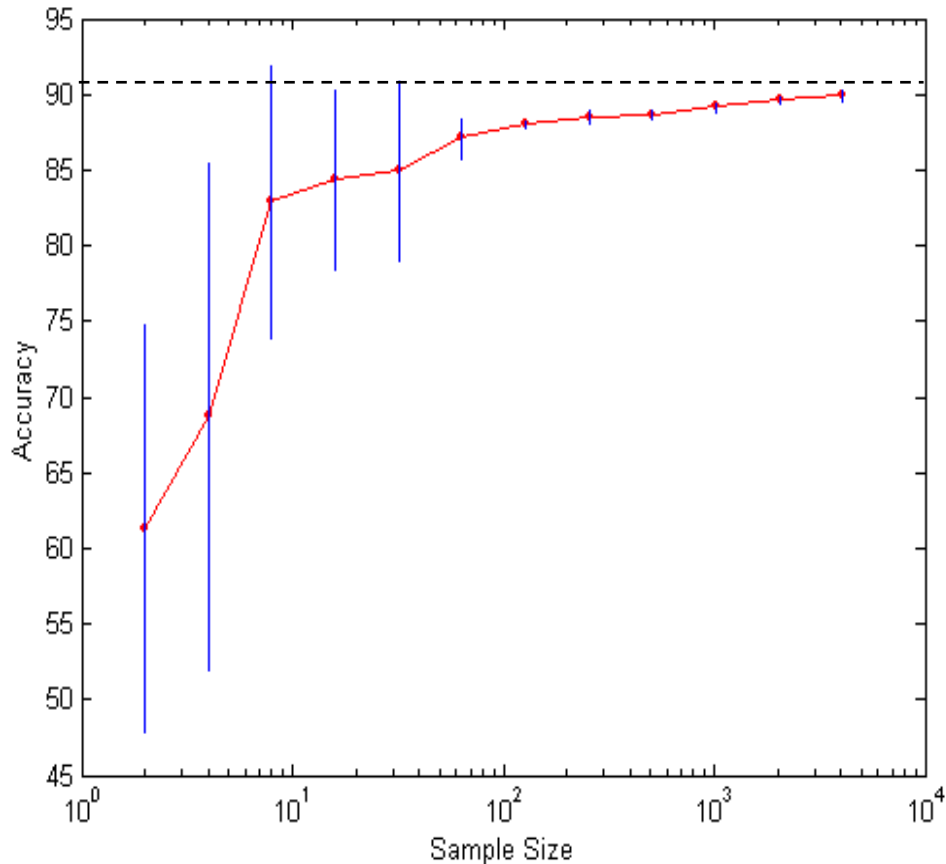
- ❑ Once evaluation is complete, **all the data** can be used to build the final classifier
- ❑ Generally, the larger the training data the better the classifier (but returns diminish)
- ❑ The larger the test data the more accurate the error estimate

Classification Step: Train, validation, and test



- ❑ How to obtain a reliable estimate of performance?

- ❑ Performance of a model may depend on other factors besides the learning algorithm:
 - ▶ Class distribution
 - ▶ Cost of misclassification
 - ▶ Size of training and test sets



- ❑ Learning curve shows how accuracy changes with varying sample size
- ❑ Requires a sampling schedule for creating learning curve:
 - ❑ Arithmetic sampling
 - ❑ Geometric sampling
- ❑ Effect of small sample size:
 - ▶ Bias in the estimate
 - ▶ Variance of estimate

- ❑ Holdout
 - ▶ Reserve $\frac{1}{2}$ for training and $\frac{1}{2}$ for testing
 - ▶ Reserve $\frac{2}{3}$ for training and $\frac{1}{3}$ for testing
- ❑ Random subsampling
 - ▶ Repeated holdout
- ❑ Cross validation
 - ▶ Partition data into k disjoint subsets
 - ▶ k -fold: train on $k-1$ partitions, test on the remaining one
 - ▶ Leave-one-out: $k=n$
- ❑ Stratified sampling
 - ▶ oversampling vs undersampling
- ❑ Bootstrap
 - ▶ Sampling with replacement

- ❑ The holdout method reserves a certain amount for testing and uses the remainder for training
- ❑ Usually, one third for testing, the rest for training
- ❑ For small or “unbalanced” datasets, samples might not be representative
- ❑ For instance, few or none instances of some classes
- ❑ Stratified sample
 - ▶ Advanced version of balancing the data
 - ▶ Make sure that each class is represented with approximately equal proportions in both subsets

- ❑ 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

- ❑ Avoids overlapping test sets
 - ▶ First step: data is split into k subsets of equal size
 - ▶ Second step: each subset in turn is used for testing and the remainder for training

- ❑ This is called k -fold cross-validation

- ❑ Often the subsets are stratified before cross-validation is performed

- ❑ The error estimates are averaged to yield an overall error estimate

- ❑ 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
- ❑ 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)
- ❑ Other approaches appear to be robust, e.g., 5x2 crossvalidation

- ❑ It is 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

- ❑ Disadvantage of Leave-One-Out is that: 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!

- ❑ Crossvalidation 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

- ❑ 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

- ❑ The error estimate on the test data will be very pessimistic, since training was on just $\sim 63\%$ 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

- ❑ Probably the best way of estimating performance for very small datasets
- ❑ However, it has some problems
 - ▶ Consider a random dataset with a 50% class distribution
 - ▶ A perfect memorizer will achieve 0% resubstitution error and $\sim 50\%$ error on test data
 - ▶ Bootstrap estimate for this classifier:

$$err = 0.632 \cdot 50\% + 0.368 \cdot 0\% = 31.6\%$$

- ▶ True expected error: 50%

Model comparison

- ❑ Metrics for Performance Evaluation
 - ▶ How to evaluate the performance of a model?

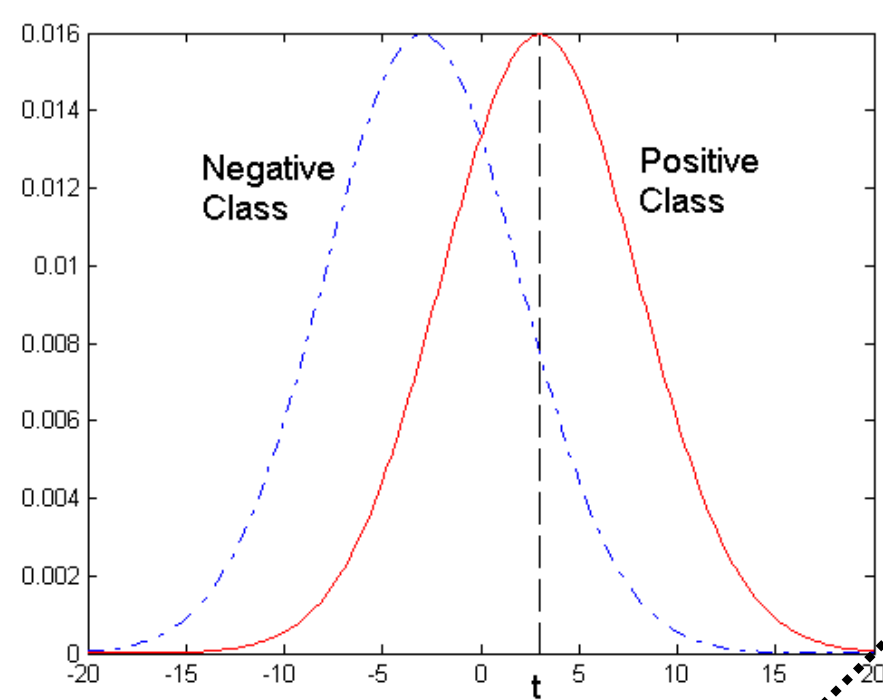
- ❑ Methods for Performance Evaluation
 - ▶ How to obtain reliable estimates?

- ❑ **Methods for Model Comparison**
 - ▶ How to compare the relative performance among competing models?

- ❑ Model selection
 - ▶ Which model should be preferred?

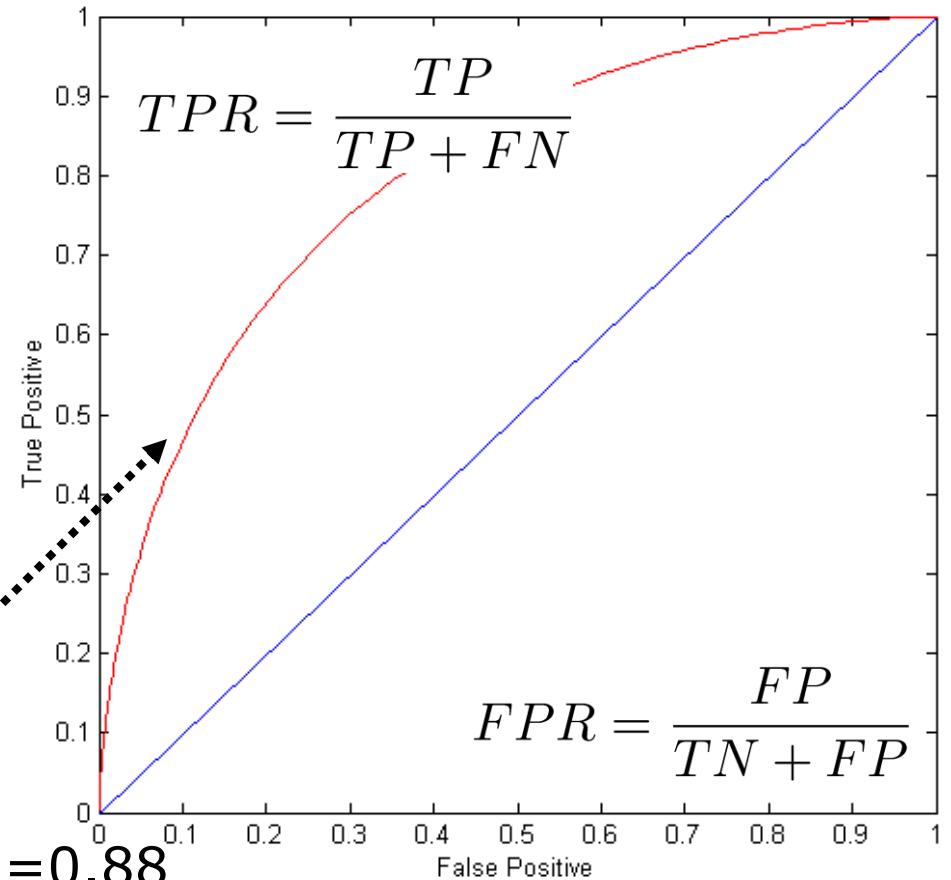
- ❑ Developed in 1950s for signal detection theory to analyze noisy signals
- ❑ Characterize the trade-off between positive hits and false alarms
- ❑ ROC curve plots TP (on the y-axis) against FP (on the x-axis)
- ❑ Performance of each classifier represented as a point on the ROC curve
- ❑ Changing the threshold of algorithm, sample distribution or cost matrix changes the location of the point

- 1-dimensional data set containing 2 classes (positive and negative)
- any points located at $x > t$ is classified as positive

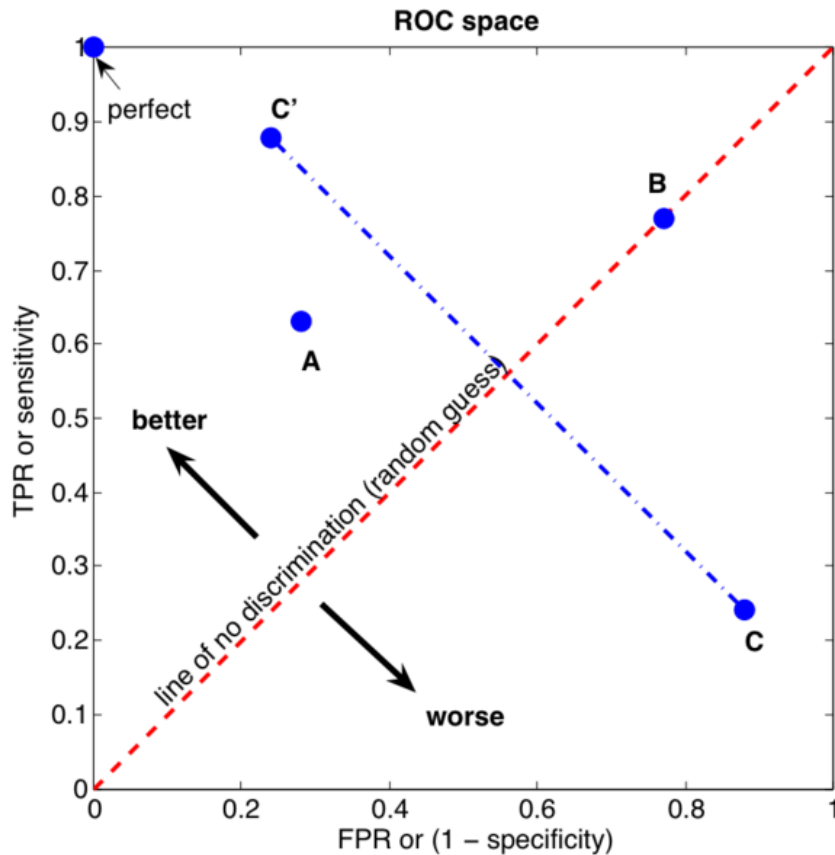


At threshold t :

$TP=0.5, FN=0.5, FP=0.12, FN=0.88$



ROC Curve



A

TP=63	FP=28	91
FN=37	TN=72	109
100	100	200

TPR = 0.63
FPR = 0.28
ACC = 0.68

B

TP=77	FP=77	154
FN=23	TN=23	46
100	100	200

TPR = 0.77
FPR = 0.77
ACC = 0.50

C

TP=24	FP=88	112
FN=76	TN=12	88
100	100	200

TPR = 0.24
FPR = 0.88
ACC = 0.18

C'

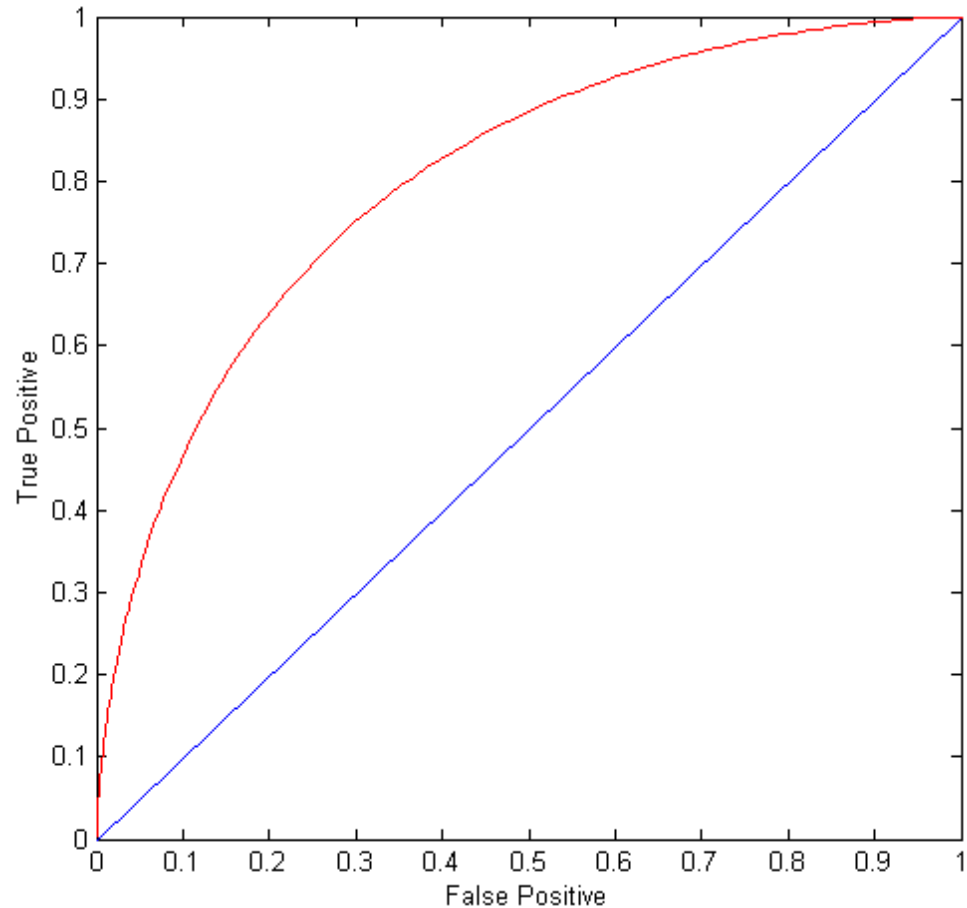
TP=88	FP=24	112
FN=12	TN=76	88
100	100	200

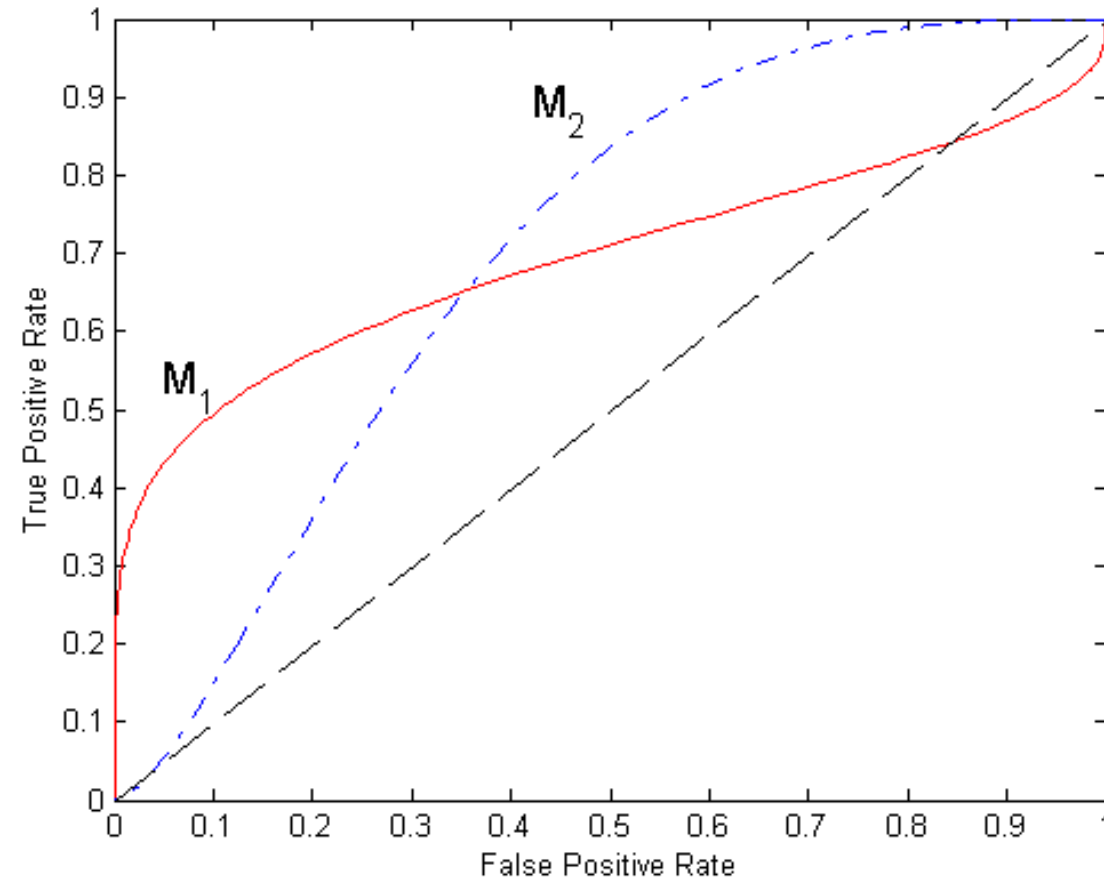
TPR = 0.88
FPR = 0.24
ACC = 0.82

(TP,FP):

- ❑ (0,0): declare everything to be negative class
- ❑ (1,1): declare everything to be positive class
- ❑ (1,0): ideal

- ❑ Diagonal line:
 - ▶ Random guessing
 - ▶ Below diagonal line, prediction is opposite of the true class





- No model consistently outperform the other
 - M_1 is better for small FPR
 - M_2 is better for large FPR
- Area Under the ROC curve
 - Ideal, area = 1
 - Random guess, area = 0.5

Instance	$P(+ A)$	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Use classifier that produces posterior probability for each test instance $P(+|A)$
- Sort the instances according to $P(+|A)$ in decreasing order
- Apply threshold at each unique value of $P(+|A)$
- Count the number of TP, FP, TN, FN at each threshold
- TP rate, $TPR = TP/(TP+FN)$
- FP rate, $FPR = FP/(FP + TN)$

Instance	P(+ A)	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Threshold = 1.0
 - None is classified as +
 - TP = 0, TN = 5, FN = 5
 - TPR = 0 FPR = 0
- Threshold = 0.95
 - One is classified as +
 - TP = 1, TN = 5, FN = 4
 - TPR = 0.2 FPR = 0
- Threshold = 0.93
 - Two + classified as +
 - TP = 2, TN = 5, FN = 3
 - TPR = 0.4 FPR = 0

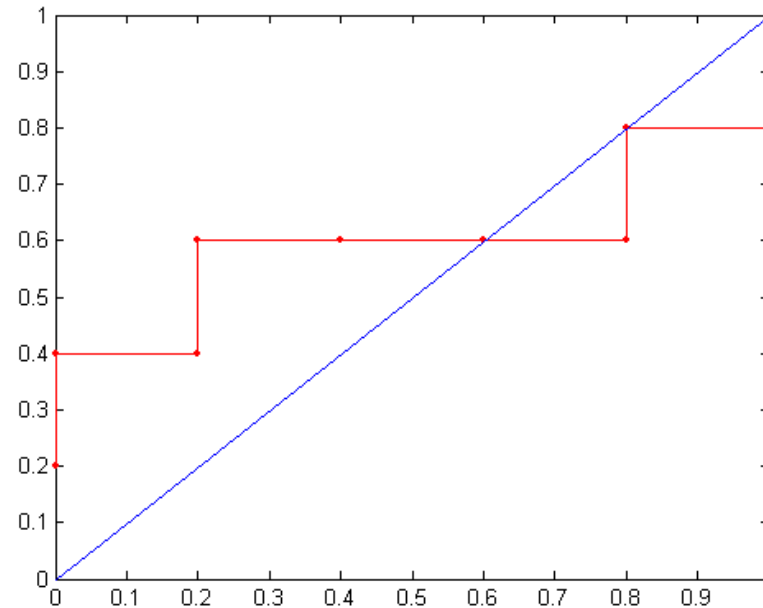
Instance	$P(+ A)$	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

Instance	$P(+ A)$	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

Instance	$P(+ A)$	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
→ TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
→ FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve:



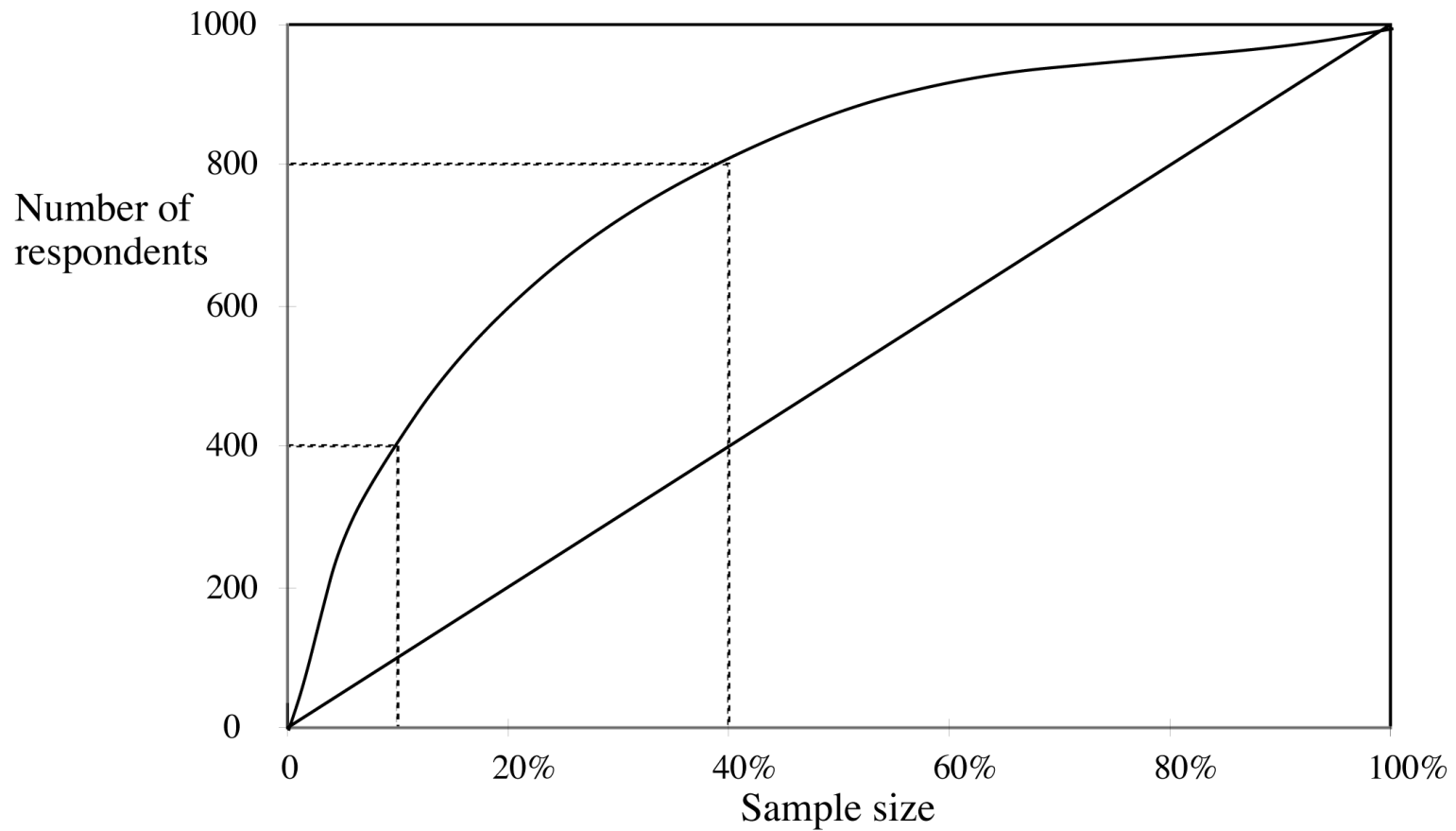
- ❑ Lift is a measure of the effectiveness of a predictive model calculated as the ratio between the results obtained with and without the predictive model.
- ❑ Cumulative gains and lift charts are visual aids for measuring model performance
- ❑ Both charts consist of a lift curve and a baseline
- ❑ The greater the area between the lift curve and the baseline, the better the model

- ❑ Mass mailout of promotional offers (1000000)
- ❑ The proportion who normally respond is 0.1%, that is 1000 responses
- ❑ A data mining tool can identify a subset of a 100000 for which the response rate is 0.4%, that is 400 responses
- ❑ In marketing terminology, the increase of response rate is known as the lift factor yielded by the model.
- ❑ The same data mining tool, may be able to identify 400000 households for which the response rate is 0.2%, that is 800 respondents corresponding to a lift factor of 2.
- ❑ The overall goal is to find subsets of test instances that have a high proportion of true positive

- ❑ Given a scheme that outputs probability, sort the instances in descending order according to the predicted probability
- ❑ Select the instance subset starting from the one with the highest predicted probability

Rank	Predicted Probability	Actual Class
1	0.95	Yes
2	0.93	Yes
3	0.93	No
4	0.88	yes
...

- ❑ In lift chart, x axis is sample size and y axis is number of true positives



- Given two models:
 - ▶ Model M1: accuracy = 85%, tested on 30 instances
 - ▶ Model M2: accuracy = 75%, tested on 5000 instances

- Can we say M1 is better than M2?
 - ▶ How much confidence can we place on accuracy of M1 and M2?
 - ▶ Can the difference in performance measure be explained as a result of random fluctuations in the test set?

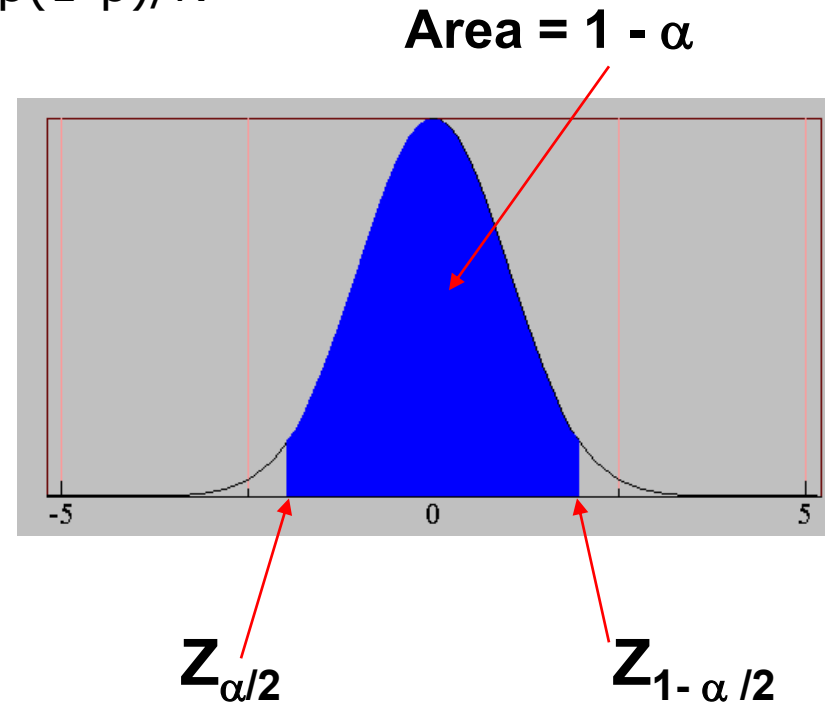
- ❑ Prediction can be regarded as a Bernoulli trial
 - ▶ A Bernoulli trial has 2 possible outcomes
 - ▶ Possible outcomes for prediction: correct or wrong
 - ▶ Collection of Bernoulli trials has a Binomial distribution

- ❑ Given x (# of correct predictions) or equivalently, $\text{acc}=x/N$, and N (# of test instances),

- ❑ Can we predict p (true accuracy of model)?

- For large test sets ($N > 30$), the accuracy acc has a normal distribution with mean p and variance $p(1-p)/N$
- Confidence Interval for p :

$$P\left(Z_{\alpha/2} < \frac{acc - p}{\sqrt{p(1-p)/N}} < Z_{1-\alpha/2}\right) = 1 - \alpha$$

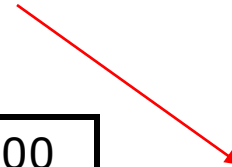


$$p = \frac{2 \times N \times acc + Z_{\alpha/2}^2 \pm \sqrt{Z_{\alpha/2}^2 + 4 \times N \times acc - 4 \times N \times acc^2}}{2(N + Z_{\alpha/2}^2)}$$

- Consider a model that produces an accuracy of 80% when evaluated on 100 test instances:
 - ▶ $N=100$, $acc = 0.8$
 - ▶ Let $1-\alpha = 0.95$ (95% confidence)
 - ▶ From probability table, $Z_{\alpha/2}=1.96$

N	50	100	500	1000	5000
p(lower)	0.670	0.711	0.763	0.774	0.789
p(upper)	0.888	0.866	0.833	0.824	0.811

$1-\alpha$	Z
0.99	2.58
0.98	2.33
0.95	1.96
0.90	1.65



- ❑ Given two models, say M1 and M2, which is better?
- ❑ M1 is tested on D1 (size= n_1), found error rate = e_1
- ❑ M2 is tested on D2 (size= n_2), found error rate = e_2
- ❑ Assume D1 and D2 are independent
- ❑ If n_1 and n_2 are sufficiently large, then

$$e_1 \sim N(\mu_1, \sigma_1)$$

$$e_2 \sim N(\mu_2, \sigma_2)$$

- ❑ Approximate:

$$\hat{\sigma}_i = \frac{e_i(1-e_i)}{n_i}$$

- To test if performance difference is statistically significant,
 $d = e1 - e2$
- $d \sim \mathcal{N}(d_t, \sigma_t)$ where d_t is the true difference
- Since D1 and D2 are independent, their variance adds up:

$$\begin{aligned}\sigma_t^2 &= \sigma_1^2 + \sigma_2^2 \cong \hat{\sigma}_1^2 + \hat{\sigma}_2^2 \\ &= \frac{e1(1-e1)}{n1} + \frac{e2(1-e2)}{n2}\end{aligned}$$

- At $(1-\alpha)$ confidence level,

$$d_t = d \pm Z_{\alpha/2} \hat{\sigma}_t$$

- Given: M1: $n_1 = 30$, $e_1 = 0.15$
M2: $n_2 = 5000$, $e_2 = 0.25$
- $d = |e_2 - e_1| = 0.1$ (2-sided test)

$$\hat{\sigma}_d = \frac{0.15(1-0.15)}{30} + \frac{0.25(1-0.25)}{5000} = 0.0043$$

- At 95% confidence level, $Z_{\alpha/2} = 1.96$

$$d_t = 0.100 \pm 1.96 \times \sqrt{0.0043} = 0.100 \pm 0.128$$

- The interval contains 0, therefore the difference may not be statistically significant

- Each learning algorithm may produce k models:
 - ▶ L1 may produce M_{11} , M_{12} , ..., M_{1k}
 - ▶ L2 may produce M_{21} , M_{22} , ..., M_{2k}

- If models are generated on the same test sets D_1, D_2, \dots, D_k (e.g., via cross-validation)
 - ▶ For each set: compute $d_j = e_{1j} - e_{2j}$
 - ▶ d_j has mean d_t and variance σ_t
 - ▶ Estimate:

$$\hat{\sigma}_t^2 = \frac{\sum_{j=1}^k (d_j - \bar{d})^2}{k(k-1)}$$

$$d_t = d \pm t_{1-\alpha, k-1} \hat{\sigma}_t$$

Model selection

- ❑ Metrics for Performance Evaluation
 - ▶ How to evaluate the performance of a model?

- ❑ Methods for Performance Evaluation
 - ▶ How to obtain reliable estimates?

- ❑ Methods for Model Comparison
 - ▶ How to compare the relative performance among competing models?

- ❑ **Model selection**
 - ▶ Which model should be preferred?

- ❑ MDL stands for minimum description length
- ❑ The description length is defined as:

space required to describe a theory
+
space required to describe the theory's mistakes

- ❑ 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 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.



- ❑ 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 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

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

$$P(T|E) = \frac{P(E|T)P(T)}{P(E)}$$

- Equivalent to,

$$-\log P(T|E) = -\log P(E|T) - \log P(T) + \underbrace{\log P(E)}_{\text{constant}}$$

- ❑ 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

- ❑ 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' principle of multiple explanations: keep all theories that are consistent with the data

- ❑ Reflects Epicurus' 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
- ❑ Let E be the training data
- ❑ Let T_j be the possible theories
- ❑ Then, BMA gives the probability of C given,

$$P(C|I \wedge E) = \sum_j P(C|I \wedge T_j)P(T_j|E)$$

- ❑ Among the several algorithms, which one is the “best”?
 - ▶ Some algorithms have a lower computational complexity
 - ▶ Different algorithms provide different representations
 - ▶ Some algorithms allow the specification of prior knowledge

- ❑ If we are interested in the generalization performance, are there any reasons to prefer one classifier over another?

- ❑ Can we expect any classification method to be superior or inferior overall?

- ❑ According to the **No Free Lunch Theorem**, the answer to all these questions is known

- ❑ If the goal is to obtain good generalization performance, there are no context-independent or usage-independent reasons to favor one classification method over another
- ❑ If one algorithm seems to outperform another in a certain situation, it is a consequence of its fit to the particular problem, not the general superiority of the algorithm
- ❑ When confronting a new problem, this theorem suggests that we should focus on the aspects that matter most
 - ▶ Prior information
 - ▶ Data distribution
 - ▶ Amount of training data
 - ▶ Cost or reward
- ❑ The theorem also justifies skepticism regarding studies that “demonstrate” the overall superiority of a certain algorithm

"[A]ll algorithms that search for an extremum of a cost [objective] function perform exactly the same, when averaged over all possible cost functions." [1]

"[T]he average performance of any pair of algorithms across all possible problems is identical." [2]

- ❑ Wolpert, D.H., Macready, W.G. (1995), No Free Lunch Theorems for Search, Technical Report SFI-TR-95-02-010 (Santa Fe Institute).
- ❑ Wolpert, D.H., Macready, W.G. (1997), No Free Lunch Theorems for Optimization, *IEEE Transactions on Evolutionary Computation* **1**, 67.

- Let
 - ▶ F be the unknown concept
 - ▶ n the number of training examples
 - ▶ $\varepsilon(E|D)$ be the error of the learning algorithm given the data
 - ▶ $P_i(h|D)$ the probability that algorithm i generates hypothesis h from the training data D

- For any two learning algorithm, $P_1(h|D)$ and $P_2(h|D)$
 1. Uniformly averaged over all the target function F ,
 $\varepsilon_1(E|F,n) - \varepsilon_2(E|F,n) = 0$
 2. For any fixed training set D , uniformly averaged over F ,
 $\varepsilon_1(E|F,D) - \varepsilon_2(E|F,D) = 0$
 3. Uniformly averaged over all priors $P(F)$,
 $\varepsilon_1(E|n) - \varepsilon_2(E|n) = 0$
 4. For any training set D , uniformly averaged over $P(F)$,
 $\varepsilon_1(E|D) - \varepsilon_2(E|D) = 0$

- For any two learning algorithm, $P_1(h|D)$ and $P_2(h|D)$
 1. Uniformly averaged over all the target function F ,
 $\epsilon_1(E|F,n) - \epsilon_2(E|F,n) = 0$
 2. For any fixed training set D , uniformly averaged over F ,
 $\epsilon_1(E|F,D) - \epsilon_2(E|F,D) = 0$
 3. Uniformly averaged over all priors $P(F)$,
 $\epsilon_1(E|n) - \epsilon_2(E|n) = 0$
 4. For any training set D , uniformly averaged over $P(F)$,
 $\epsilon_1(E|D) - \epsilon_2(E|D) = 0$

(1) No matter how clever we choose a good algorithm $P_1(h|D)$ and a bad algorithm $P_2(h|D)$, if all functions are equally likely, then the "good" algorithm will not outperform the "bad" one

(2) Even if we know D , then averaged over all the target functions no learning algorithm will outperform another one

Summary

- ❑ Evaluation metrics
 - ▶ Accuracy, precision, recall
 - ▶ Accuracy vs Cost
 - ▶ Cost matrix
- ❑ What evaluation of performance?
 - ▶ Cross-validation
 - ▶ Bootstrap
- ❑ How to compare performance among models?
 - ▶ Lift charts
 - ▶ Roc curves
 - ▶ T-test
- ❑ Model selection
 - ▶ Simplicity
 - ▶ Averaging
 - ▶ No Free Lunch