Evaluation

Based on slides from Jude Shavlik and Tom Dietterich

Leave One Out?

- Have isaCourseWebPage data from CS Depts at Wisconsin, CMU, Cornell, and Texas (Craven et al, *AI* journal, 1999)
    - Leave out one UNIVERSITY
    - Assumes a new university will 'arrive tomorrow' to be analyzed
- Have advisedBy(Student,Professor) from AI, Graphics, PL, Systems, and Theory (Richardson & Domingos, *ML* journal, 2006)
    - Leave out one RESEARCH AREA
    - Assumes a new area will 'arrive tomorrow' to be analyzed
    - Could instead leave *N* professors and *M* students out of the TRAIN set when they are in the TEST set
- Or might be assuming a new protein, journal article, or gene-expression time series will arrive tomorrow

Contingency Tables

(special case of 'confusion matrices')

		True Answer	
		+	-
Algorithm Answer	+	n(1,1) [true pos]	n(1,0) [false pos]
	-	n(0,1) [false neg]	n(0,0) [true neg]

Counts of occurrences

TPR and FPR

True Positive Rate (TPR)
= n(1,1) / (n(1,1) + n(0,1))
= correctly categorized +'s / total positives
≅ P(algo outputs + | + is correct)

False Positive Rate (FPR)
= n(1,0) / (n(1,0) + n(0,0))
= incorrectly categorized –'s / total neg's
≅ P(algo outputs + | - is correct)

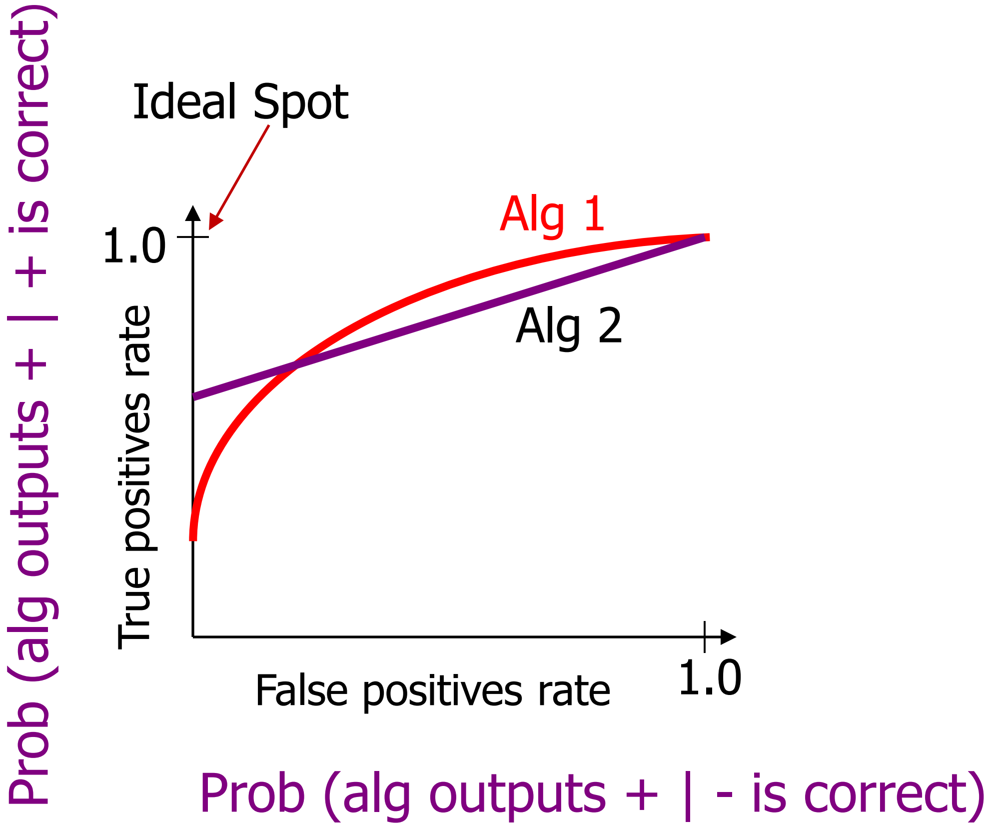
Can similarly define False Negative Rate and True Negative Rate

See http://en.wikipedia.org/wiki/Type_I_and_type_II_errors

ROC Curves

- ROC: *Receiver Operating Characteristics*
- Started for radar research during WWII
- Judging algorithms on accuracy alone may not be good enough when <u>getting a positive wrong costs</u> more than <u>getting a negative wrong</u> (or vice versa)
  - Eg, medical tests for serious diseases
  - Eg, a movie-recommender (ala' NetFlix) system

ROC Curves Graphically



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

Creating an ROC Curve - the Standard Approach

- You need an ML algorithm that outputs NUMERIC results such as prob(example is +)
- You can use ensembles (later) to get this from a model that only provides Boolean outputs

    Eg, have 100 models vote & count votes

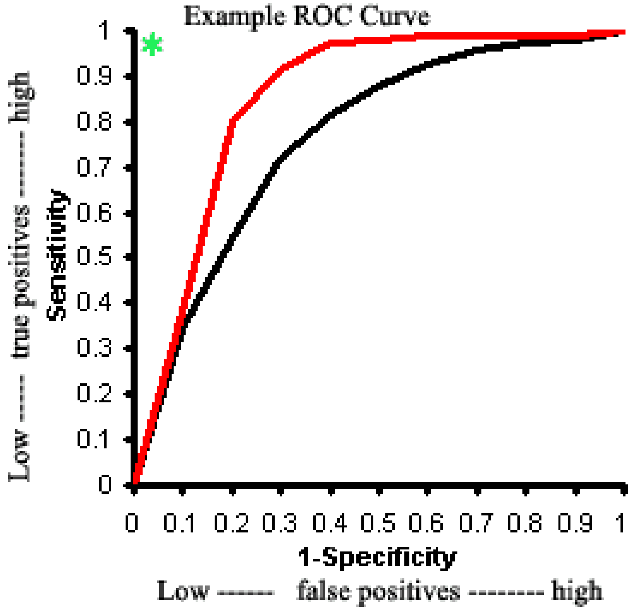
Algo for Creating ROC Curves

Step 1: Sort predictions on test set

Step 2: Locate a *threshold* between examples with opposite categories

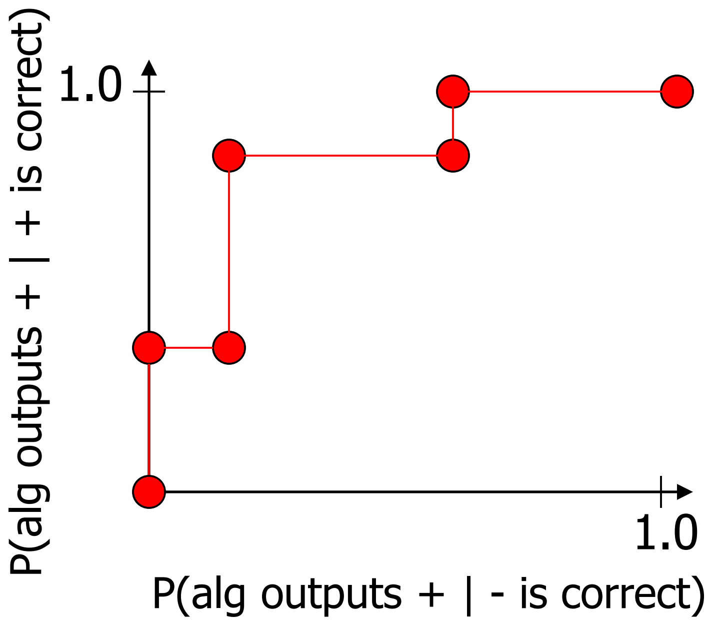
Step 3: Compute TPR & FPR for each threshold of Step 2

Step 4: Connect the dots



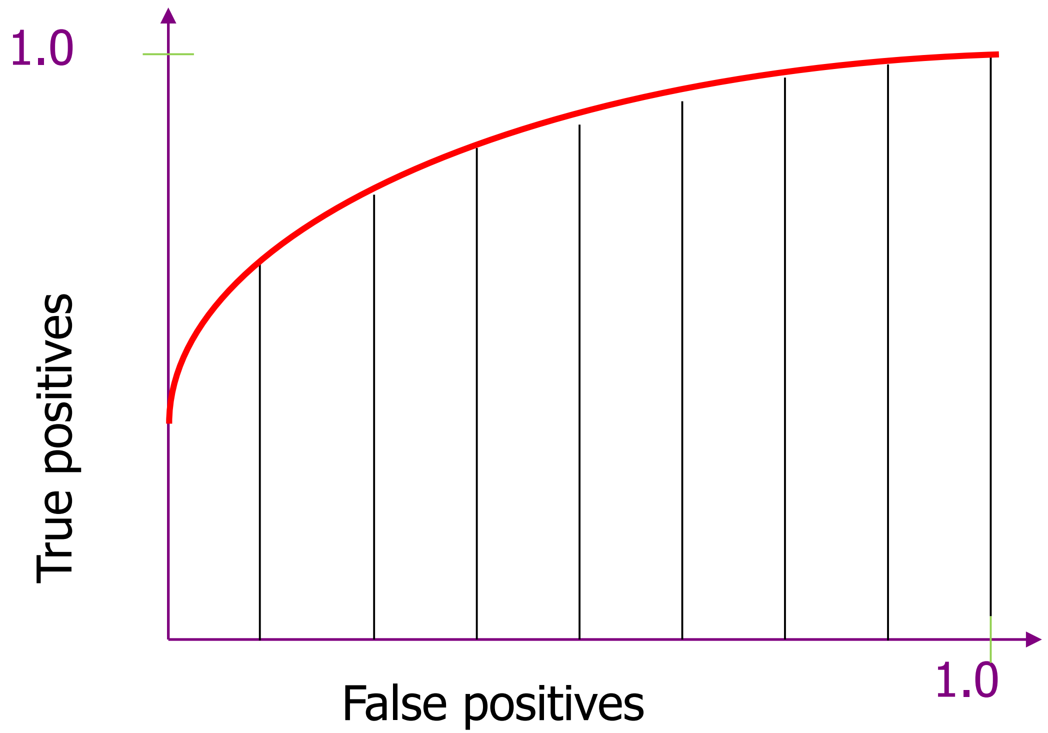
Plotting ROC Curves - Example

ML Algo Output (Sorted)			Correct Category
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)	-



Area Under ROC Curve

A common metric for experiments is to numerically integrate the ROC Curve



Asymmetric Error Costs

- Assume that cost(FP) ≠ cost(FN)
- You would like to pick a threshold that mimimizes

  E(total cost) =

  cost(FP) x prob(FP) x (# of neg ex’s) +

  cost(FN) x prob(FN) x (# of pos ex’s)
- You could also have (maybe negative) costs for TP and TN (assumed zero in above)

ROC's & Skewed Data

- One strength of ROC curves is that they are a good way to deal with skewed data
(|+| >> |-|) since the axes are fractions (rates) *independent* of the # of examples
- You must be careful though!
- Low FPR * (many negative ex)
= sizable number of FP
- Possibly more than # of TP

Precision vs. Recall

(think about search engines)

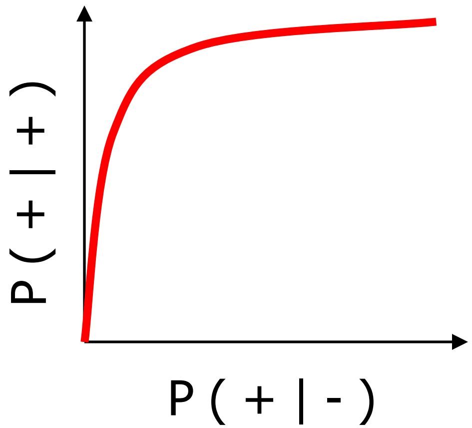
- **Precision** = (# of relevant items retrieved) / (total # of items retrieved)
  $= n(1,1) / (n(1,1) + n(1,0))$
  $\cong$ P(is pos | called pos)

- **Recall** = (# of relevant items retrieved) / (# of relevant items that exist)
  $= n(1,1) / (n(1,1) + n(0,1))$ = <u>TPR</u>
  $\cong$ P(called pos | is pos)

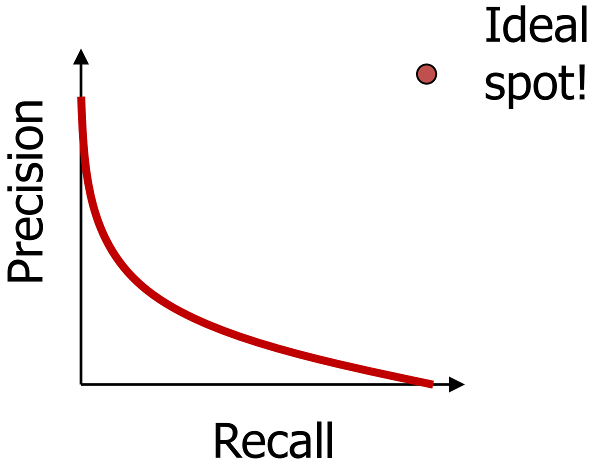
- Notice that n(0,0) is not used in either formula
  Therefore you get <u>no</u> credit for filtering out <u>ir</u>relevant items

ROC vs. Precision-Recall

You can get very different visual results on the same data!



vs.



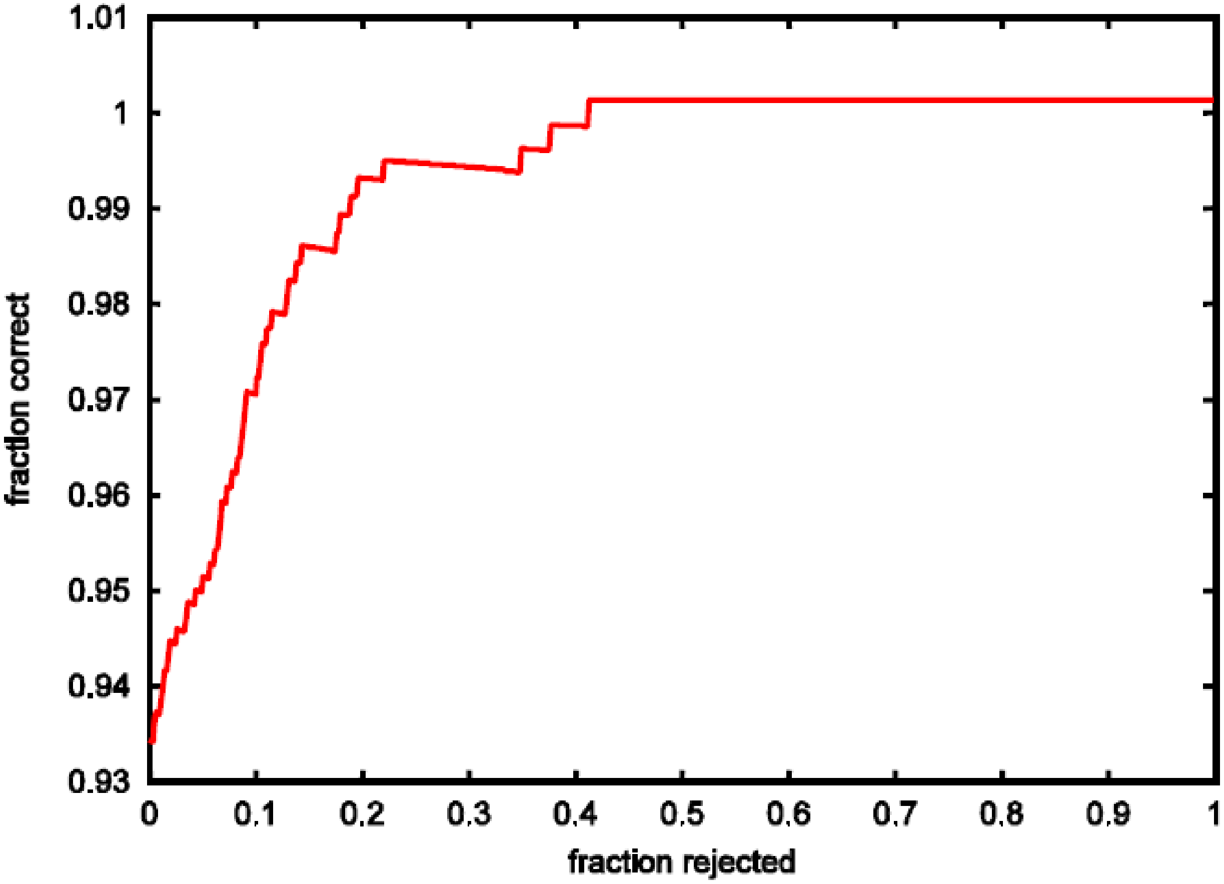
The reason for this is that there may be lots of – ex's
(eg, might need to include 100 neg's to get 1 more pos)

Rejection Curves

- In most learning algorithms, we can specify a threshold for making a rejection decision
  - Probabilistic classifiers: adjust cost of rejecting versus cost of FP and FN
  - Decision-boundary method: if a test point **x** is within θ of the decision boundary, then reject
- Equivalent to requiring that the “activation” of the best class is larger than the second-best class by at least θ

Rejection Curves

- Vary θ and plot fraction correct versus fraction rejected



The F1 Measure

- Figure of merit that combines precision and recall

$$F_1 = 2.\frac{P.R}{P+R}$$

where P = precision; R = recall. This is twice the harmonic mean of P and R.

- We can plot F1 as a function of the classification threshold θ

Summarizing a single operating point

- WEKA and many other systems normally report various measures for a single operating point (e.g., $\theta = 0.5$). Here is example output from WEKA

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.971	0.735	0.86	0.971	0.912	0.613	0
	0.265	0.029	0.667	0.265	0.379	0.783	1
W Avg.	0.846	0.61	0.825	0.846	0.817	0.643	

One more method

- Goal: decide which of two classifiers *h*1 and *h*2 has lower error rate
- Method: Run them both on the same test data set and record the following information:
  - n11: the number of examples correctly classified by both classifiers
  - n*0*1: the number of examples correctly classified by *h*1 but misclassified by *h*2
  - n10: The number of examples misclassified by *h*1 but correctly classified by *h*2
  - n00: The number of examples misclassified by both *h*1 and *h*2.

n00	n01
n10	n11

McNemar's test

$$M = \frac{(|n01 - n10| - 1)^2}{n01 + n10} > \psi^2_{1,\alpha}$$

- M is distributed approximately as χ2 with 1 degree of freedom. For a 95% confidence test, $\chi 2_{1,095} = 3.84$. So if M is larger than 3.84, then with 95% confidence, we can reject the null hypothesis that the two classifies have the same error rate

Permutation Tests

- Another way to judge significance of an empirical result
- This is just starting to appear in a few ML papers, but is an old idea in stats community
- Method (*one* way to use permutation tests)

  Multiple times

  1) permute the class labels of train and tune sets
  2) train
  3) evaluate on the (unpermuted) test sets
- See how likely it is that you get as good or better results on random outputs

  Ie, plot distribution of accuracy on permuted data,
  see where algo's results on unpermuted data lie