دانشگاه آزاد اسلامی واحد تبریز

نام درس: داده کاوی

بخش: طبقه بندی توسط درخت تصمیم و ارزیابی مدل

نام استاد: دکتر مسعود کارگر

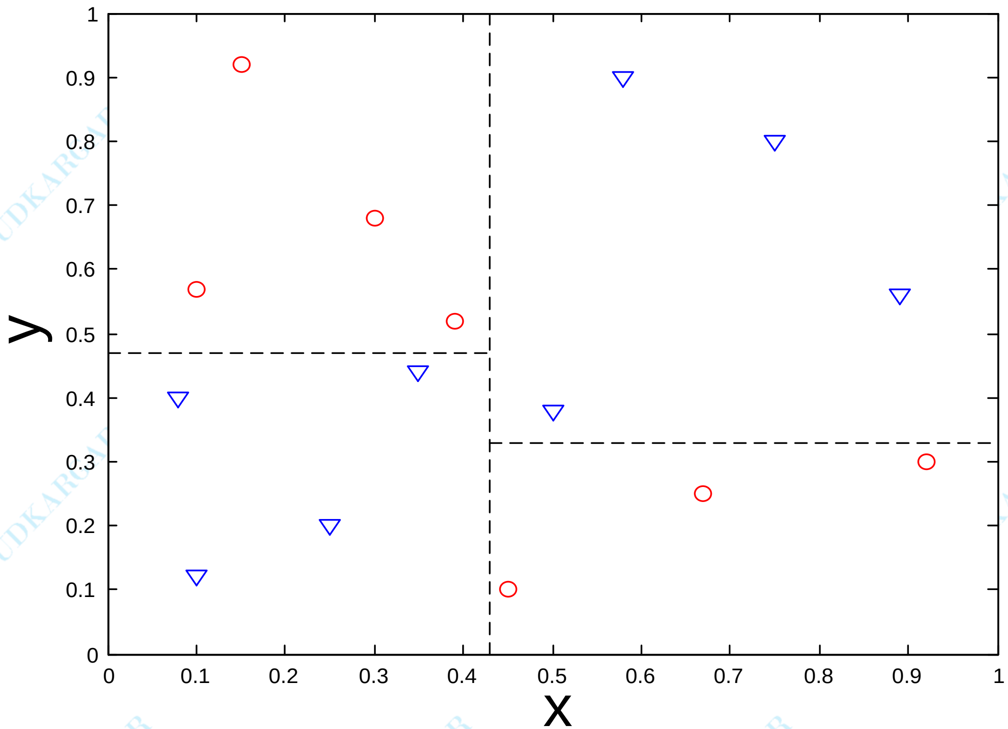
Roadmap

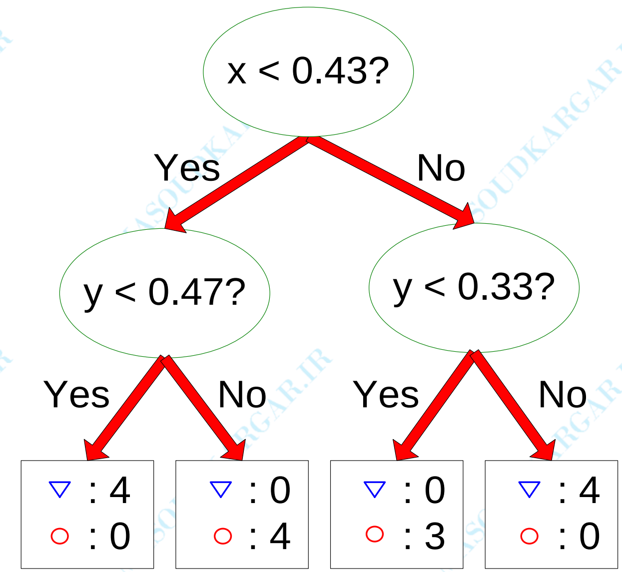
- Decision Tree Patents!
- More on Decision Tree
- Classifier Evaluation
- Overfitting
- Cross-validation
- Confidence of prediction accuracy

Decision Tree Patents

- Google Patent Search: http://www.google.com/patents
- **IBM2003**: **Method and system for building a decision-tree classifier from privacy**
- **SAS**: Method for selecting node variables in a binary decision tree structure
- **Sprint**: Method and system for dynamic variation of decision tree architecture
- **IBM2005:** Method for building space-splitting decision tree
- **Lucent2001:** Decision tree classifier with integrated building and pruning phases
- **Please read one of the patents and you should be able to understand and appreciate their innovation point.**

Decision Boundary: How Decision Tree works

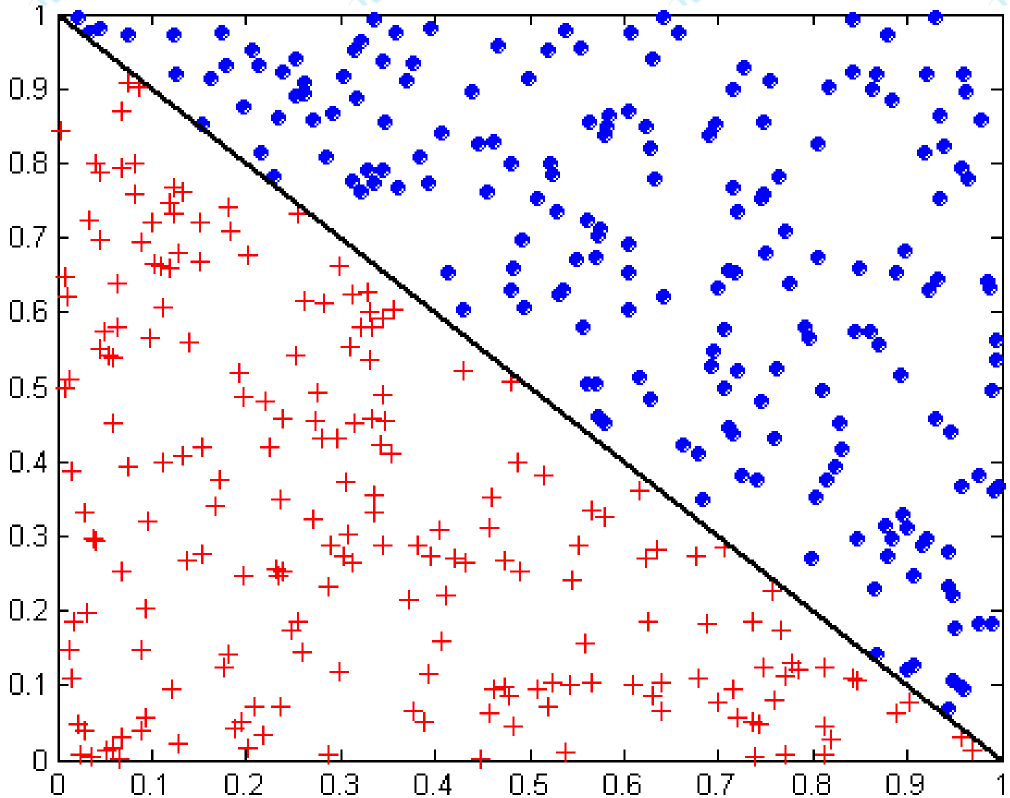


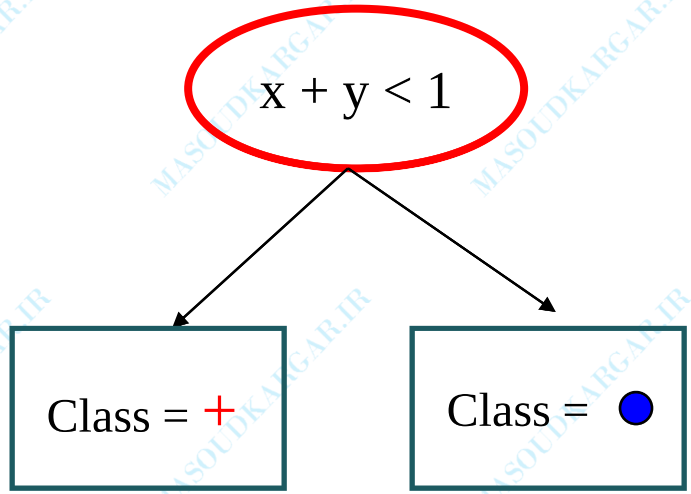


- Border line between two neighboring regions of different classes is known as decision boundary

- Decision boundary is parallel to axes because test condition involves a single attribute at-a-time

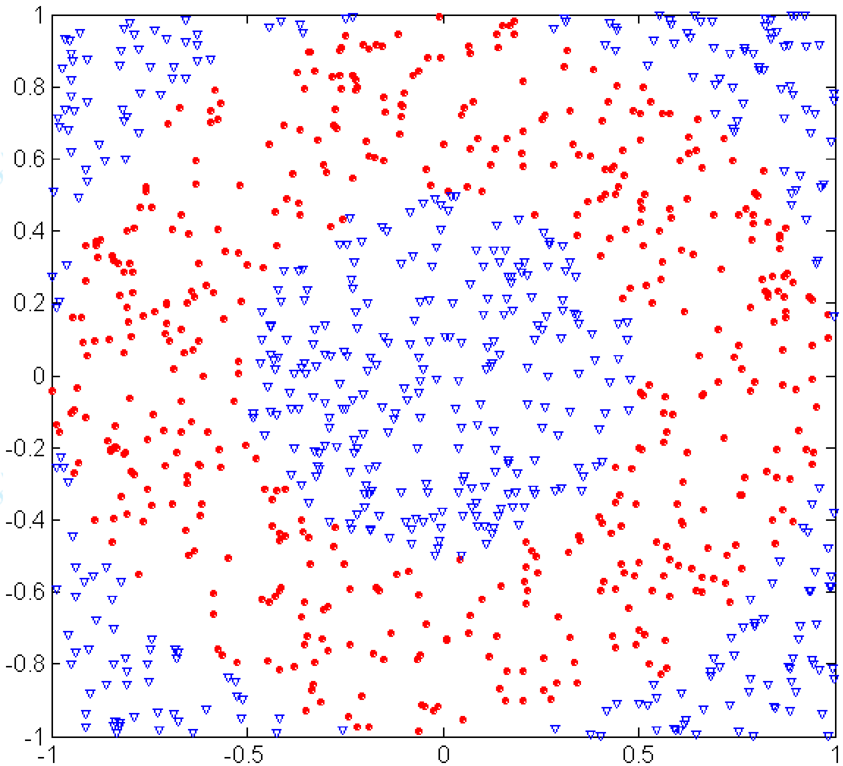
Oblique Decision Trees





- Test condition may involve multiple attributes
- More expressive representation
- Finding optimal test condition is computationally expensive

Limitation of Decision Tree Classifiers



500 circular and 500 triangular data points.

Circular points:

$0.5 \leq \text{sqrt}(x_1^2+x_2^2) \leq 1$

Triangular points:

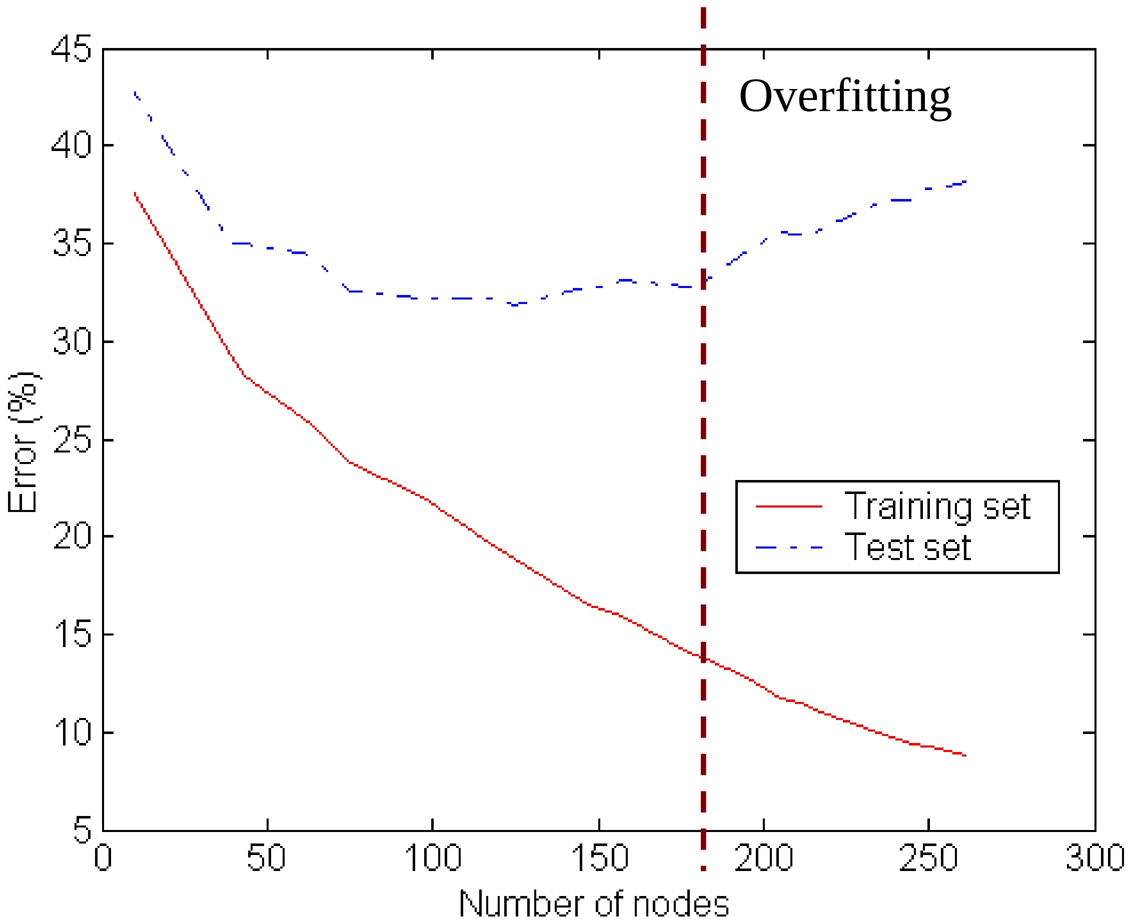
$\text{sqrt}(x_1^2+x_2^2) > 0.5$ or

$\text{sqrt}(x_1^2+x_2^2) < 1$

By using complex predicates, we can build complex decision tree to divide all training instances into pure subsets.

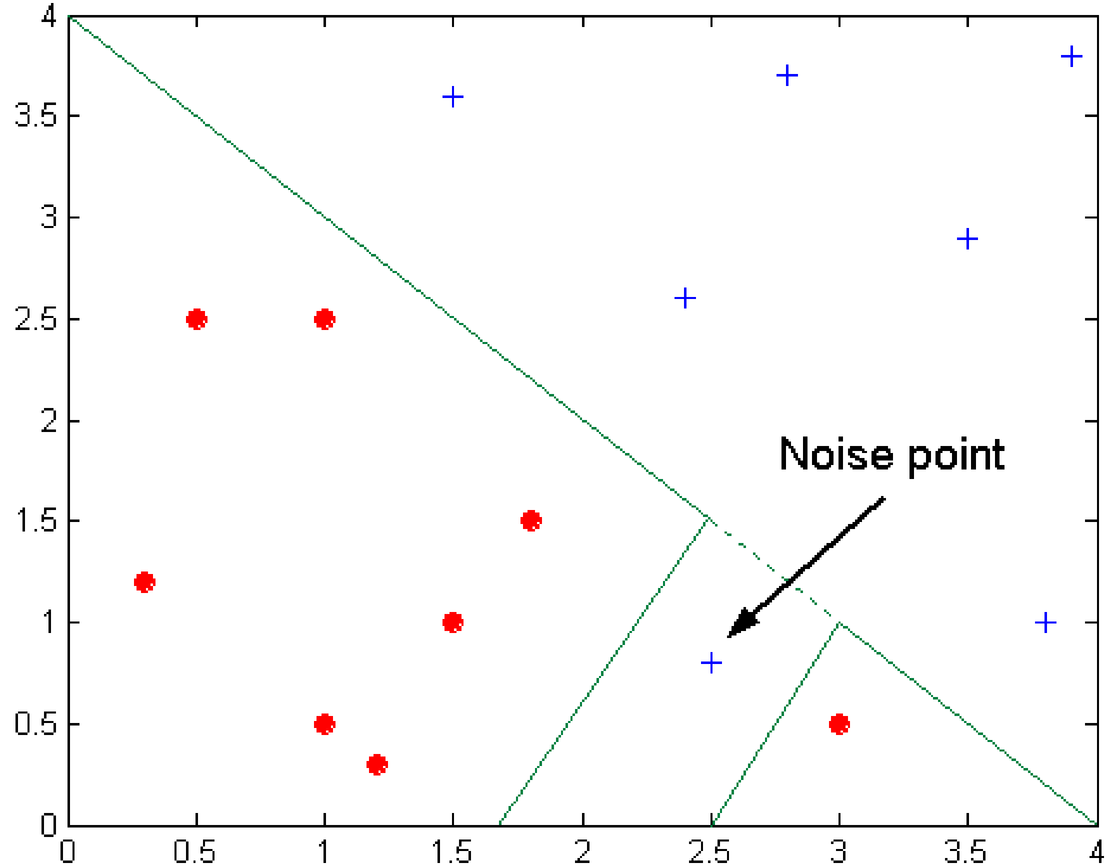
What is the consequences?

What is Overfitting?



Underfitting: when model is too simple, both training and test errors are large

Overfitting due to Noise



Decision boundary is distorted by noise point

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records
- Need new ways for estimating errors

How to Address Overfitting

- Pre-Pruning (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
  - Typical stopping conditions for a node:
    - Stop if all instances belong to the same class
    - Stop if all the attribute values are the same
  - More restrictive conditions:
    - Stop if number of instances is less than some user-specified threshold
    - Stop if class distribution of instances are independent of the available features (e.g., using χ^2 test)
    - Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain).

How to Address Overfitting…

- Post-pruning
  - – Grow decision tree to its entirety
  - – Trim the nodes of the decision tree in a bottom-up fashion
  - – If generalization error improves after trimming, replace sub-tree by a leaf node.
  - – Class label of leaf node is determined from majority class of instances in the sub-tree
  - – Can use MDL for post-pruning

Classification—Review Again

- **Model construction: describing a set of predetermined classes**
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The set of tuples used for model construction is training set
  - The model is represented as classification rules, decision trees, or mathematical formulae
- **Model Evaluation: Estimate accuracy of the model**
  - The known label of test sample is compared with the classified result from the model
  - Accuracy rate is the percentage of test set samples that are correctly classified by the model
  - Test set is independent of training set, otherwise over-fitting will occur
  - If the accuracy is acceptable, then
- **Model usage: use it to classify future or unknown objects**

Model Evaluation

- 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?

Accuracy: Good or Bad?

- Accuracy: $\dfrac{\#of\ correct\ predictions}{\#of\ total\ predictions}$

- You can easily get >99% accuracy (if 1positive 99 negative) using simplest KNN in Assignment 1.
- Should you be satisfied or not?   Why?
- Problem too easy?

Don’t Get Fooled by Ourselves

- In the training set of asuspect prediction Problem:
  - 64209 negative instances (non-suspect)
  - 651 positive instances (suspect)
- Without checking any attributes, a FOOL classifier can just predict any new person as non-suspect, Its classification accuracy on training set is:
  - 64209/(64209+651)=99%!

Handling Unbalanced Data

- Sometimes, classes have very unequal frequency
  - –Attrition prediction: 97% stay, 3% attrite (in a month)
  - –medical diagnosis: 90% healthy, 10% disease
  - –eCommerce: 99% don’t buy, 1% buy
  - –Security: >99.99% of Americans are not terrorists
- Similar situation with multiple classes
- Majority class classifier can achieve an accuracy of 97% or higher!

Balancing unbalanced data

- With two classes, a good approach is to build **BALANCED train and test sets, and train model on a balanced set**
  - randomly select desired number of minority class instances
  - add equal number of randomly selected majority class
- Generalize “balancing”to multiple classes
- Ensure that each class is represented with approximately equal proportions in train and test

- If accuracy is not a good measure,
- What would be a good performance measure?

Confusion Matrix: Seeking Better Performance Measures

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

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

a+b=651

c+d=64209

Metrics for Performance Evaluation…

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

- Most widely-used metric:

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

Better Measure of Prediction Performance

- *True positive* (TP): A tuple t_i predicted to be in class C_j, and is actually in it.
- *False positive* (FP): A tuple t_i predicted to be in class C_j, but is actually not in it.
- *True negative* (TN): A tuple t_i not predicted to be in class C_j, and is actually not in it.
- *False negative* (FN): A tuple t_i not predicted to be in class C_j, but is actually in it.
- The *precision* and *recall* are used to determine the accuracy of the classifier.

$$Precision = \frac{TP}{TP + FP} \qquad (1)$$

$$Recall = \frac{TP}{TP + FN} \qquad (2)$$

Sensitivity and Specificity

- **sensitivity** = probability of a positive test among patients with disease
- **specificity** = probability of a negative test among patients without disease

S $$\text{sensitivity} = \frac{\text{number of True Positives}}{\text{number of True Positives} + \text{number of False Negatives}}.$$

P $$\text{specificity} = \frac{\text{number of True Negatives}}{\text{number of True Negatives} + \text{number of False Positives}}$$

		Patients with bowel cancer (as confirmed on endoscopy)		
		True	*False*	?
FOB test	*Positive*	TP = 2	FP = 18	= TP / (TP + FP) = 2 / (2 + 18) = 2 / 20 ≡ **10%**
	Negative	FN = 1	TN = 182	= TN / (TN + FN) 182 / (1 + 182) = 182 / 183 ≡ **99.5%**
		↓ = TP / (TP + FN) = 2 / (2 + 1) = 2 / 3 ≡ **66.67%**	↓ = TN / (FP + TN) = 182 / (18 + 182) = 182 / 200 ≡ **91%**	

Precision

Recall/sensitivity specificity

What is the sensitivity and specificity of Your KNN classifier?

Cost Matrix

	PREDICTED CLASS	
ACTUAL CLASS		
C(i|j)	**Class=Yes**	**Class=No**
Class=Yes	C(Yes|Yes)	C(No|Yes)
Class=No	C(Yes|No)	C(No|No)

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

Computing Cost of Classification

Cost Matrix	PREDICTED CLASS		
ACTUAL CLASS	C(i|j)	+	-
	+	-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

Cost vs Accuracy

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

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

Accuracy is proportional to cost if
1. C(Yes|No)=C(No|Yes) = q
2. C(Yes|Yes)=C(No|No) = p

$N = a + b + c + d$

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

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

Cost-Sensitive Measures

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

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

$$\text{F-measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

- Precision is biased towards C(Yes|Yes) & C(Yes|No)
- Recall is biased towards C(Yes|Yes) & C(No|Yes)
- F-measure is biased towards all except C(No|No)

$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

Model Evaluation

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

- Methods for Performance Evaluation
  – How to obtain reliable estimates?

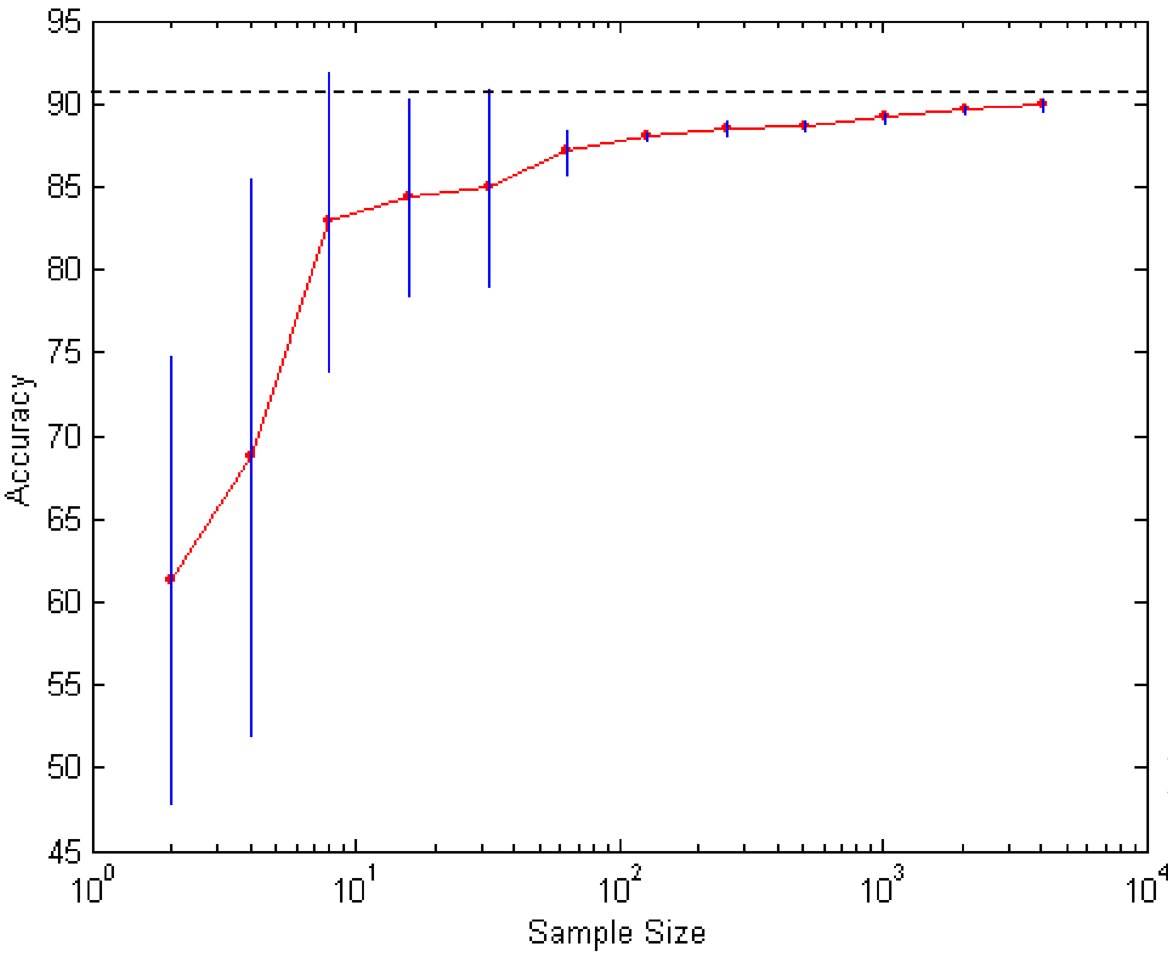
- Methods for Model Comparison
  – How to compare the relative performance among competing models?

Methods for Performance Evaluation

- 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: Accuracy w.r.t Size of Training Set



- Learning curve shows how accuracy changes with varying sample size
- Requires a sampling schedule for creating learning curve:
  - Arithmetic sampling (Langley, et al)
  - Geometric sampling (Provost et al)

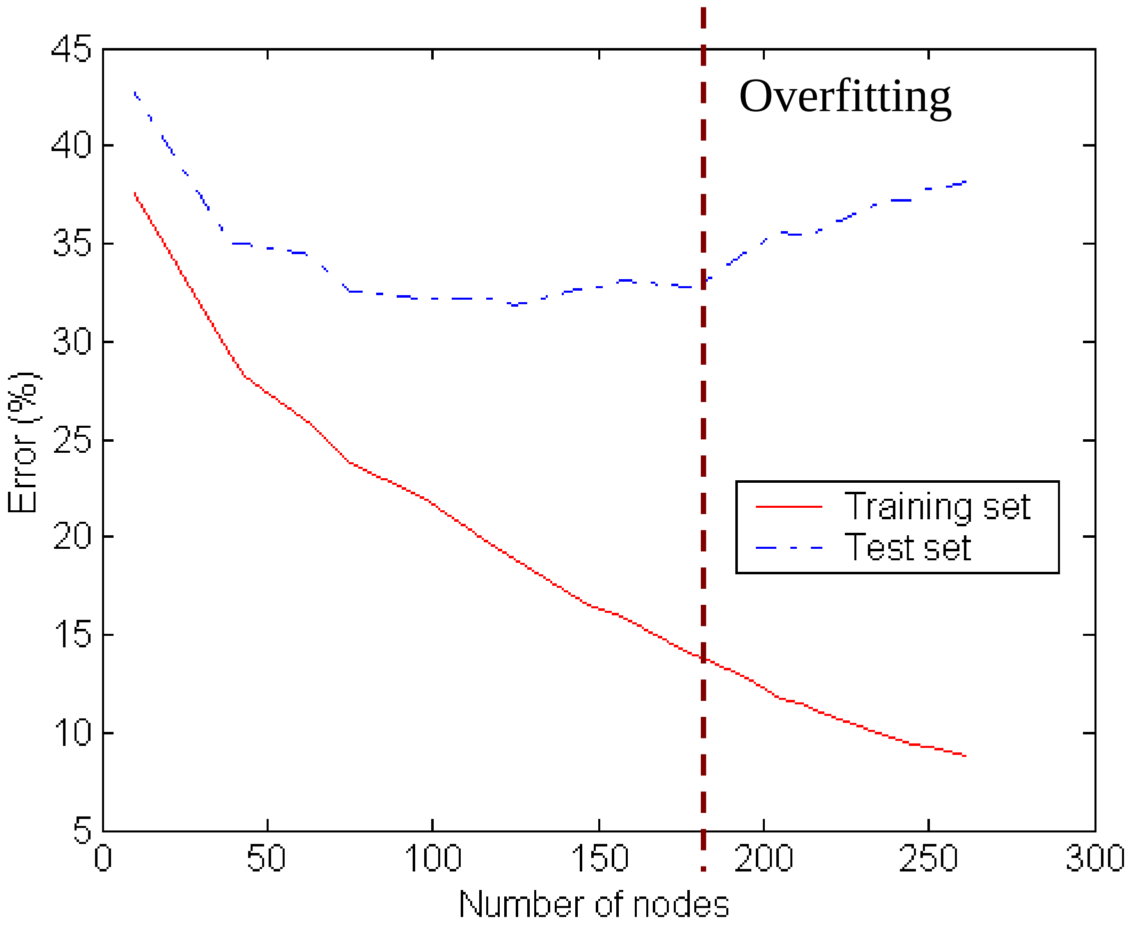
Effect of small sample size:

- Bias in the estimate
- Variance of estimate

Roadmap

- Assignment Issues!
- Decision Tree Patents!
- More on Decision Tree
- Classifier Evaluation
- Over-fitting
- Cross-validation
- Confidence of prediction accuracy

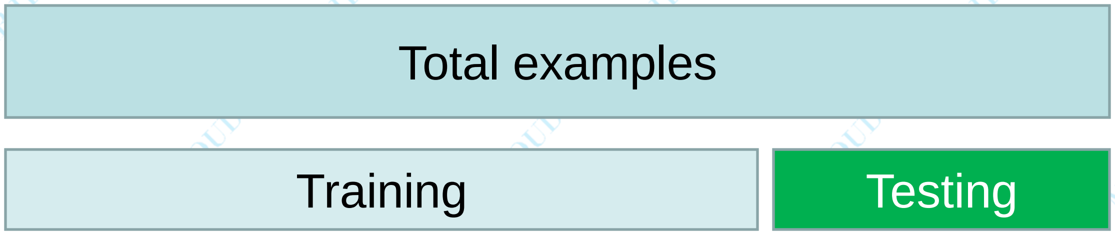
What is Overfitting?



Underfitting: when model is too simple, both training and test errors are large

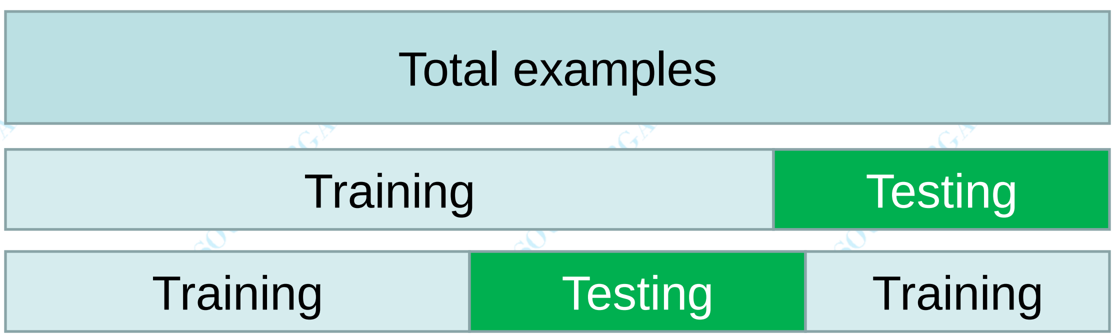
Methods for Evaluating Performance

- Holdout
  - Reserve 2/3 for training and 1/3 for testing



  - Wasting samples, Not good if sample size is small.

- Random subsampling
  - Repeated holdout



Methods for Evaluating Performance

- 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

Training	Testing	Training
Testing	Training	Training
Training	Training	Testing

- Bootstrap
  – Sampling with replacement

Model Evaluation

- 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?

ROC (Receiver Operating Characteristic)

- Developed in 1950s for signal detection theory to analyze noisy signals
  - Characterize the trade-off between positive hits and false alarms

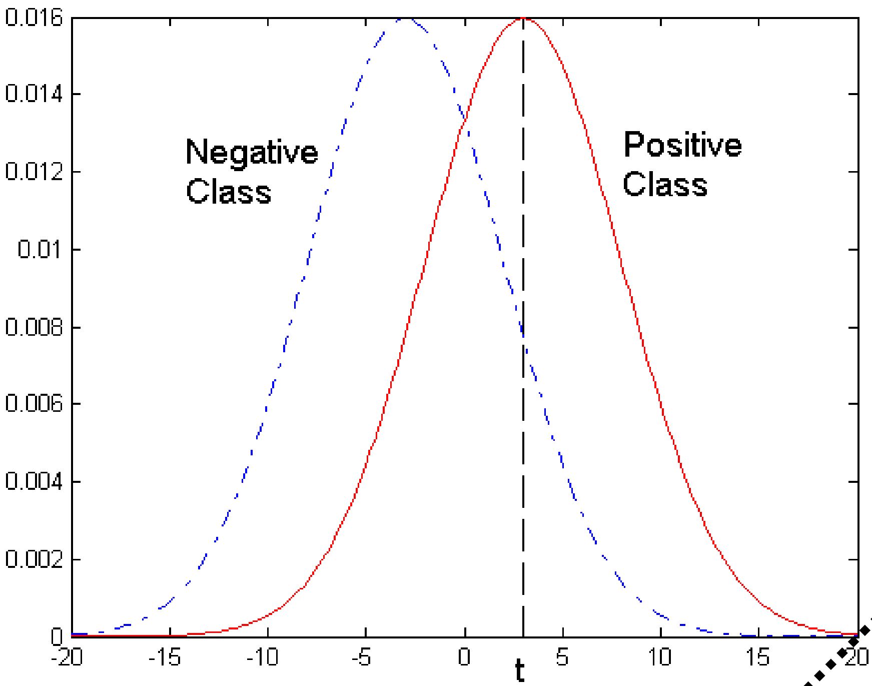
- ROC curve plots TP rate (on the y-axis) against FP rate (on the x-axis)

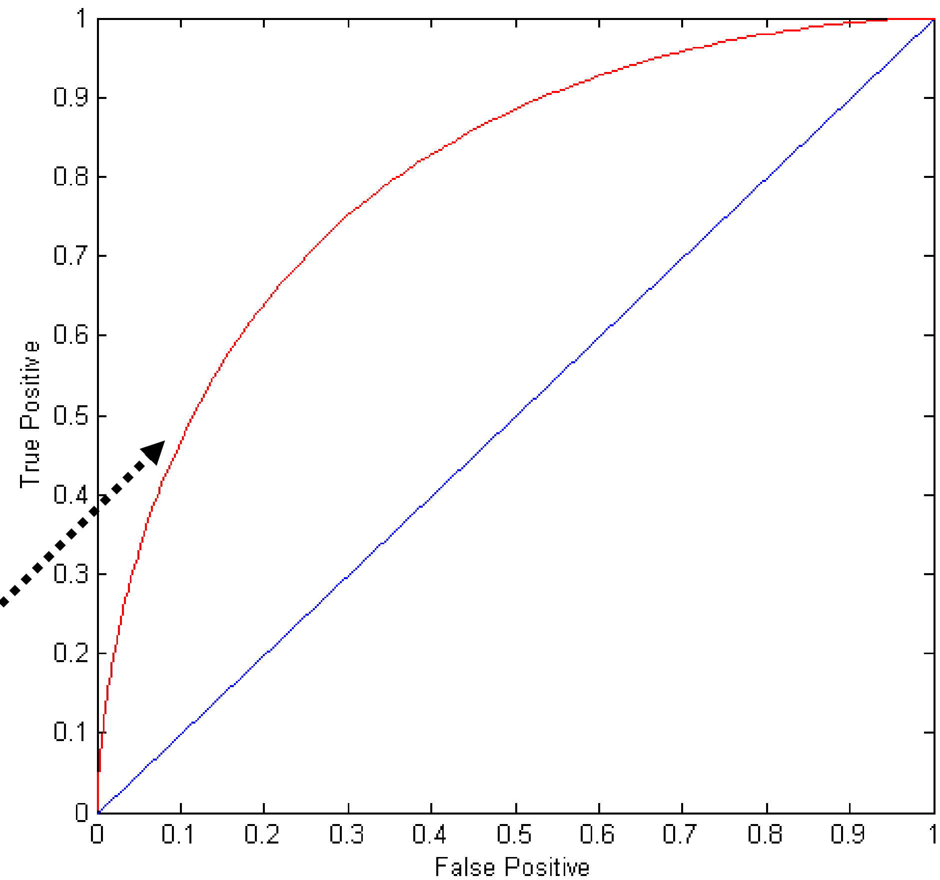
- Performance of each classifier represented as a point on the ROC curve
  - changing the threshold of algorithm, sample distribution or cost matrix changes the location of the point

ROC Curve

- 1-dimensional data set containing 2 classes (positive and negative)

- any points located at x > t is classified as positive





At threshold t:

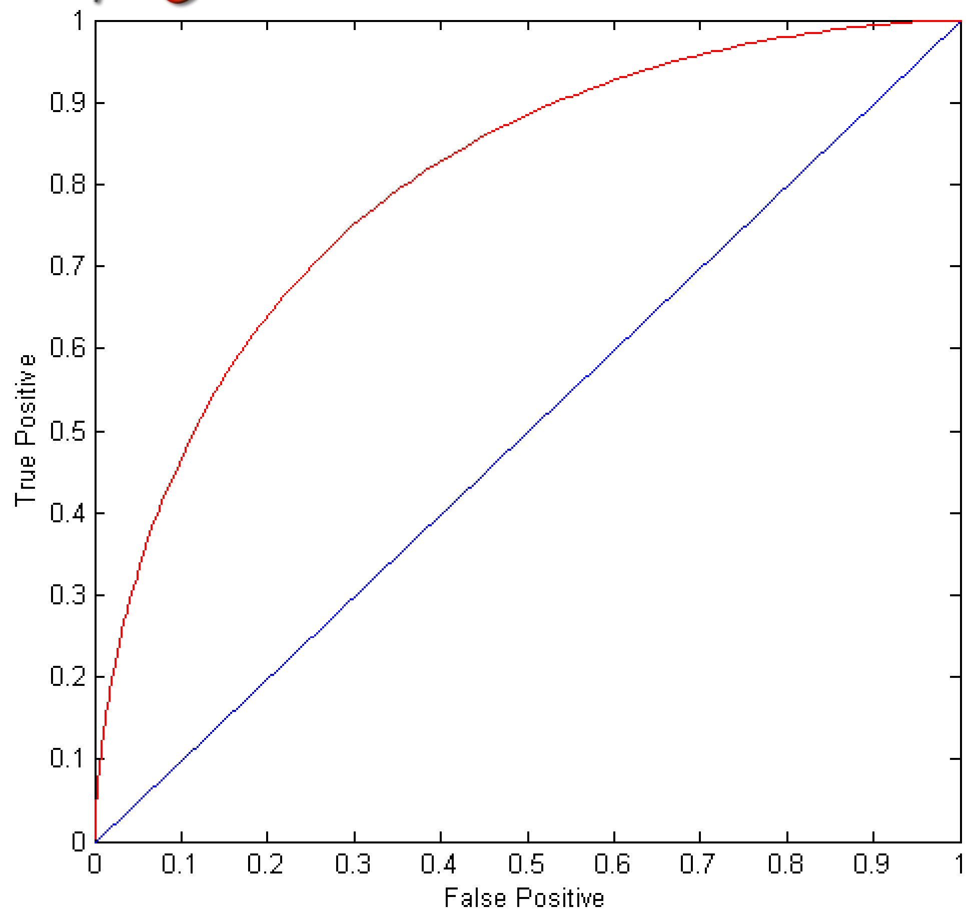
TP=0.5, FN=0.5, FP=0.12, FN=0.88

ROC Curve

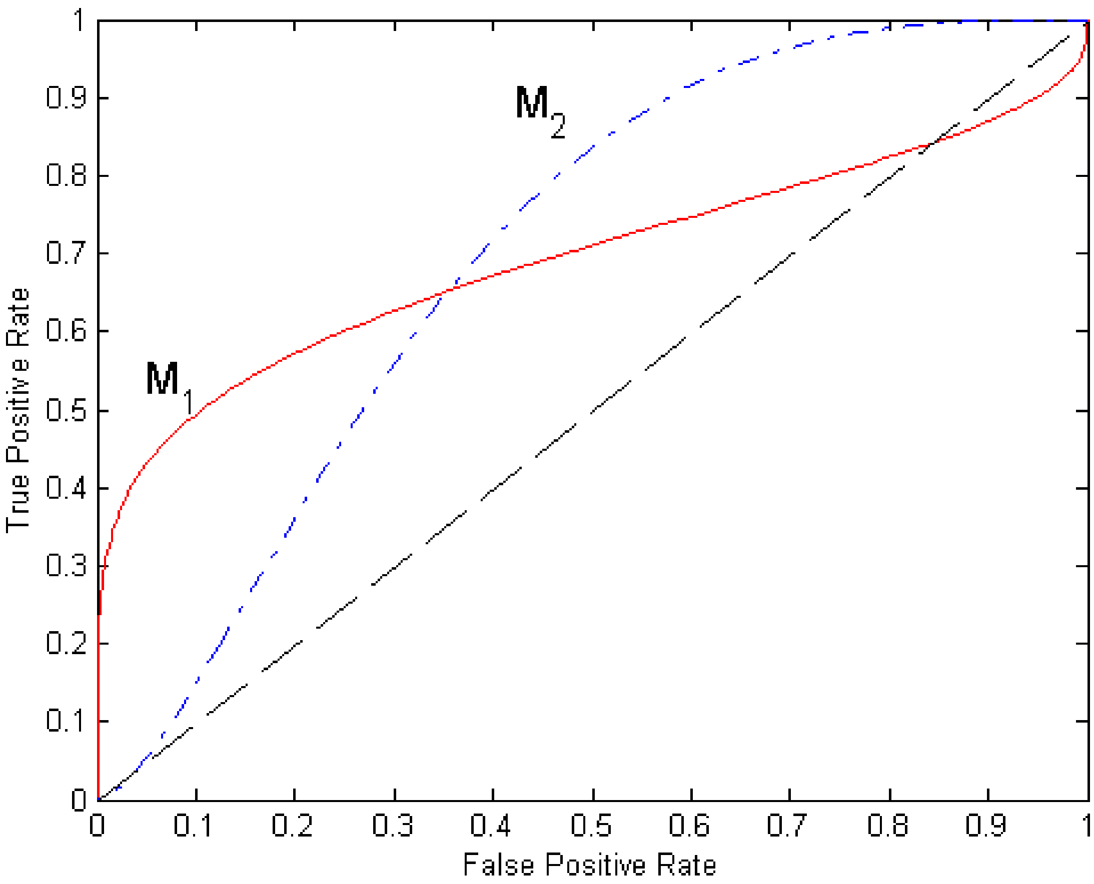
(TP,FP):

- (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 o
      the true class



Using ROC for Model Comparison



- 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

How to Construct an ROC curve

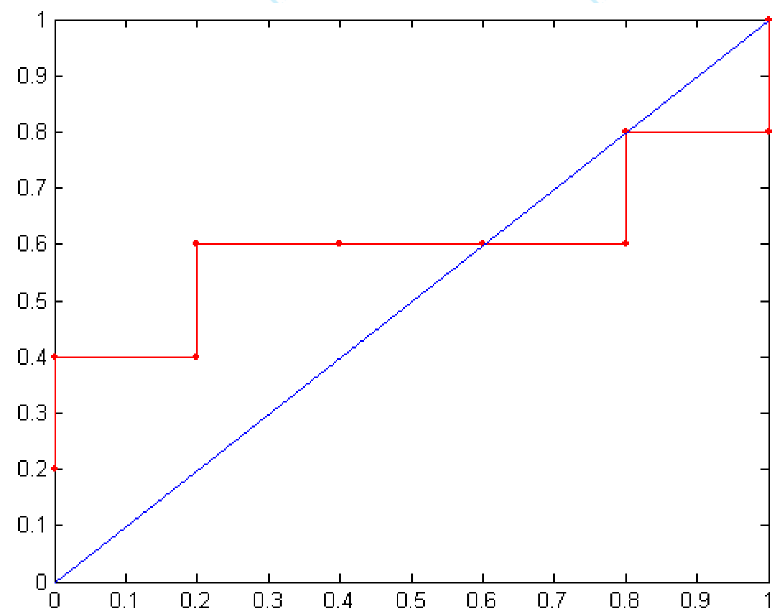
Instance	P(+|A)	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- 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)

How to construct an ROC curve

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:



Test of Significance

- 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?

Confidence Interval for Accuracy

- 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:
    - $x \sim \text{Bin}(N, p)$ x: number of correct predictions
    - e.g: Toss a fair coin 50 times, how many heads would turn up?
      Expected number of heads = $N \times p = 50 \times 0.5 = 25$

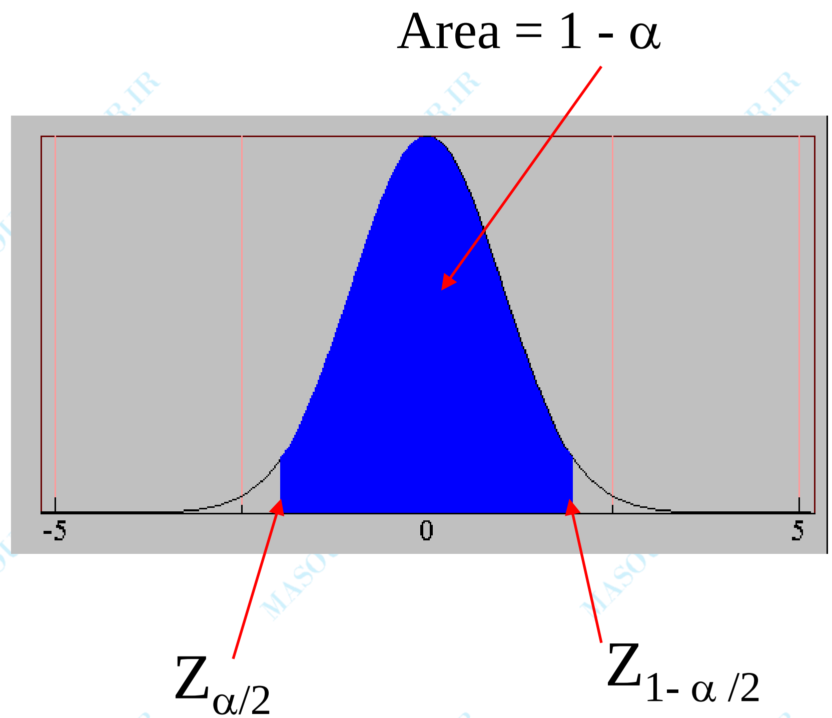
- Given x (# of correct predictions) or equivalently, acc=x/N, and N (# of test instances),

  Can we predict p (true accuracy of model)?

Confidence Interval for Accuracy

- For large test sets (N > 30),
  - acc has a normal distribution with mean p and variance p(1-p)/N

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



- Confidence Interval for p:

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

Confidence Interval for Accuracy

- 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

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

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

Comparing Performance of 2 Models

- Given two models, say M1 and M2, which is better?
  - M1 is tested on D1 (size=n1), found error rate = e_1
  - M2 is tested on D2 (size=n2), found error rate = e_2
  - Assume D1 and D2 are independent
  - If n1 and n2 are sufficiently large, then

$$e_1 \sim N(\mu_1, \sigma_1)$$
$$e_2 \sim N(\mu_2, \sigma_2)$$

  - Approximate:

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

Comparing Performance of 2 Models

- To test if performance difference is statistically significant: d = e1 – e2
  - d ~ *N*(d_t,σ_t)   where d_t is the true difference
  - Since D1 and D2 are independent, their variance adds up:

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

  - At (1-α) confidence level,

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

An Illustrative Example

- Given: M1: n1 = 30, e1 = 0.15
  M2: n2 = 5000, e2 = 0.25
- d = |e2 – e1| = 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

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

=> Interval contains 0 => difference may not be statistically significant

Comparing Performance of 2 Algorithms

- Each learning algorithm may produce k models:
  - L1 may produce M11 , M12, …, M1k
  - L2 may produce M21 , M22, …, M2k
- If models are generated on the same test sets D1,D2, …, Dk (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:

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

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

Summary

- Bad and good measurements of classification performance
- How to evaluate/estimate the measurements
- How to compare the performance of classifiers
- You will apply all these methods in Assignment 2, To be posted on Sunday.

قدردانی

- Dr. Jianjun Hu
  http://mleg.cse.sc.edu/edu/csce822/
- University of South Carolina
- Department of Computer Science and Engineering