ROC Analysis for Ranking and Probability Estimation

Peter A. Flach
University of Bristol, UK
www.cs.bris.ac.uk/~flach/

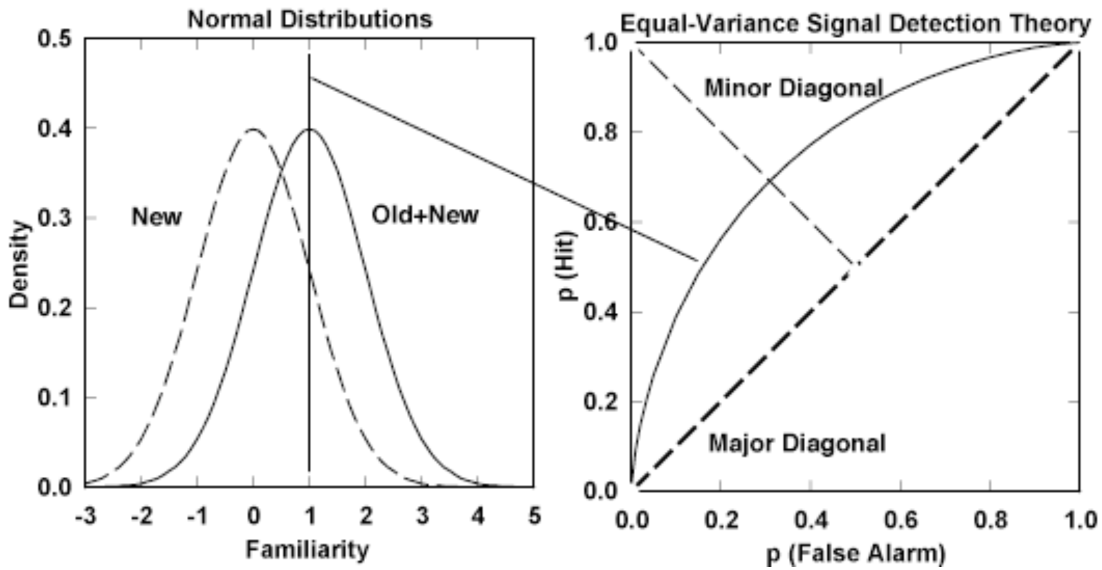
Machine Learning and Biological Computation Group
Department of Computer Science
University of Bristol

Outline

- classification: ROC plots, the ROC convex hull, iso-accuracy lines
- ranking: ROC curves, the AUC metric, turning rankers into classifiers
- probability estimation: probability estimates from ROC curves, calibration
- model manipulation: new models without re-training, ordering decision tree branches and rules, locally adjusting rankings
- more than two classes: multi-objective optimisation and the Pareto front, approximations

Receiver Operating Characteristic

- Originated from signal detection theory
  - binary signal corrupted by Gaussian noise
  - how to set the threshold (operating point) to distinguish between presence/absence of signal?
  - depends on (1) strength of signal, (2) noise variance, and (3) desired hit rate or false alarm rate



from http://wise.cgu.edu/sdt/

Signal detection theory

- slope of ROC curve is equal to likelihood ratio

$$L(x) = \frac{P(x \mid \text{signal})}{P(x \mid \text{noise})}$$

- if variances are equal, L(x) increases monotonically with x and ROC curve is convex
  - optimal threshold for x_0 such that $L(x_0) = \frac{P(\text{noise})}{P(\text{signal})}$

- concavities occur with unequal variances

ROC analysis for classification

- Based on contingency table or confusion matrix

	Predicted positive	Predicted negative	
Positive examples	**True positives**	**False negatives**	
Negative examples	**False positives**	**True negatives**	

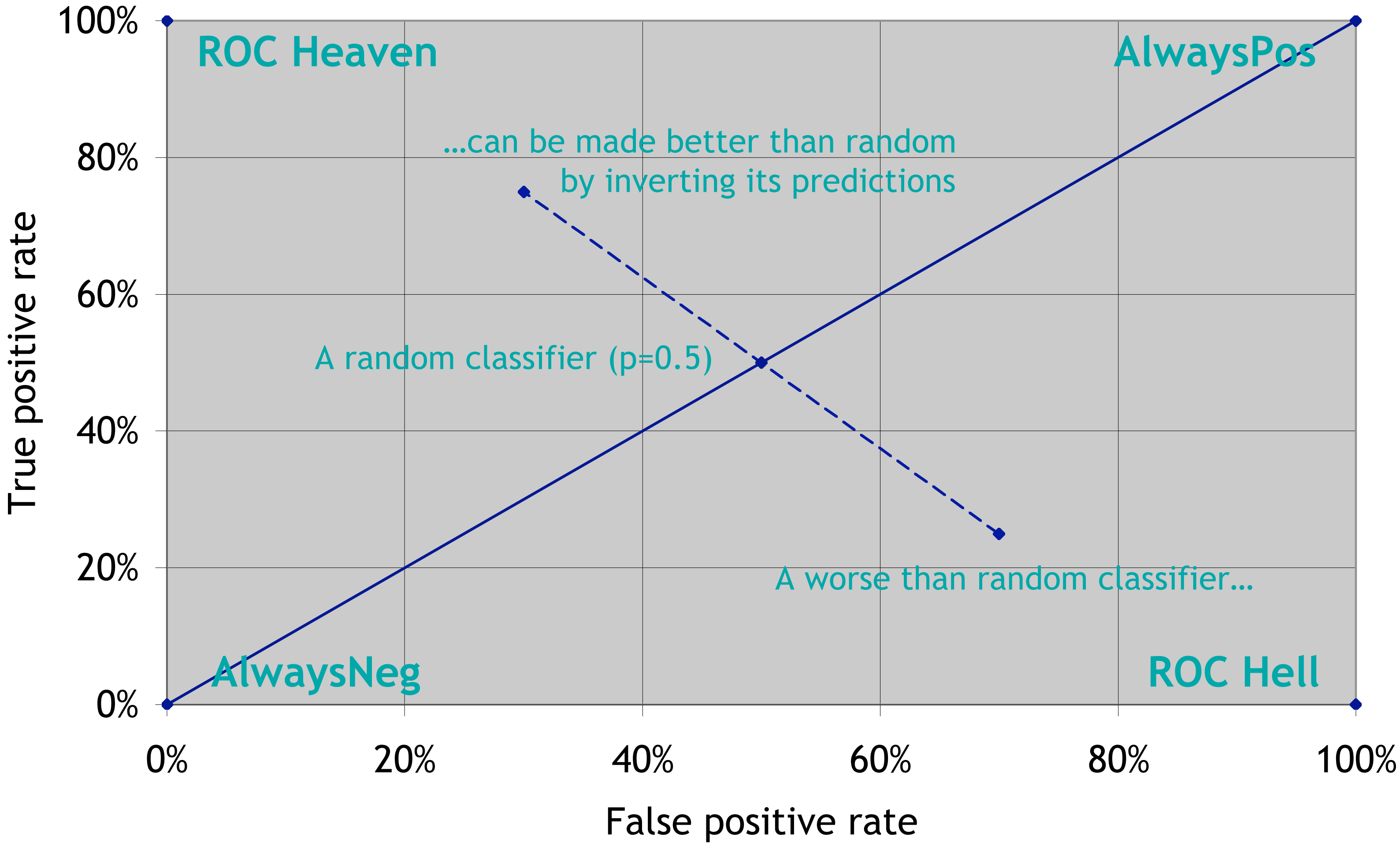
- Terminology:
  - true positive = hit
  - true negative = correct rejection
  - false positive = false alarm (aka Type I error)
  - false negative = miss (aka Type II error)
    - positive/negative refers to prediction
    - true/false refers to correctness

More terminology & notation

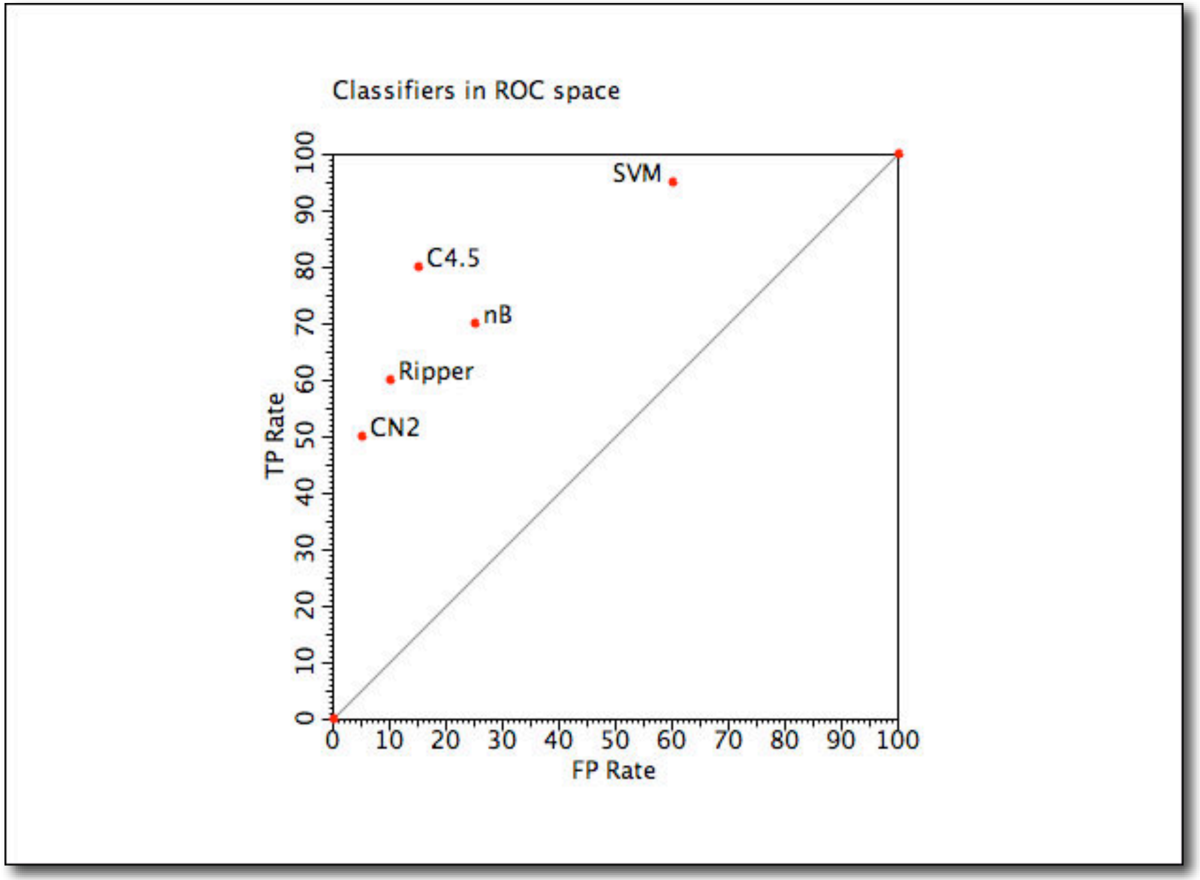
	Predicted positive	Predicted negative	
Positive examples	**TP**	**FN**	**Pos**
Negative examples	**FP**	**TN**	**Neg**
	PPos	**PNeg**	**N**

- True positive rate tpr = TP/Pos = TP/TP+FN
  - fraction of positives correctly predicted
- False positive rate fpr = FP/Neg = FP/FP+TN
  - fraction of negatives incorrectly predicted
  - = 1 – true negative rate TN/FP+TN
- Accuracy acc = pos*tpr + neg*(1–fpr)
  - weighted average of true positive and true negative rates

A closer look at ROC space

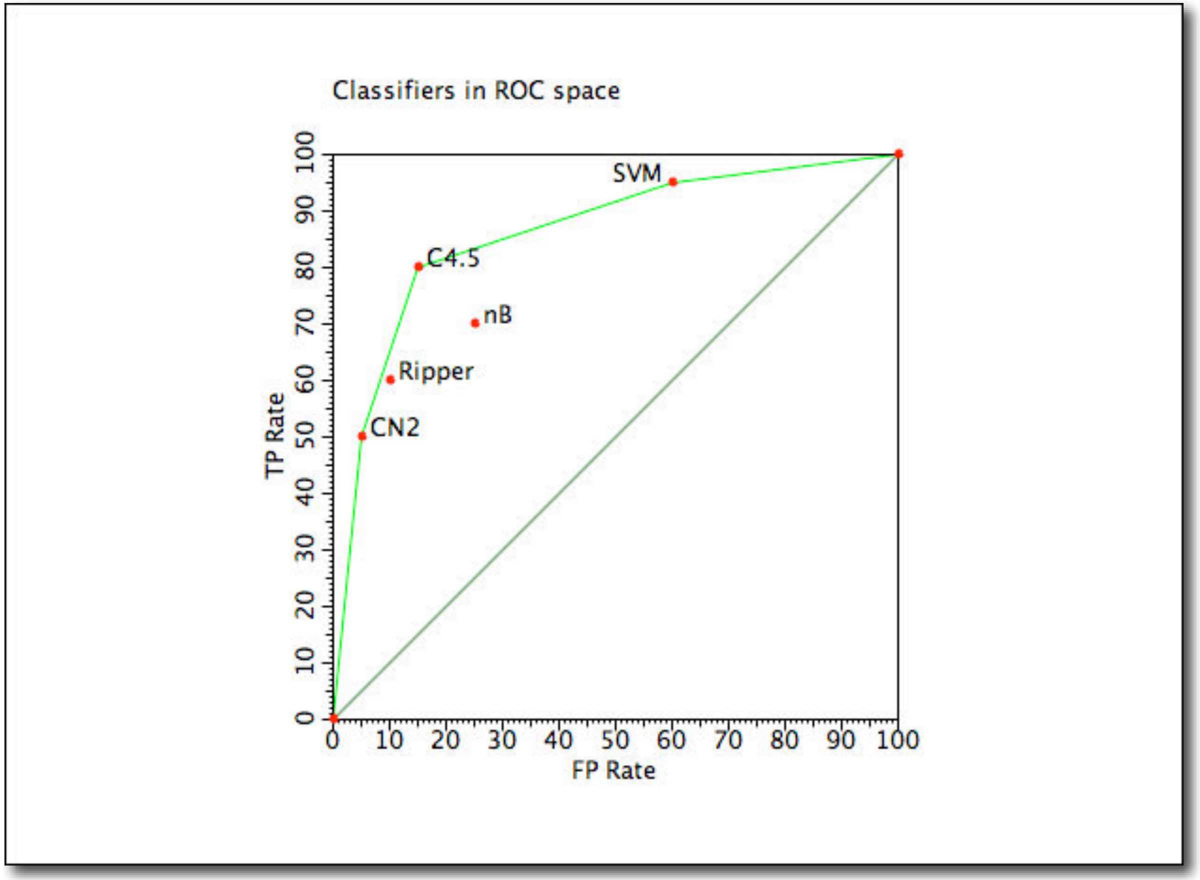


Example ROC plot



ROC plot produced by ROCon (http://www.cs.bris.ac.uk/Research/MachineLearning/rocon/)

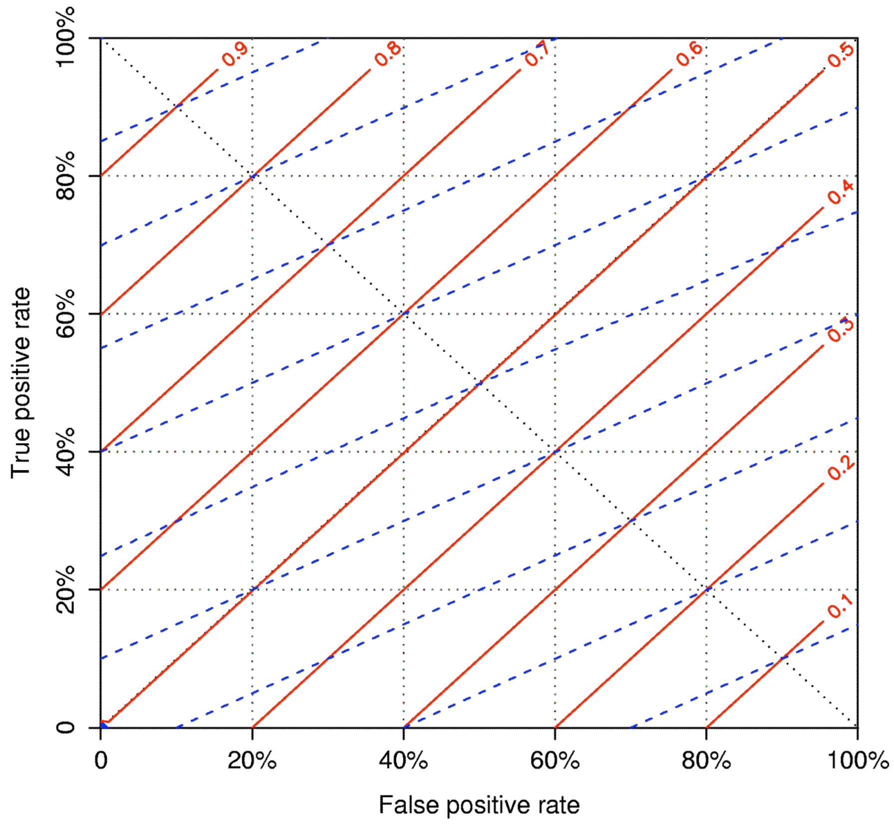
The ROC convex hull



- Classifiers on the convex hull achieve the best accuracy for some class distributions
- Classifiers below the convex hull are always sub-optimal

Iso-accuracy lines

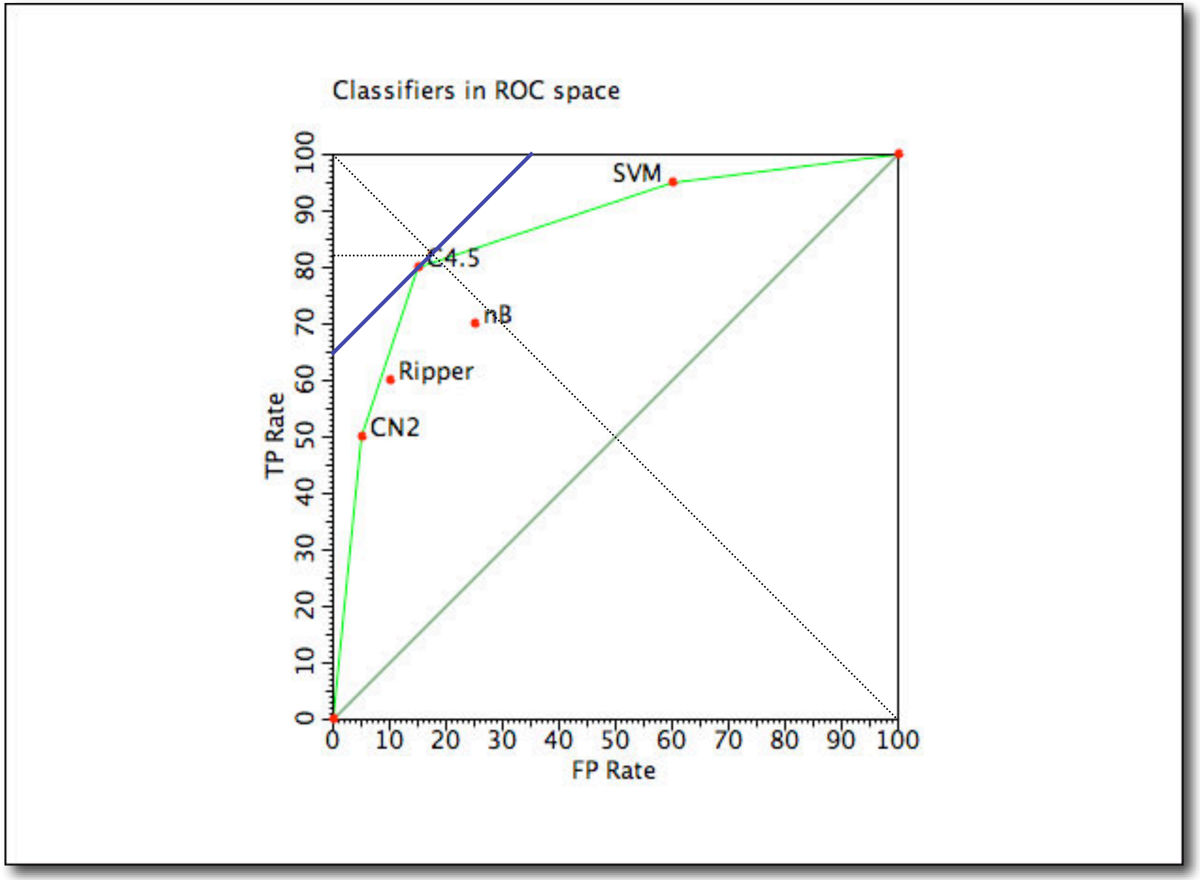
- Iso-accuracy line connects ROC points with the same accuracy
  - $pos*tpr + neg*(1-fpr) = a$
  - $tpr = \frac{a - neg}{pos} + \frac{neg}{pos} fpr$
- Parallel ascending lines with slope neg/pos
  - higher lines are better
  - on descending diagonal, $tpr = a$



Iso-accuracy & convex hull

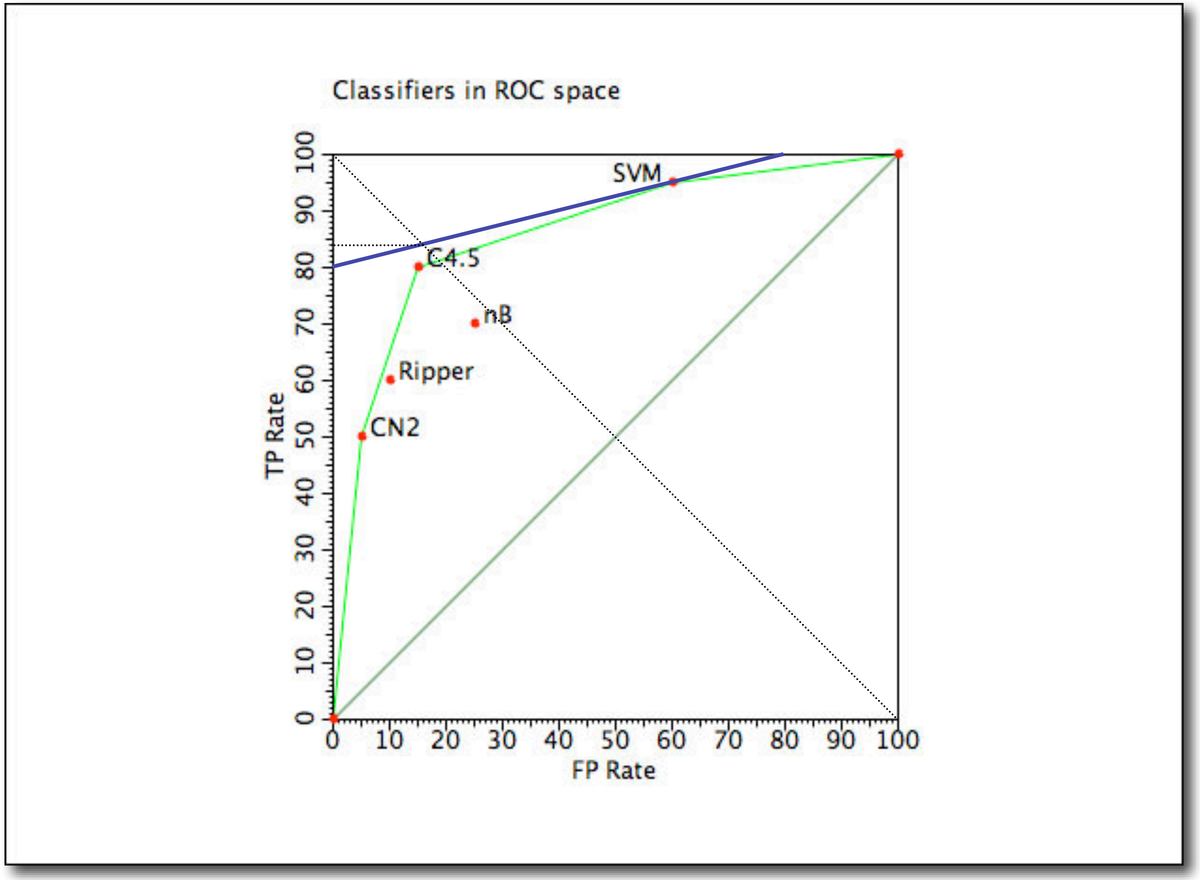
- Each line segment on the convex hull is an iso-accuracy line for a particular class distribution
  - under that distribution, the two classifiers on the end-points achieve the same accuracy
  - for distributions skewed towards negatives (steeper slope), the left one is better
  - for distributions skewed towards positives (flatter slope), the right one is better
- Each classifier on convex hull is optimal for a specific range of class distributions

Selecting the optimal classifier



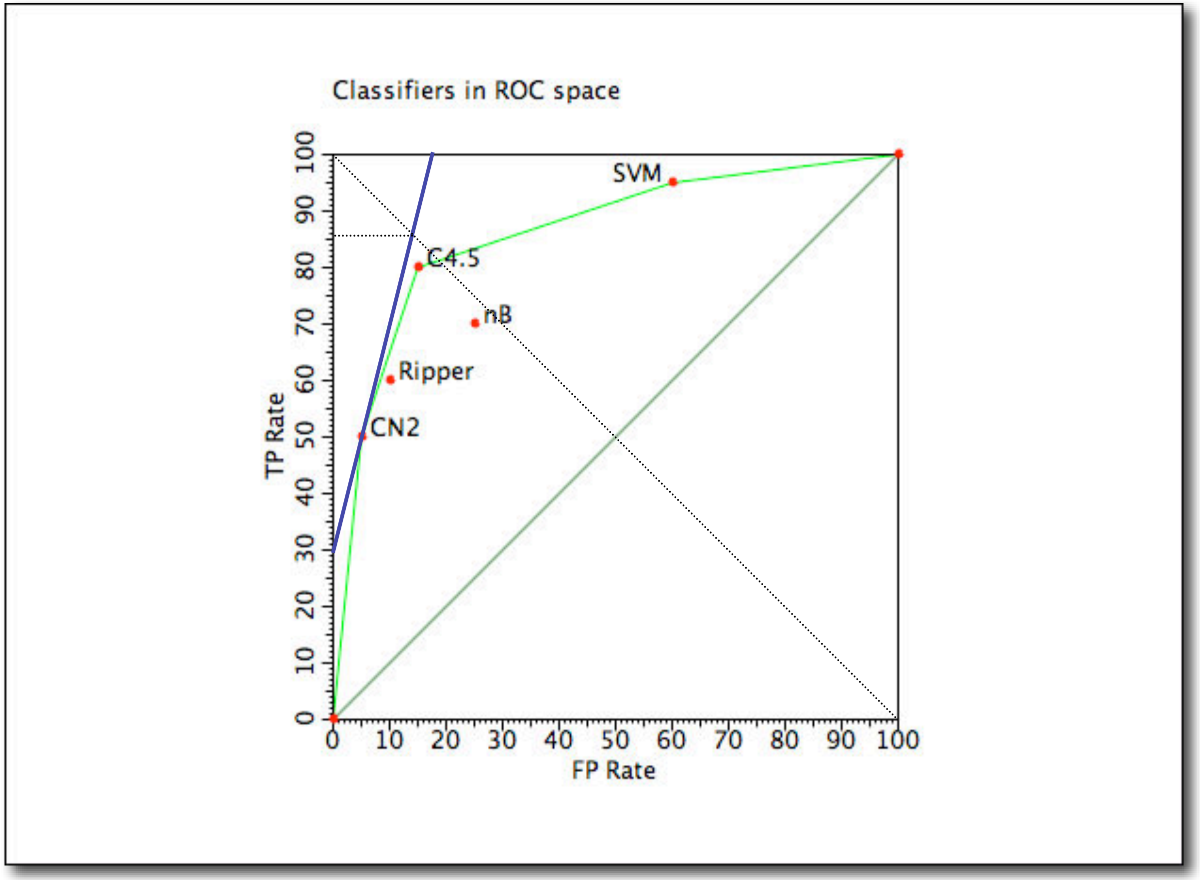
- For uniform class distribution, C4.5 is optimal
  - and achieves about 82% accuracy

Selecting the optimal classifier



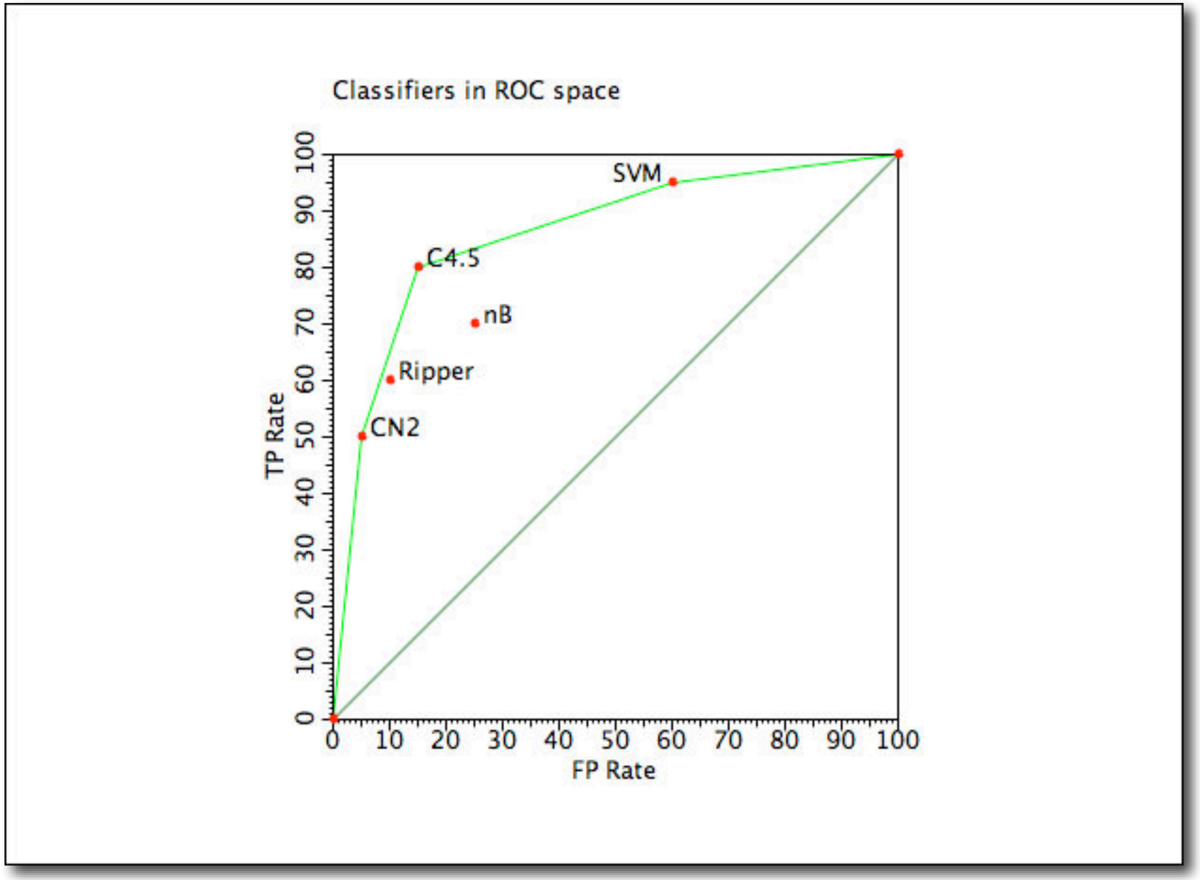
- With four times as many +ves as –ves, SVM is optimal
  - and achieves about 84% accuracy

Selecting the optimal classifier



- With four times as many –ves as +ves, CN2 is optimal
  - and achieves about 86% accuracy

Selecting the optimal classifier



- With less than 9% positives, AlwaysNeg is optimal
- With less than 11% negatives, AlwaysPos is optimal

Incorporating costs and profits

- Iso-accuracy and iso-error lines are the same
  - err = pos*(1–tpr) + neg*fpr
  - slope of iso-error line is neg/pos
- Incorporating misclassification costs:
  - cost = pos*(1–tpr)*C(-|+) + neg*fpr*C(+|-)
  - slope of iso-cost line is neg*C(+|–)/pos*C(–|+)
- Incorporating correct classification profits:
  - cost = pos*(1–tpr)*C(-|+) + neg*fpr*C(+|-) + pos*tpr*C(+|+) + neg*(1–fpr)*C(-|-)
  - slope of iso-yield line is neg*[C(+|-)-C(-|-)]/pos*[C(-|+)-C(+|+)]

Skew

- From a decision-making perspective, the cost matrix has one degree of freedom
  - need full cost matrix to determine absolute yield
- There is no reason to distinguish between cost skew and class skew
  - skew ratio expresses relative importance of negatives vs. positives
- ROC analysis deals with skew-sensitivity rather than cost-sensitivity

Rankers and classifiers

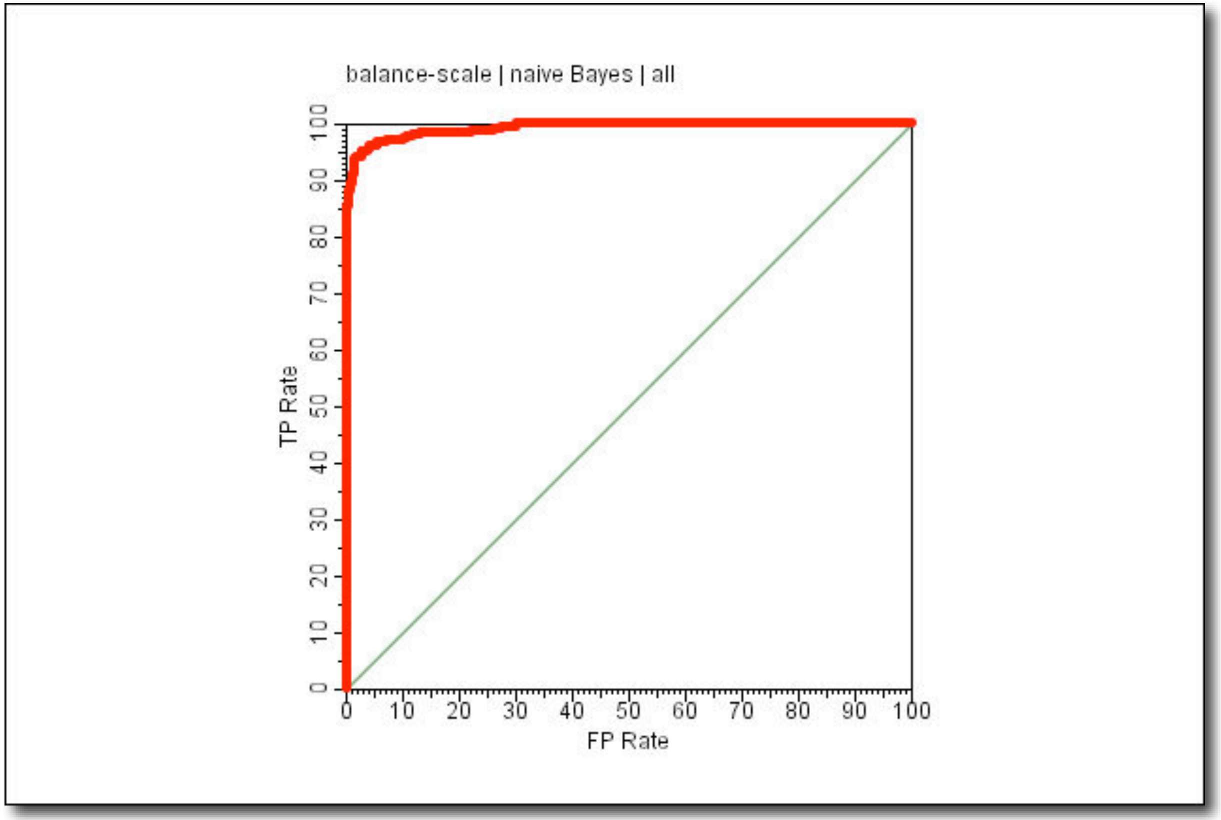
- A scoring classifier outputs scores f(x,+) and f(x,–) for each class
  - e.g. posterior P(+|x) and P(–|x), or likelihoods P(x|+) and P(x|-)
  - scores don't need to be normalised
- f(x) = f(x,+)/f(x,–) can be used to rank instances from most to least likely positive
  - e.g. posterior odds P(+|x)/P(–|x), or likelihood ratio P(x|+)/P(x|–)
- Rankers can be turned into classifiers by setting a threshold on f(x)

Drawing ROC curves for rankers

- Naïve method:
  - consider all possible thresholds
    - in fact, only k+1 for k instances
  - construct contingency table for each threshold
  - plot in ROC space

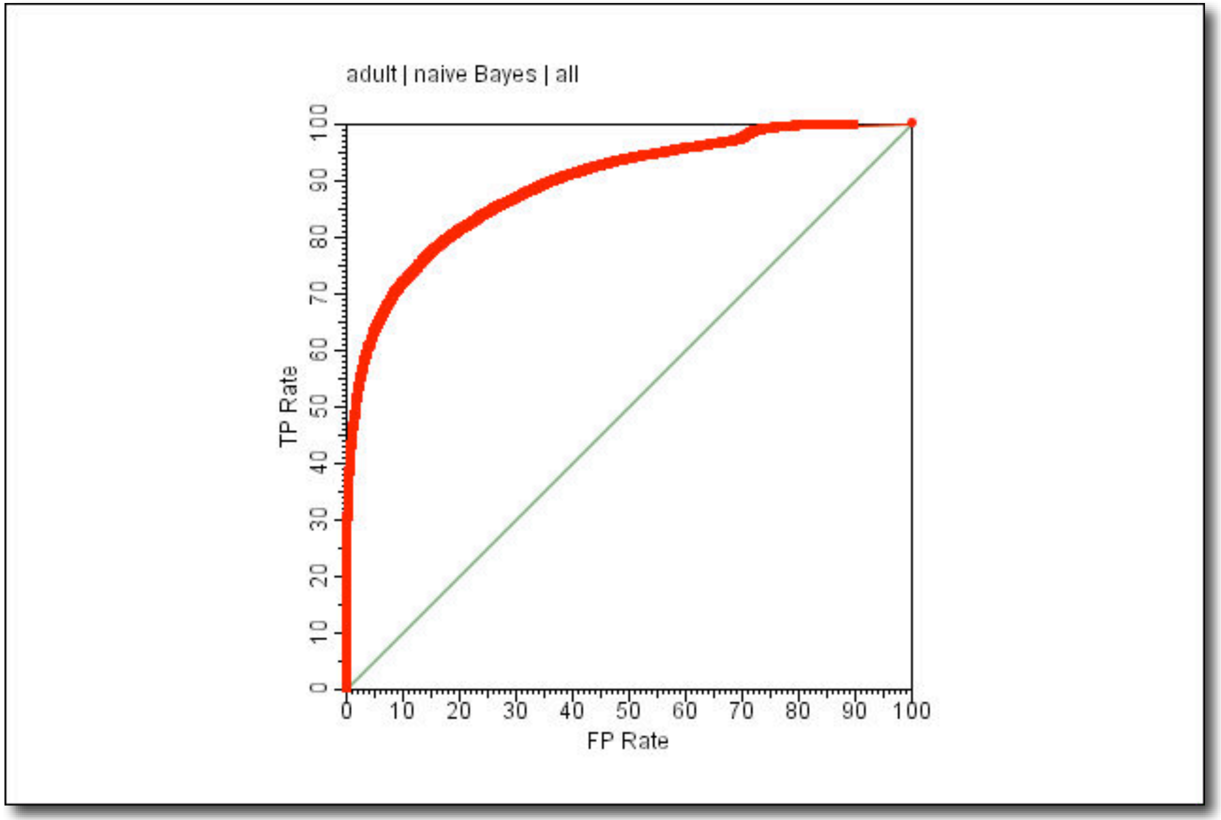
- Practical method:
  - rank test instances on decreasing score f(x)
  - starting in (0,0), if the next instance in the ranking is +ve move 1/Pos up, if it is –ve move 1/Neg to the right
    - make diagonal move in case of ties

Some example ROC curves



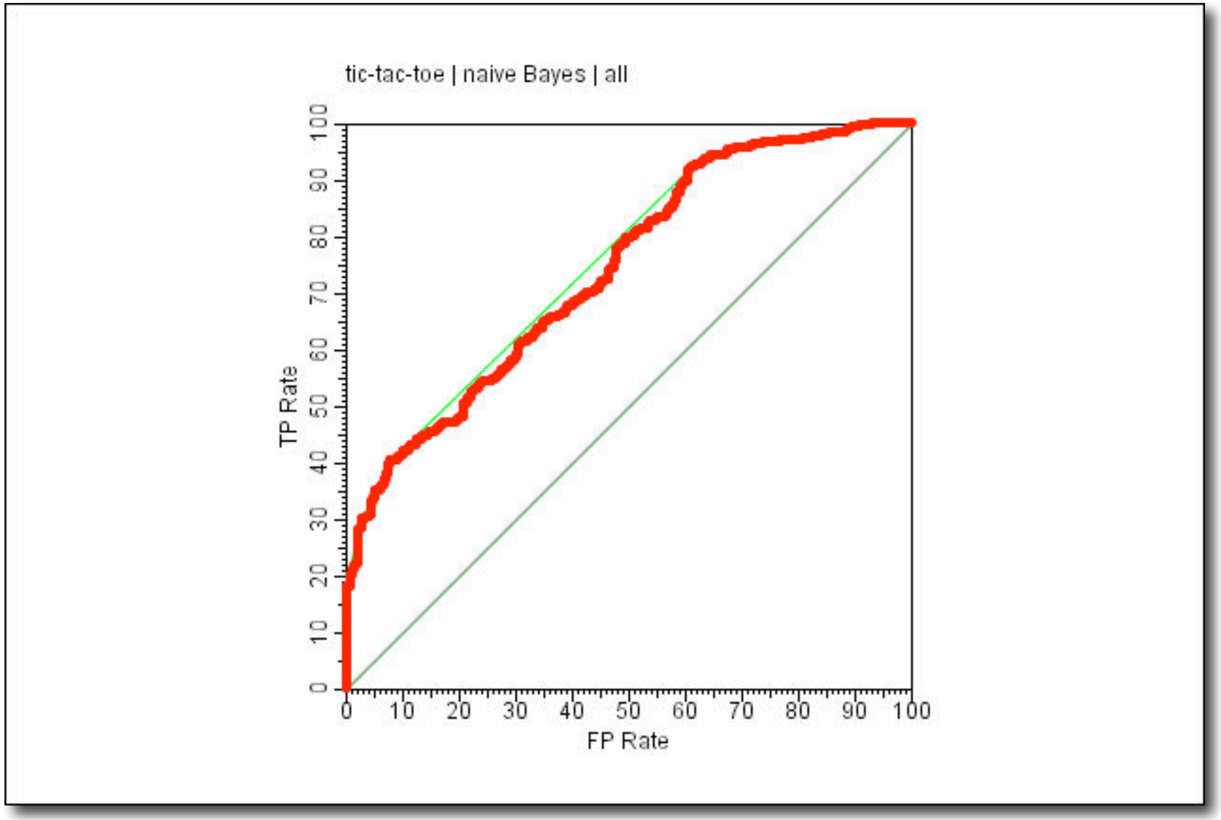
- Good separation between classes, convex curve

Some example ROC curves



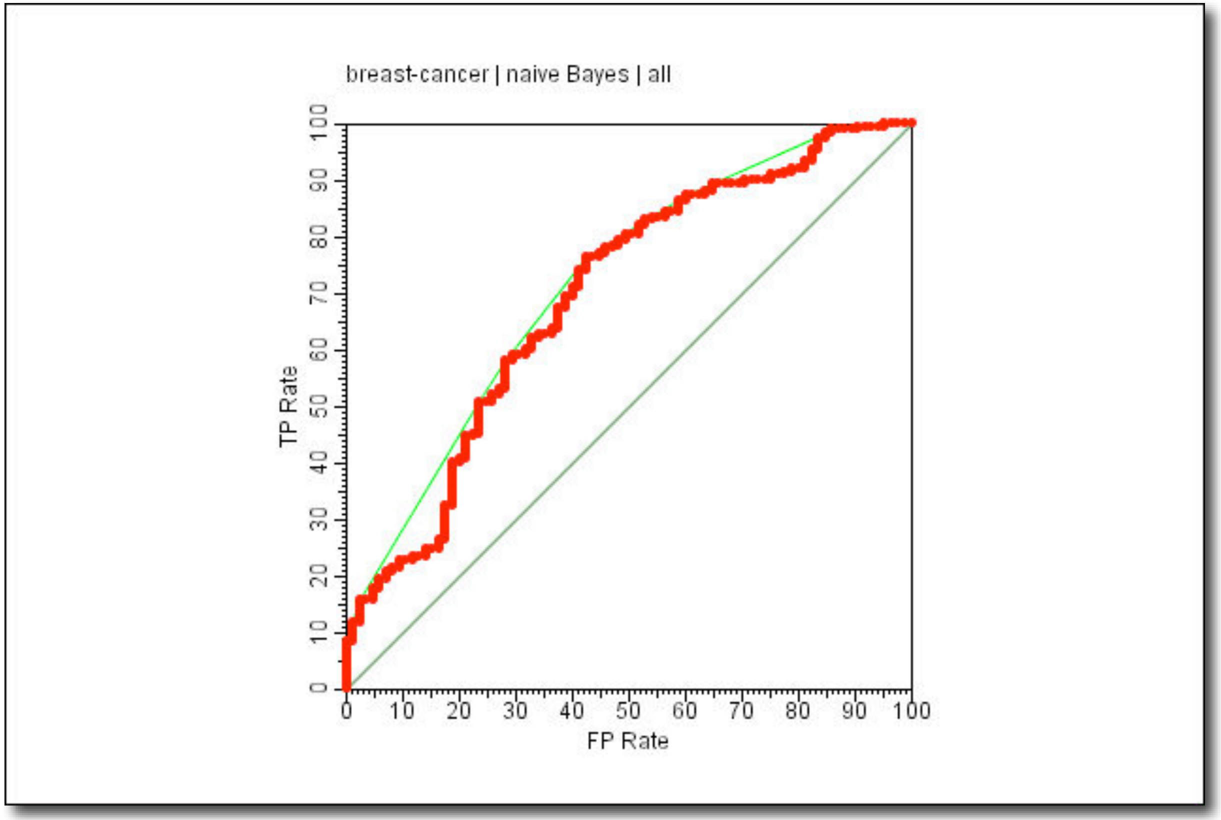
- Reasonable separation, mostly convex

Some example ROC curves



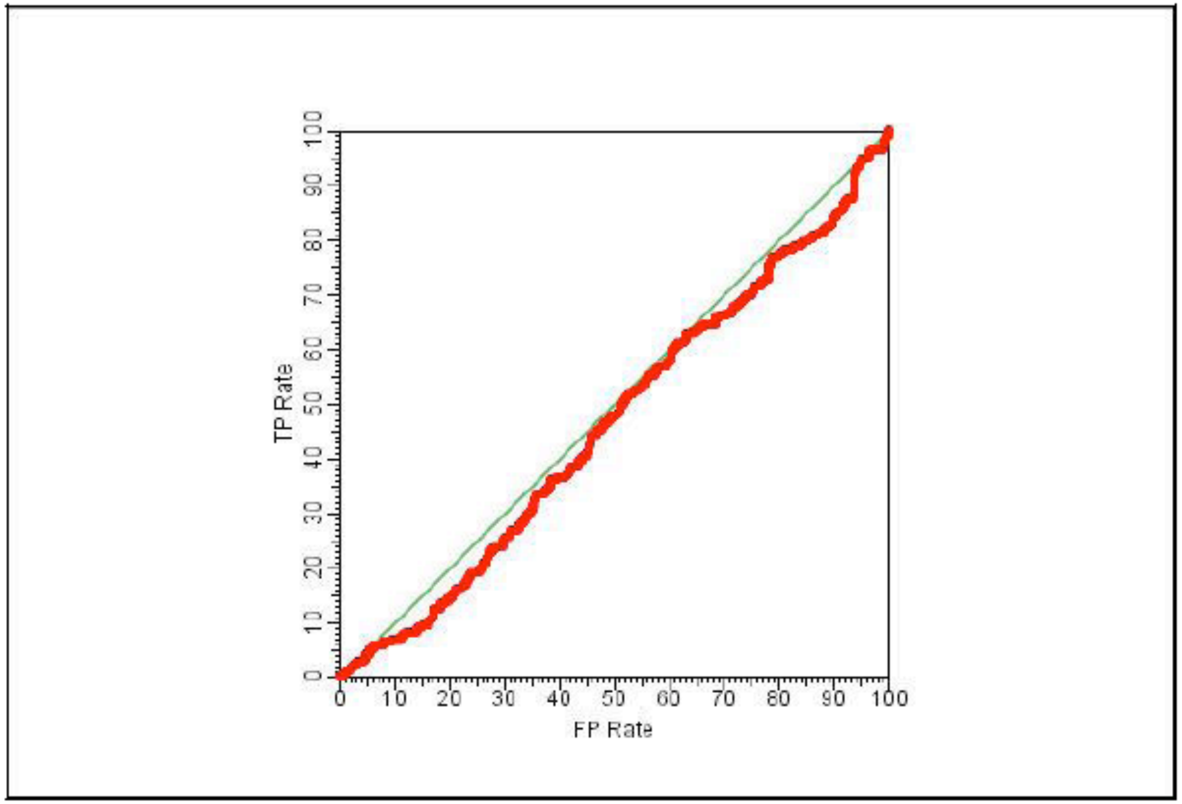
- Decent performance in first and last segments of ranking, more or less random performance in middle segment

Some example ROC curves



- Poor separation, large and small concavities indicating locally worse-than-random behaviour

Some example ROC curves



- Random performance

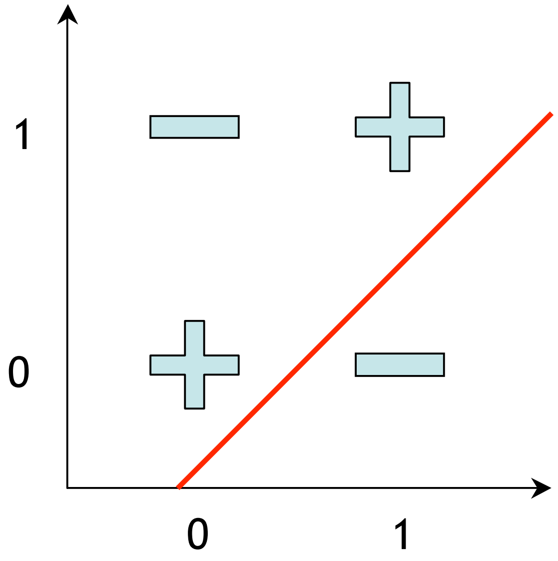
ROC curves for rankers

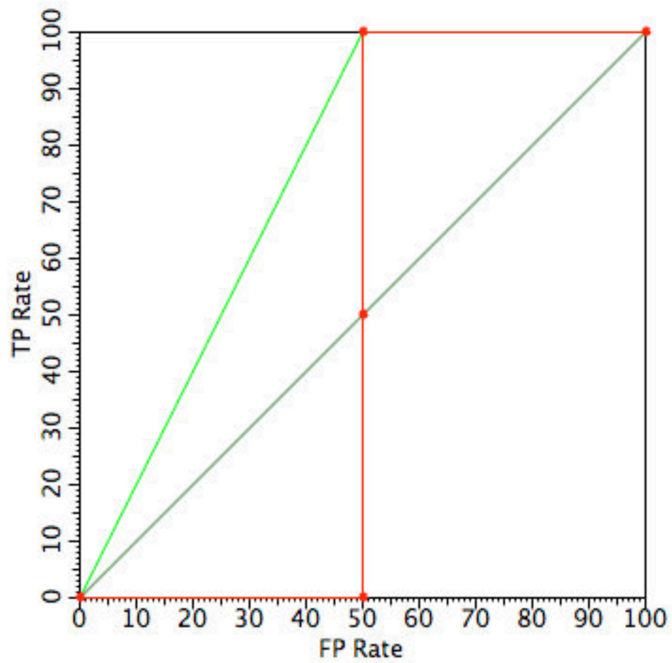
- The curve visualises the quality of the ranker or probabilistic model on a test set, without committing to a classification threshold
- The slope of the curve indicates empirical (test set) class distribution in local segment
  - straight segment -> test set indicates no need to distinguish between those examples
  - slope can be used for calibration
- Concavities indicate locally worse than random behaviour
  - distinguishing between those examples is harmful
  - convex hull gets rid of concavities by binning scores

The AUC metric

- The Area Under ROC Curve (AUC) assesses the ranking in terms of separation of the classes
  - all the +ves before the –ves: AUC=1
  - random ordering: AUC=0.5
  - all the –ves before the +ves: AUC=0
- Equivalent to the Mann-Whitney-Wilcoxon sum of ranks statistic
  - estimates probability that randomly chosen +ve is ranked before randomly chosen –ve
  - $\frac{S_{-} - Pos(Pos+1)/2}{Pos \cdot Neg}$ where S_{-} is the sum of ranks of –ves
- Gini coefficient = 2*AUC–1 (area above diag.)
  - NB. not the same as Gini index!

AUC=0.5 not always random



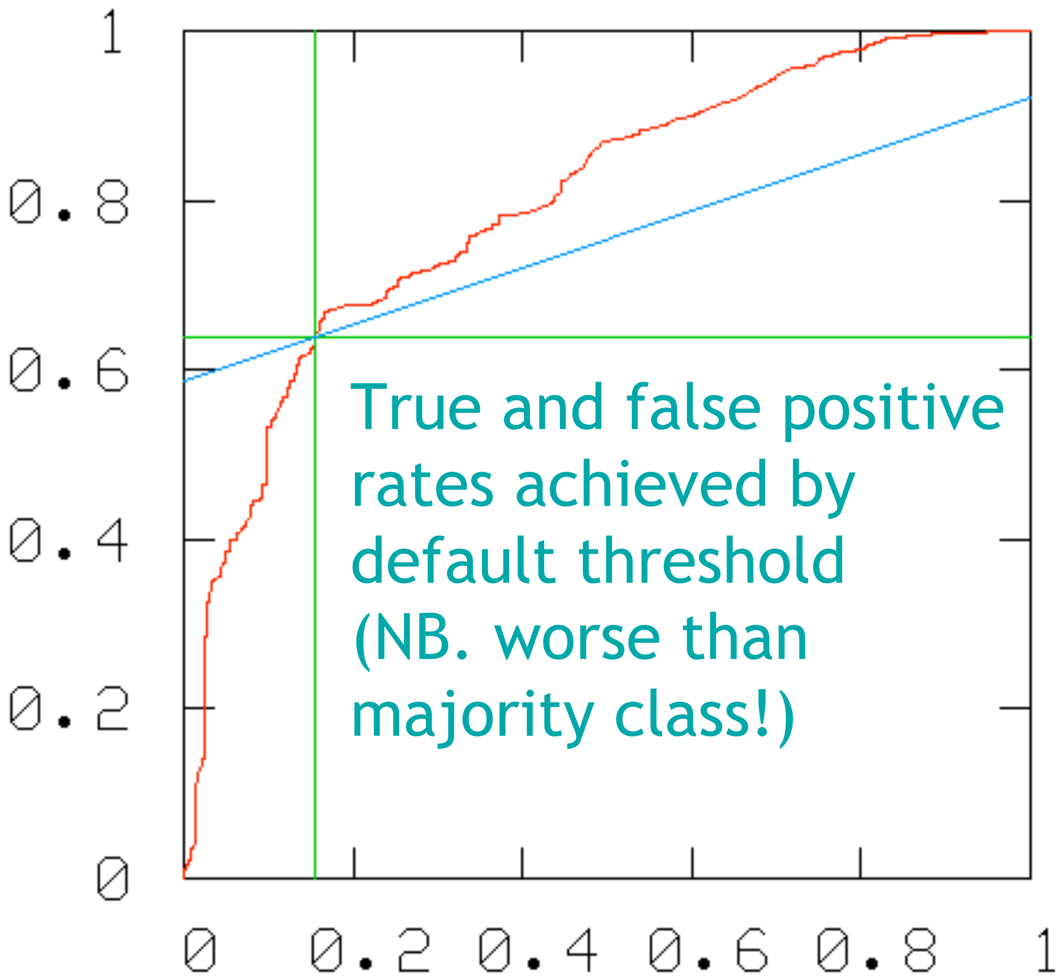


- Poor performance because data requires two classification boundaries

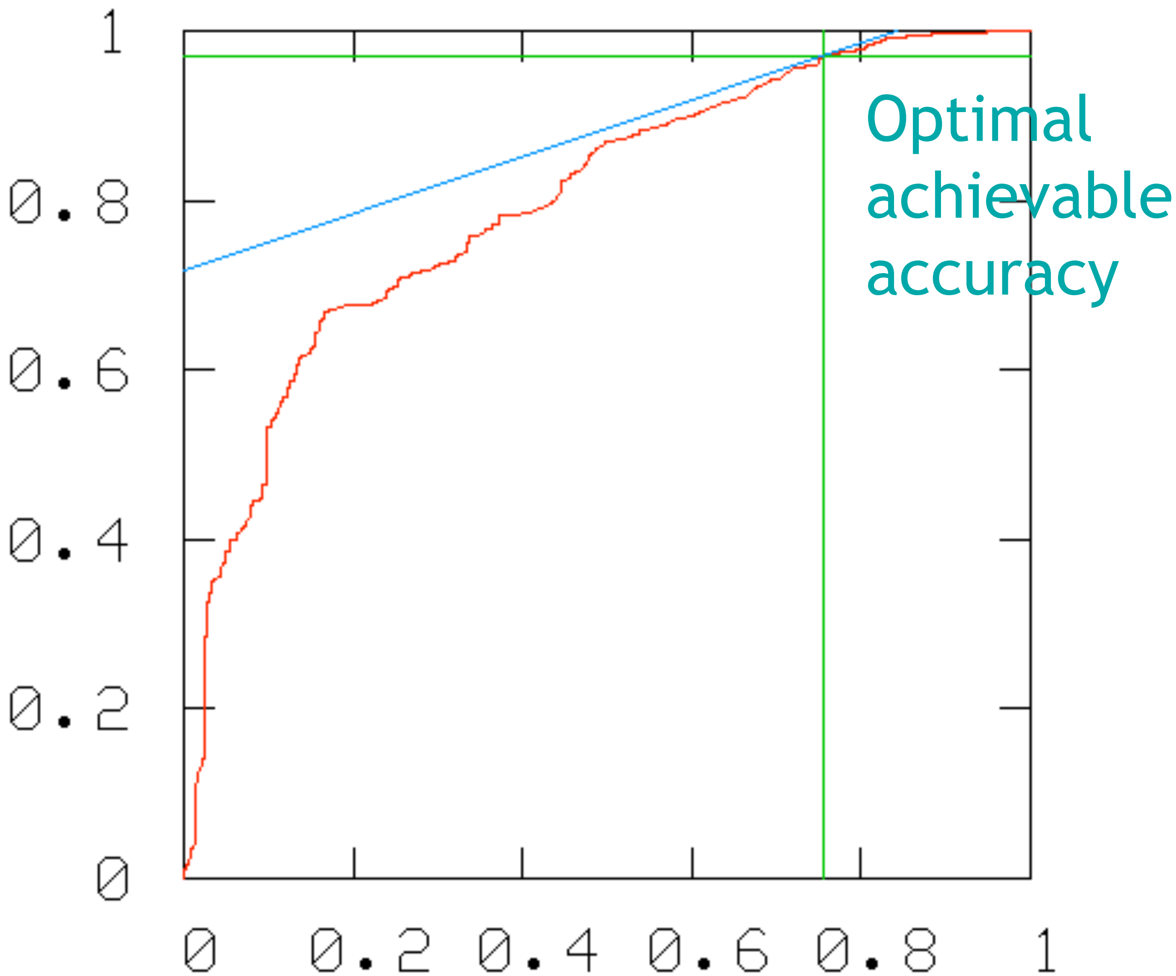
Turning rankers into classifiers

- Requires decision rule, i.e. setting a threshold on the scores f(x)
  - e.g. Bayesian: predict positive if $\frac{P(x \mid +)}{P(x \mid -)} > \frac{Neg}{Pos}$
  - equivalently: $\frac{P(+ \mid x)}{P(- \mid x)} = \frac{P(x \mid +) \cdot Pos}{P(x \mid -) \cdot Neg} > 1$
- If scores are calibrated we can use a default threshold of 1 on the posterior odds
  - with uncalibrated scores we need to learn the threshold from the data
  - NB. naïve Bayes is uncalibrated
    - i.e. don't use Pos/Neg as prior!

Uncalibrated threshold



Calibrated threshold



Classification vs. ranking

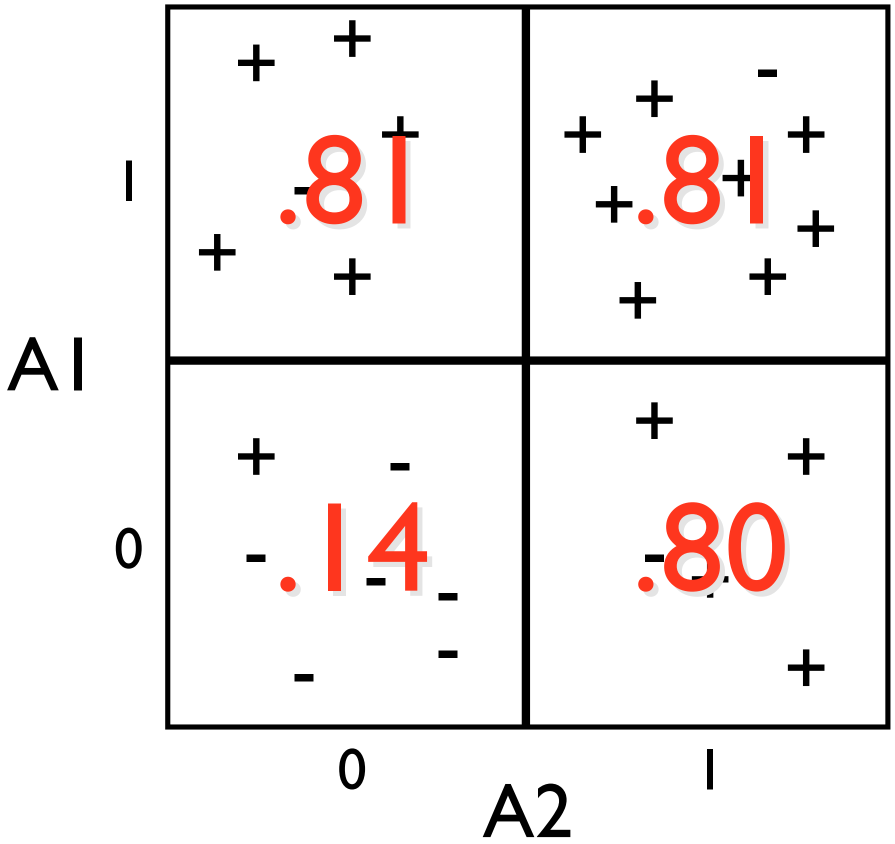
- Classifiers and rankers optimise a different loss function
  - classifier minimises classification errors (O(n))
  - ranker minimises ranking errors (O(n^2))
    - number of misclassified (+ve,–ve) pairs

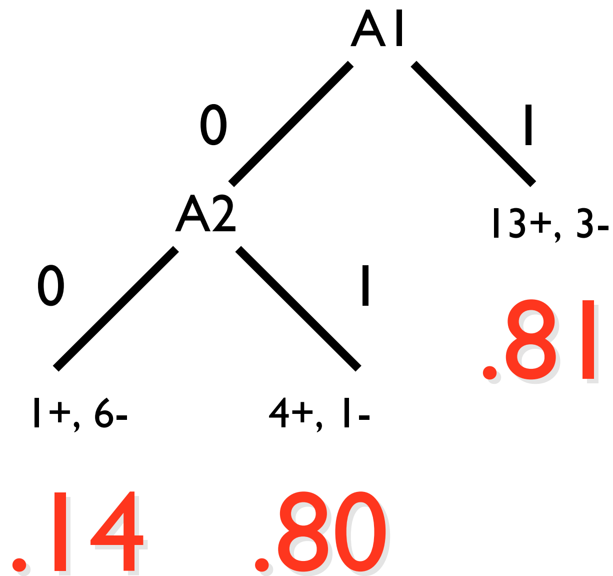
- The best achievable ROC point may not lie on the best achievable ROC curve
  - would probably learn a different weight vector for linear model

Probability estimation

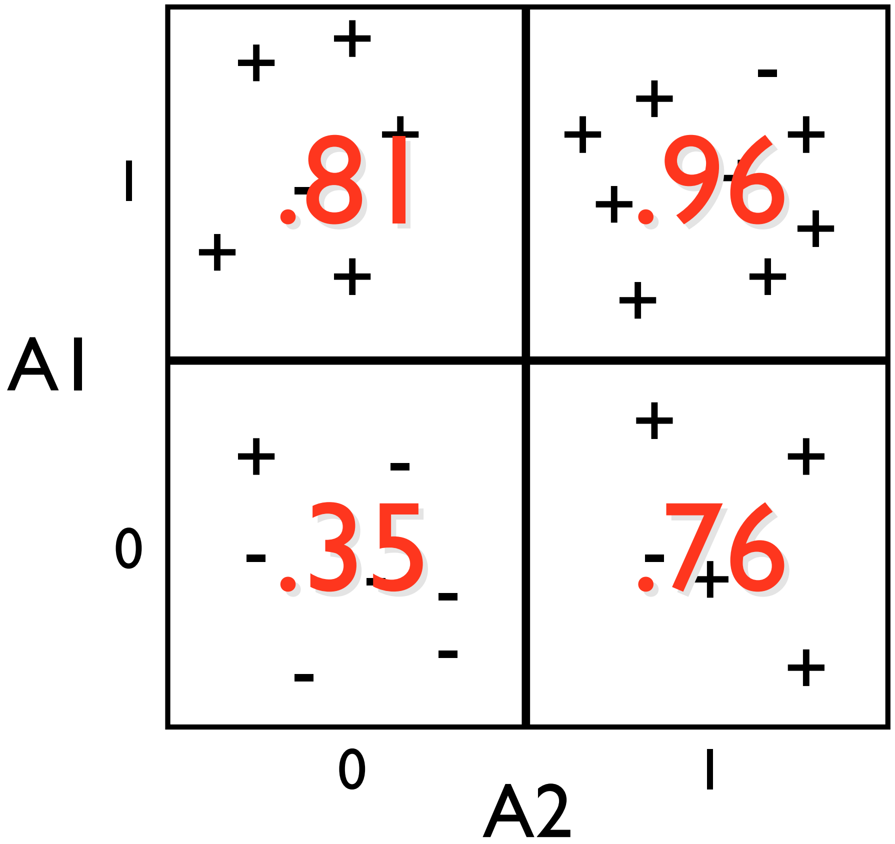
- A probability estimator assigns a probability to each point in instance space
  - more restrictive than scores, which can be shifted or scaled without affecting the ranking
- Scores are not necessarily good probability estimates, even when normalised
  - e.g., naive Bayes scores tend to be close to 0 or 1
- Turning a ranker into a probability estimator requires calibration

Probabilities from trees





Naïve Bayes probabilities



- $LR_{A1}=13/3$, $LR_{\neg A1}=5/7$, $LR_{A2}=12/2$, $LR_{\neg A2}=6/8$,
- $LR_{1}=156/6$, $LR_{2}=60/14$, $LR_{3}=78/24$, $LR_{4}=30/56$

Good probabilities ≠ good ranking

- .8+.7+ .6+ .4- .3- .2-
  - AUC = 1
  - MSE (aka Brier score) = .097
- 1+.9+ .51- .49+ .1- 0-
  - AUC = 8/9 (worse)
  - MSE = .090 (better)

Calibration

- Well-calibrated probabilities have the following property:
  - in a sample with predicted probability *p*, the expected proportion of positives is close to *p*
- This means that the predicted likelihood ratio approximates the slope of the ROC curve
  - perfect calibration implies convex ROC curve
- This suggests a simple calibration procedure:
  - discretise scores using convex hull and derive probability in each bin from ROC slope
    - = isotonic regression (Zadrozny & ELkan, 2001; Fawcett & Niculescu-Mizil, 2007; Flach & Matsubara, 2007)

Decomposing the Brier score

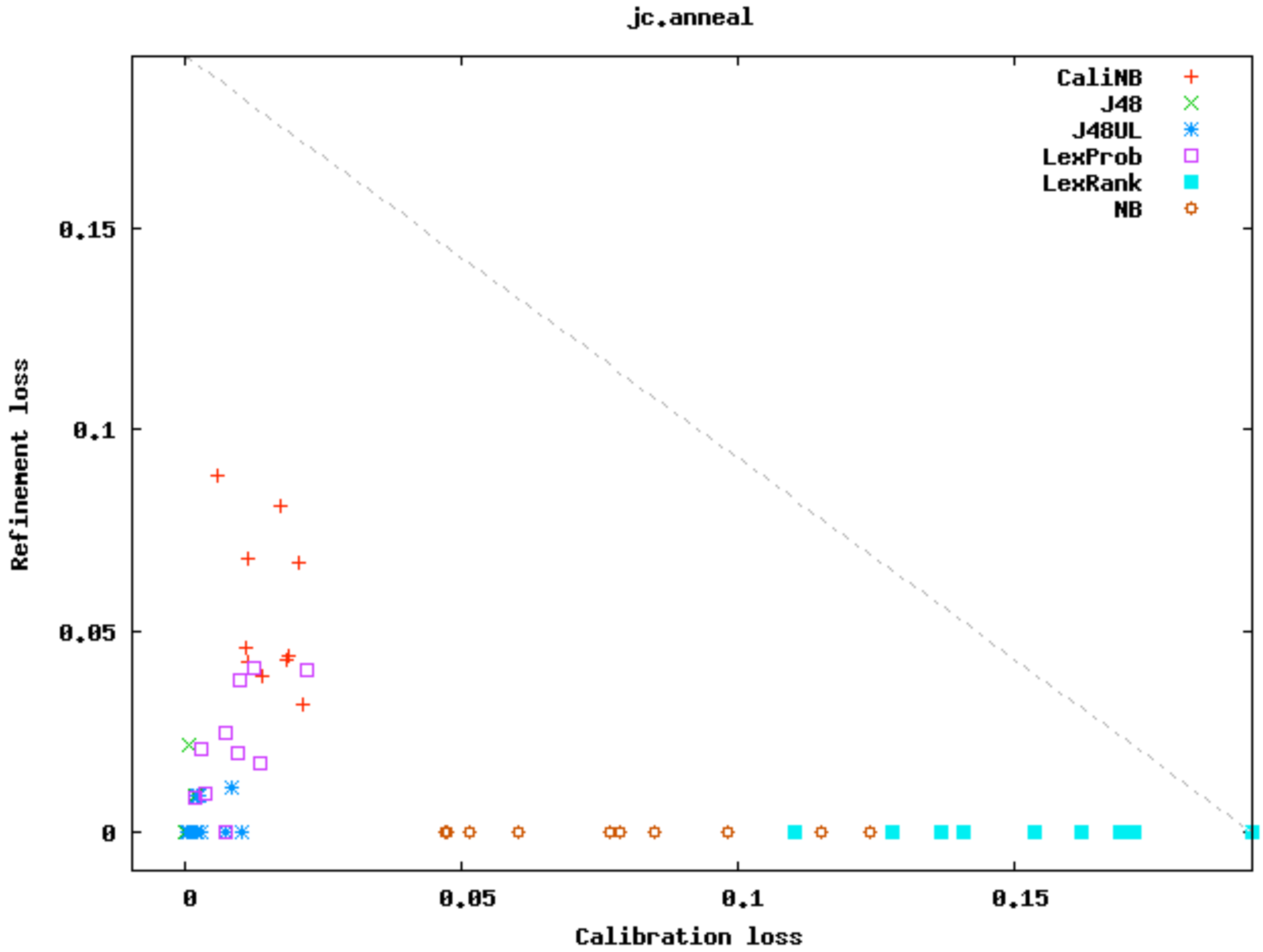
Theorem 1 *Given an ROC curve produced by a ranker on a test set, let n_i^+ and n_i^- be the number of positives and negatives in the i-th segment of the ROC curve, $n_i = n_i^+ + n_i^-$, $p_i = \frac{n_i^+}{n_i}$, and $\hat{p}_i$ be the predicted probability in that segment. The Brier score is equal to*

$$BS = \frac{1}{|X|}\sum_i n_i(\hat{p}_i - p_i)^2 + \frac{1}{|X|}\sum_i n_i p_i(1 - p_i)$$

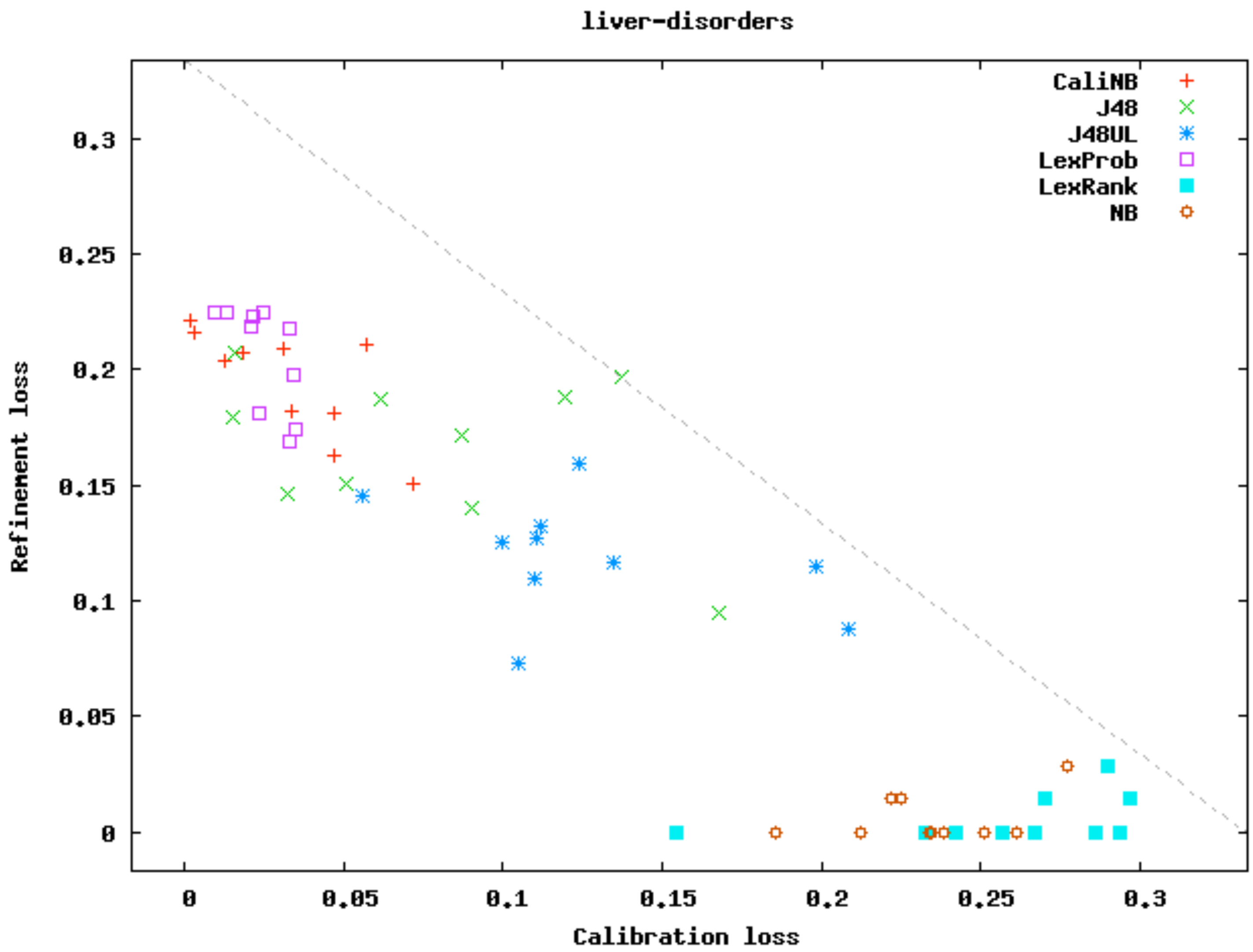
calibration loss:
mean squared deviation from empirical probabilities derived from slope of ROC segments

refinement loss:
defined purely in terms of empirical probabilities

Calibration and refinement



Calibration and refinement



From ranks to probabilities

- One way to obtain a well-calibrated probability estimator:
  - train a ranker from labelled training data
  - draw ROC curve on test set
  - obtain a calibration map from convex hull
- NB. This is exactly what decision trees do, taking into account that:
  - test set could be training set (risk of overfitting)
  - decision tree training set ROC curves are provably convex, so no need for convex hull

ROC-based model manipulation

- ROC analysis allows creation of model variants without re-training
  - e.g., manipulating ranker thresholds or scores
- Example: re-labelling decision trees
  - (Ferri et al., 2002)
- Example: locally adjusting rankings
  - (Flach & Wu, 2003)

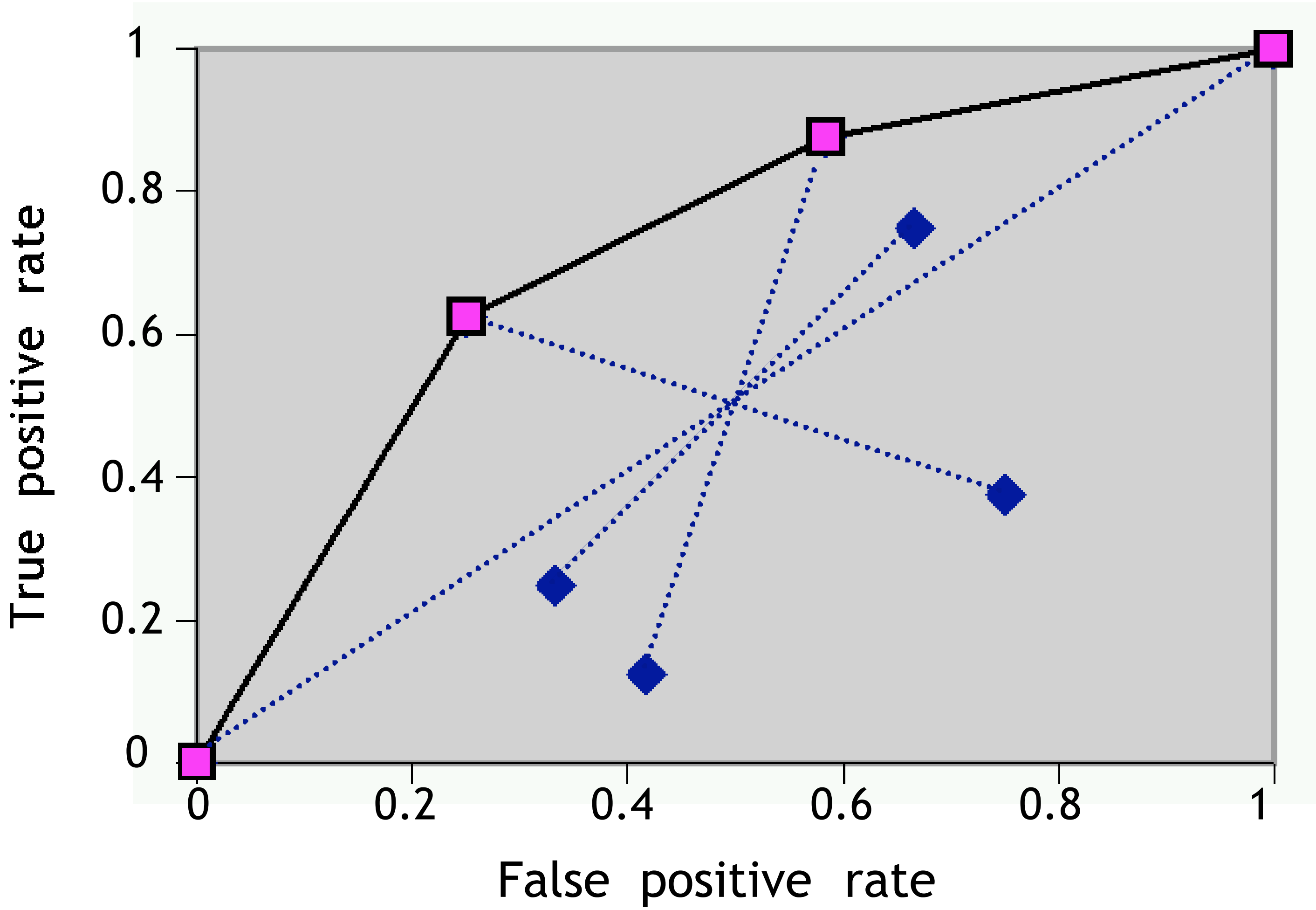
Re-labelling decision trees

- A decision tree can be seen as an unlabelled tree (a clustering tree):
  - Given n leaves and 2 classes, there are 2^n possible labellings, each representing a classifier

- Use ROC analysis to select the best labellings

	Training Distribution		Labellings							
	+	-								
Leaf 1	40	20	-	-	-	-	+	+	+	+
Leaf 2	50	10	-	-	+	+	-	-	+	+
Leaf 3	30	50	-	+	-	+	-	+	-	+

DT labellings in ROC space

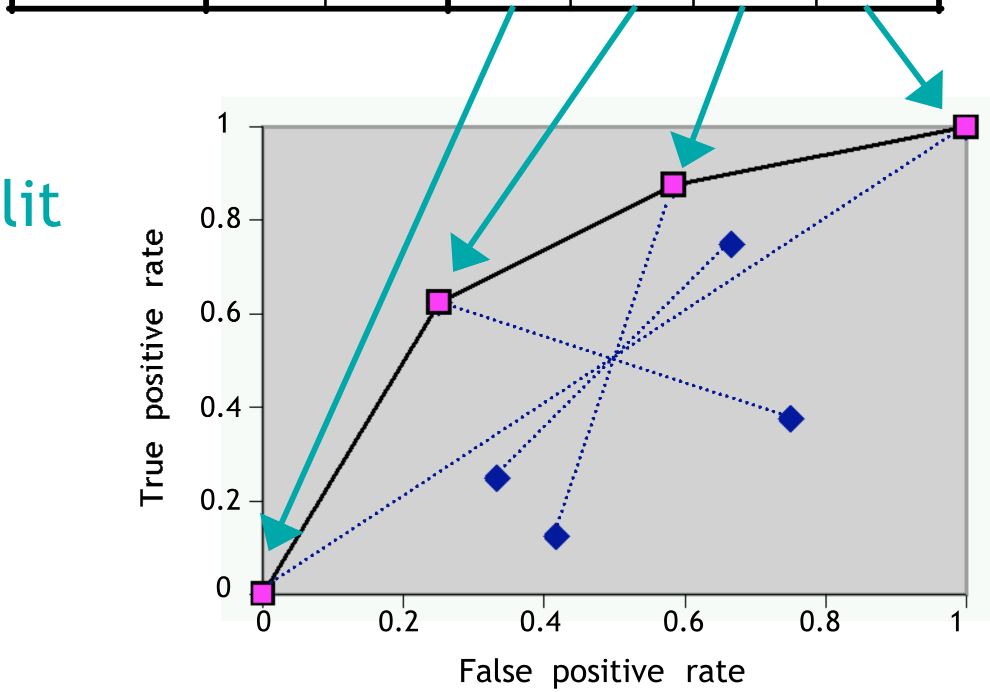


Selecting optimal labellings

1. Rank leaves by likelihood ratio P(l|+)/P(l|–)

2. For each possible split point, label leaves before split + and after split –

	+	-				
Leaf 2	50	10	-	+	+	+
Leaf 1	40	20	-	-	+	+
Leaf 3	30	50	-	-	-	+



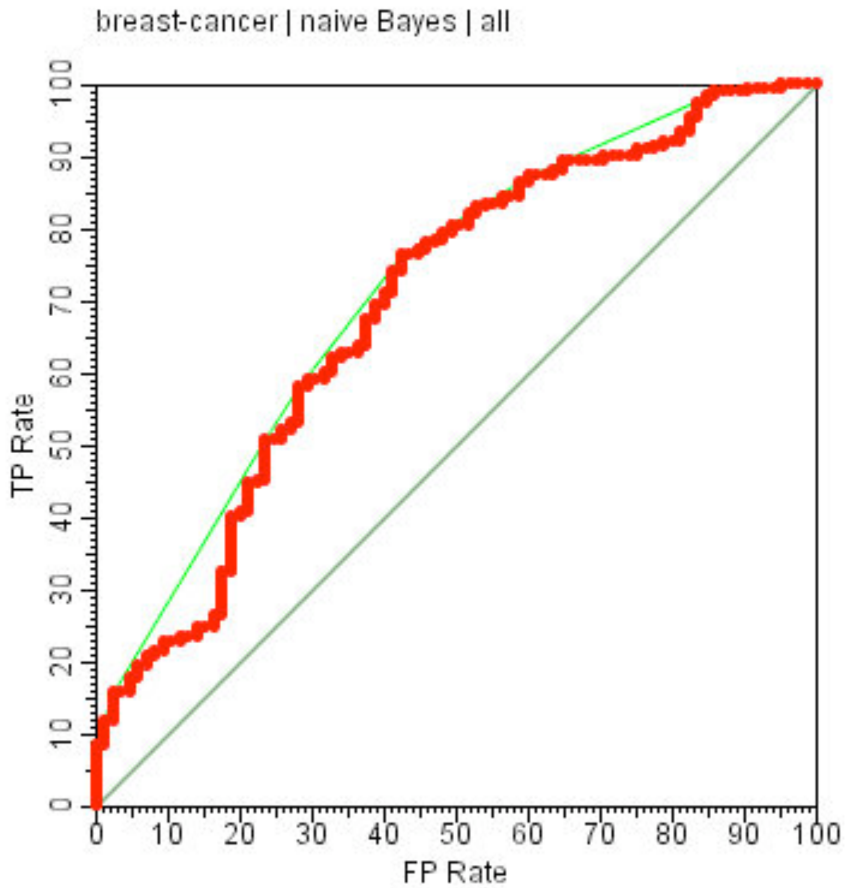
Why does it work?

- Decision trees are rankers if we use class distributions in the leaves
  - Probability Estimation Trees (Provost & Domingos, 2003)
- ROC curve can be constructed by sliding threshold
  - just as with naïve Bayes
- Equivalently, we can order instances, which boils down to ordering leaves
  - because all instances in a leaf are ranked together
- NB. Curve may not be convex on test set

Repairing concavities

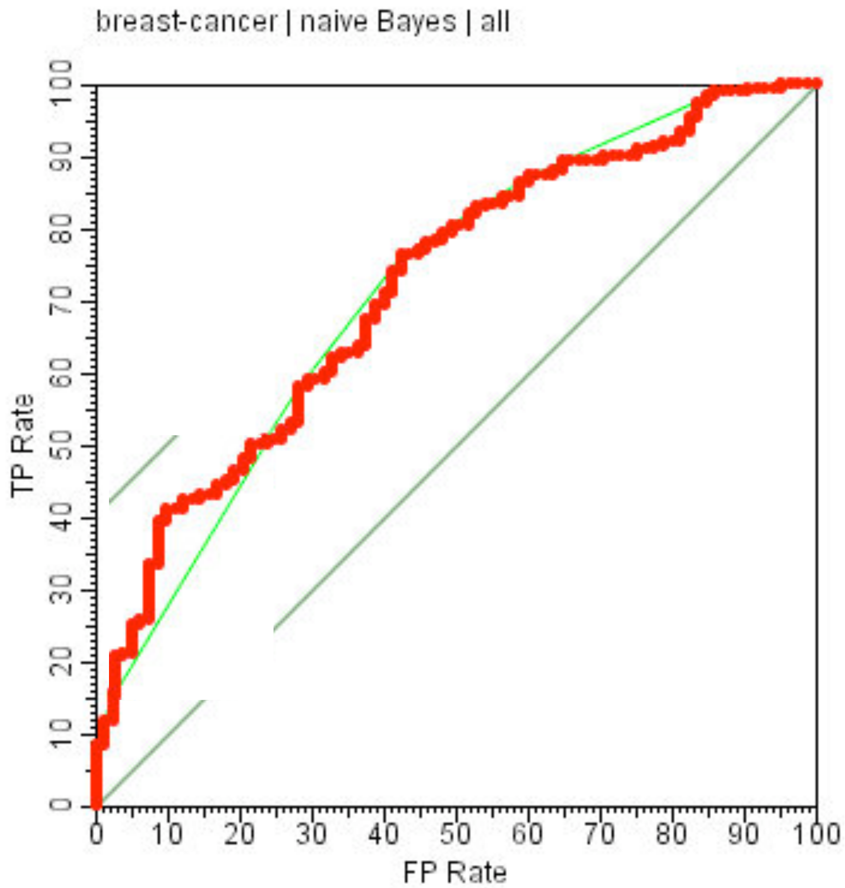
- Concavities in ROC curves from rankers indicate worse-than-random segments in the ranking
- Idea 1: use binned ranking (aka discretised scores) —> convex hull
- Idea 2: invert ranking in segment
- Need to avoid overfitting

Repairing concavities



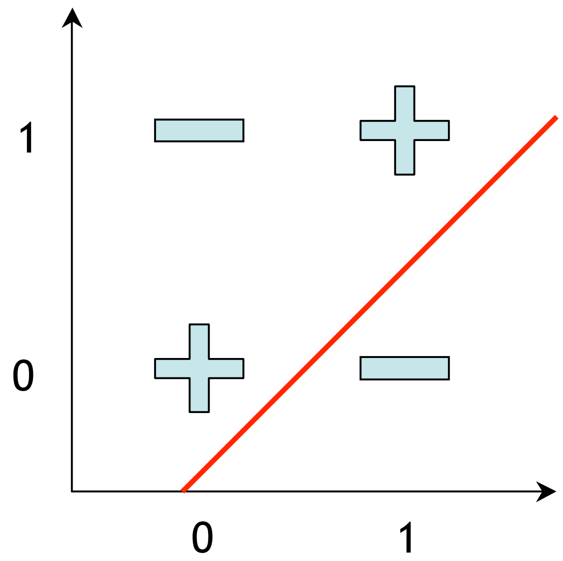
- Convex hull corresponds to binning the scores into variable-sized bins in order to eliminate locally worse-than-random ranking (concavity)

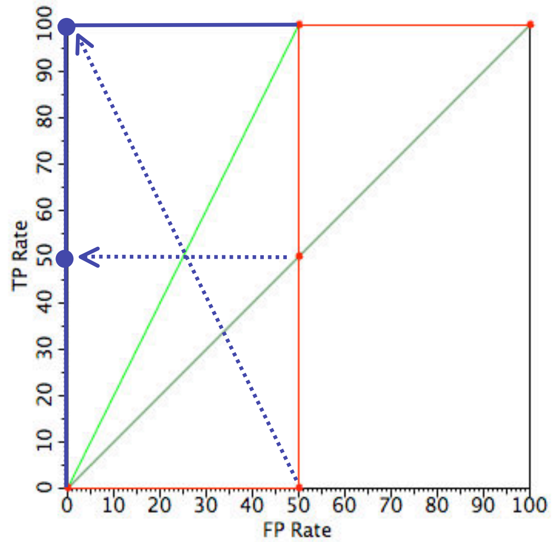
Repairing concavities



- Convex hull corresponds to binning the scores into variable-sized bins in order to eliminate locally worse-than-random ranking (concavity)
- Can do better than this: invert ranking in each concavity

Example: XOR





above line?

yes — no

invert XXXX ted XXX use ranking in 1st segment

use ranking in 2nd segment

More than two classes

- Two-class ROC analysis is a special case of multi-objective optimisation
  - don’t commit to trade-off between objectives
- Pareto front is the set of points for which no other point improves all objectives
  - points not on the Pareto front are dominated
  - assumes monotonic trade-off between objectives
- Convex hull is subset of Pareto front
  - assumes linear trade-off between objectives
    - e.g. accuracy, but not precision

How many dimensions?

- Depends on the cost model
  - 1-vs-rest: fixed misclassification cost $C(\neg c|c)$ for each class $c \in C$ —> $|C|$ dimensions
    - ROC space spanned by either tpr for each class or fpr for each class
  - 1-vs-1: different misclassification costs $C(c_i|c_j)$ for each pair of classes $c_i \neq c_j$ —> $|C|(|C|-1)$ dimensions
    - ROC space spanned by fpr for each (ordered) pair of classes
- Results about convex hull, optimal point given linear cost function etc. generalise
  - (Srinivasan, 1999)

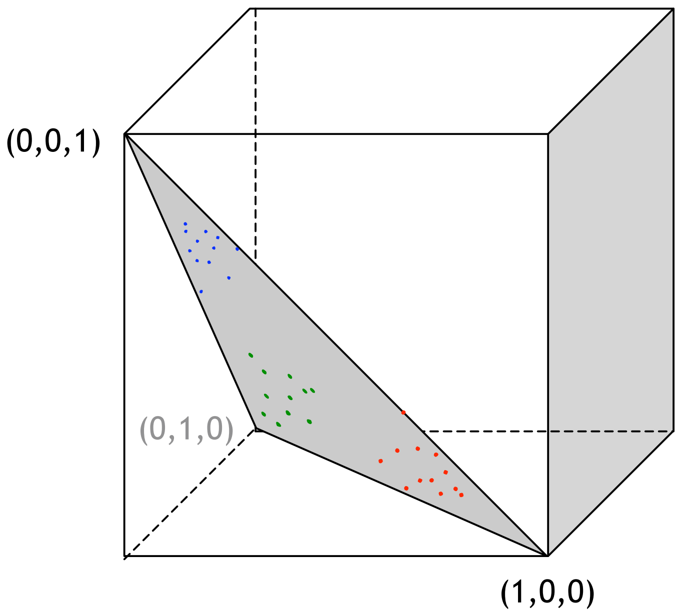
Multi-class AUC

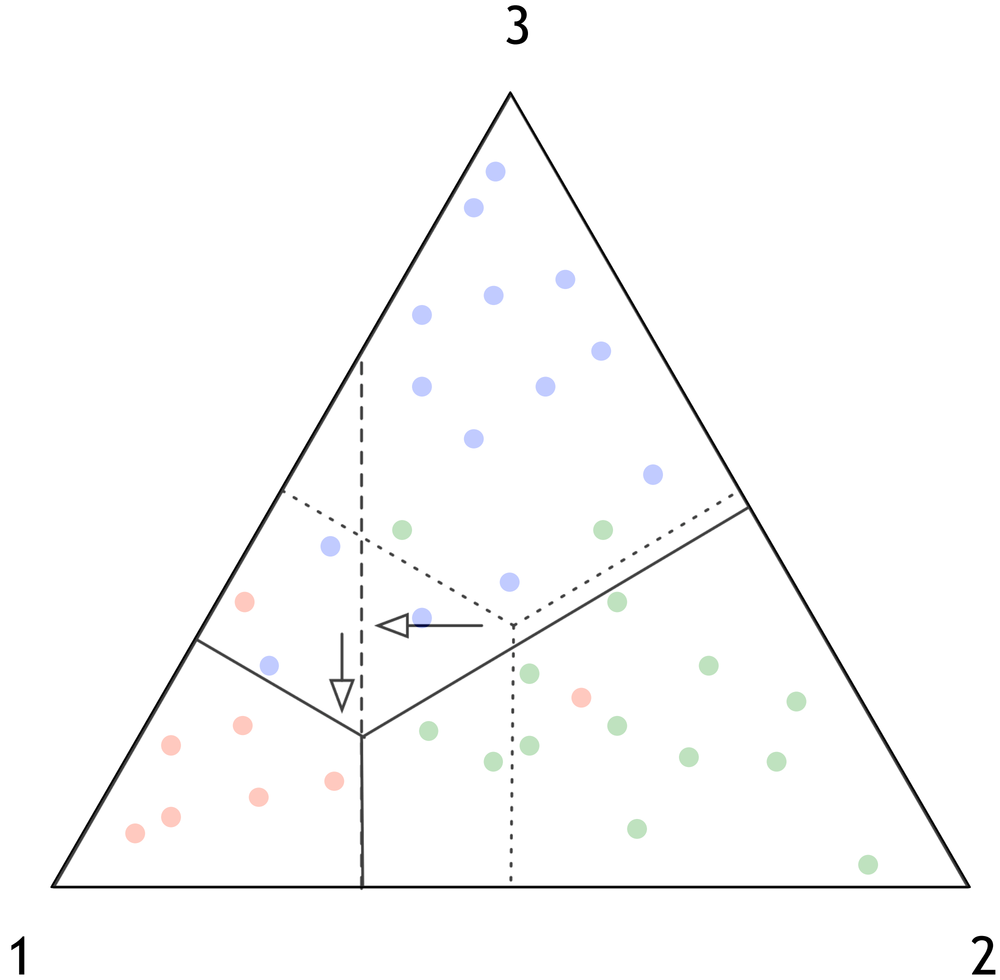
- In the most general case, we want to calculate Volume Under ROC Surface (VUS)
  - See (Mossman, 1999) for VUS in the 1-vs-rest three-class case
- Can be approximated by projecting down to set of two-dimensional curves and averaging
  - MAUC (Hand & Till, 2001): 1-vs-1, unweighted average
  - (Provost & Domingos, 2001): 1-vs-rest, AUC for class c weighted by P(c)

Multi-class calibration

1. From thresholds to weights:
   - predict $\text{argmax}_c\ w_c\ f(x,c)$
   - NB. two-class thresholds are a special case:
     - $w_+\ f(x,+) > w_-\ f(x,-) \Leftrightarrow f(x,+)/f(x,-) > w_-/w_+$
2. Setting the weights (Lachiche & Flach, 2003)
   - Assume an ordering on classes and set the weights in a greedy fashion
     - Set $w_1 = 1$
     - For classes c=2 to n
       - look for the best weight w_c according to the weights fixed so far for classes c'<c, using the two-class algorithm

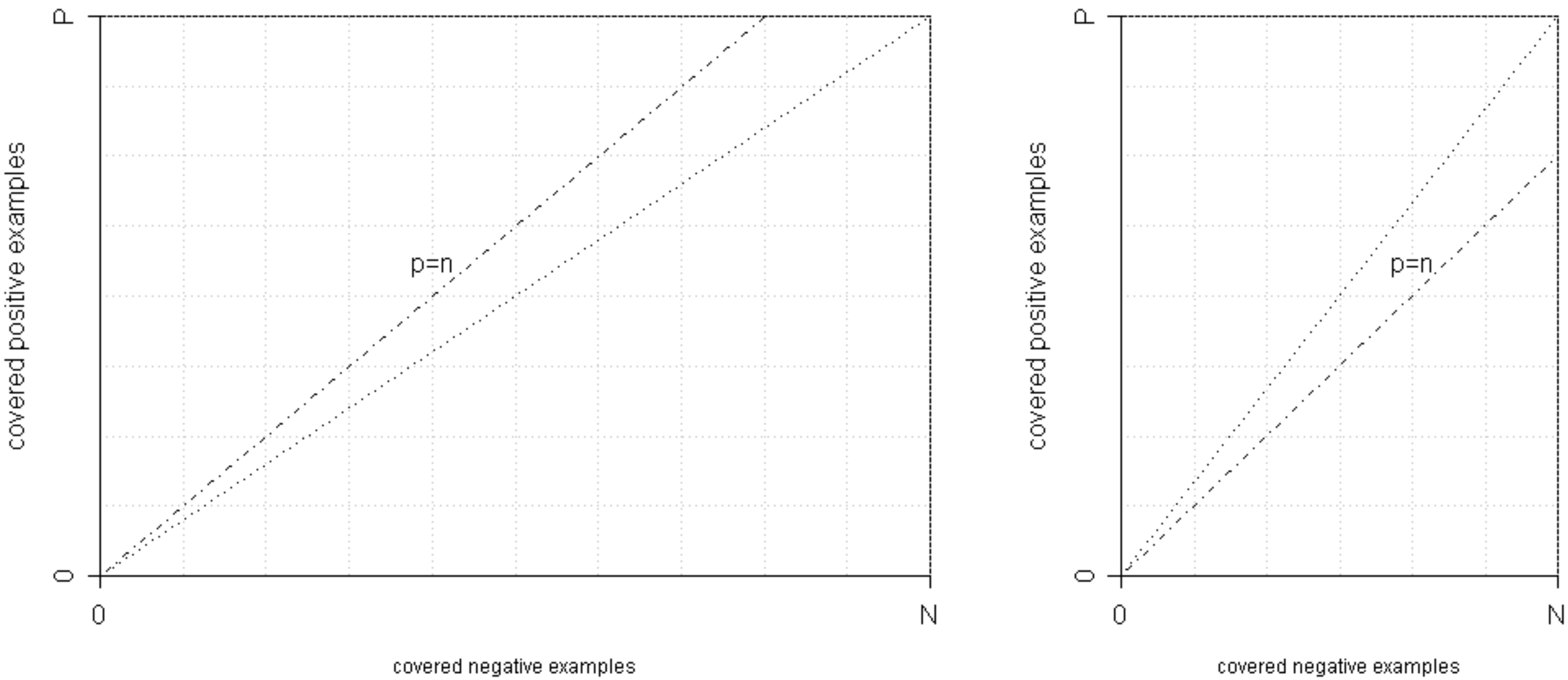
Example: 3 classes





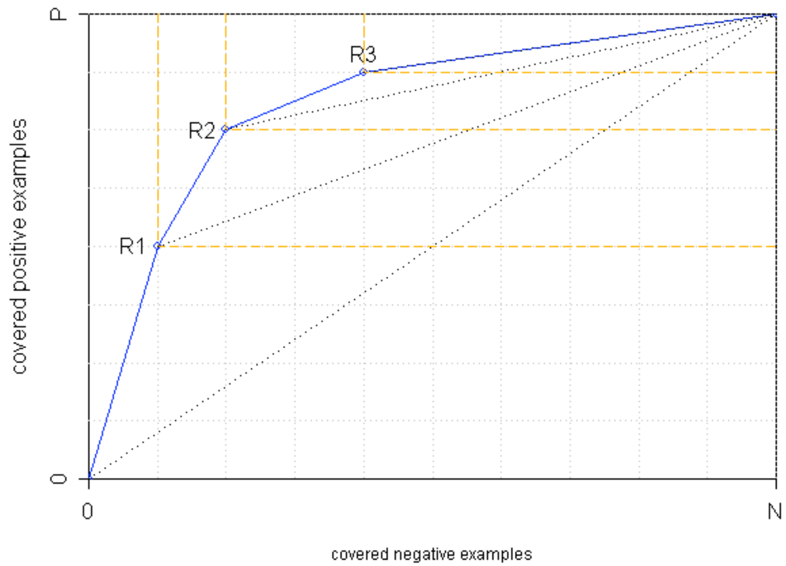
Coverage space (Fürnkranz & Flach, 2005)

- Coverage space is ROC space with absolute rather than relative frequencies
  - x-axis: covered –ves n (instead of fpr = n/Neg)
  - y-axis: covered +ves p (instead of tpr = p/Pos)



Coverage space vs. ROC space

- Coverage space can be used if class distribution (reflected by shape) is fixed
  - slope now corresponds to posterior odds rather than likelihood ratio
  - iso-accuracy lines always have slope 1
  - very useful to analyse behaviour of particular learning algorithm



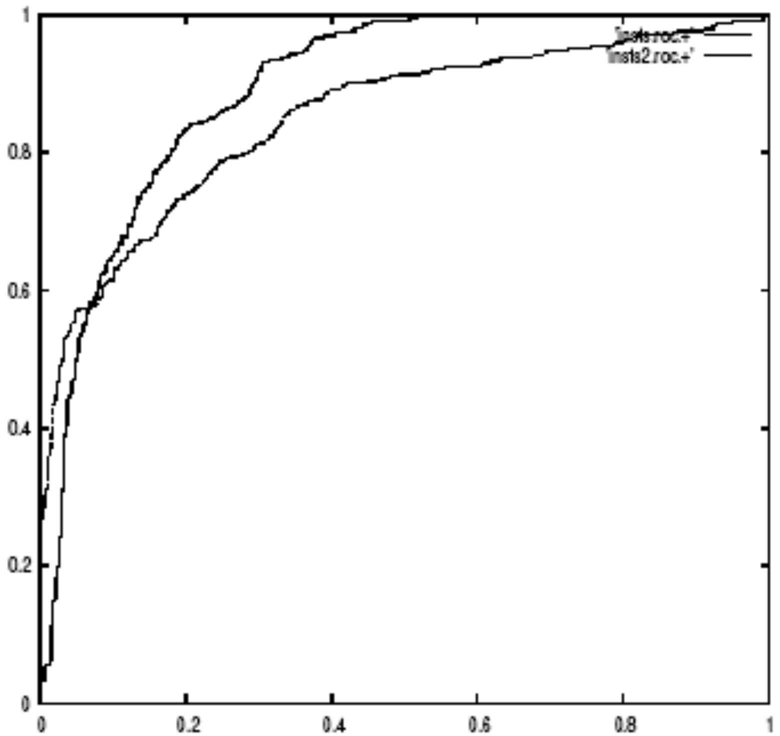
Precision-recall curves

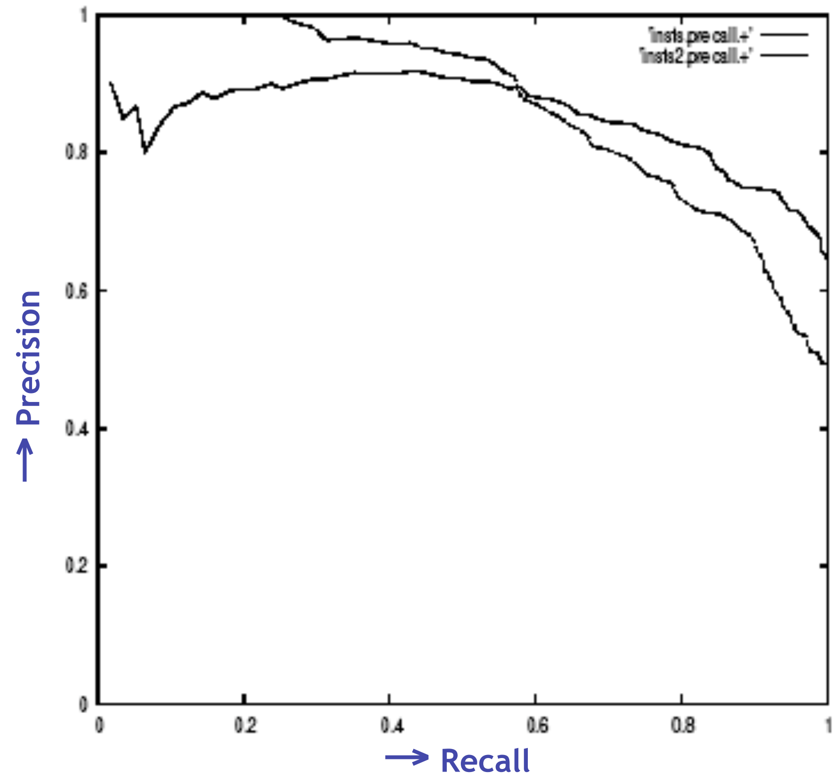
	Predicted positive	Predicted negative	
Positive examples	**TP**	**FN**	**Pos**
Negative examples	**FP**	**TN**	**Neg**
	PPos	**PNeg**	**N**

- Precision prec = TP/PPos = TP/TP+FP
  - fraction of positive predictions correct
- Recall rec = tpr = TP/Pos = TP/TP+FN
  - fraction of positives correctly predicted
- Note: neither depends on true negatives
  - makes sense in information retrieval, where true negatives tend to dominate —> low fpr easy

PR curves vs. ROC curves

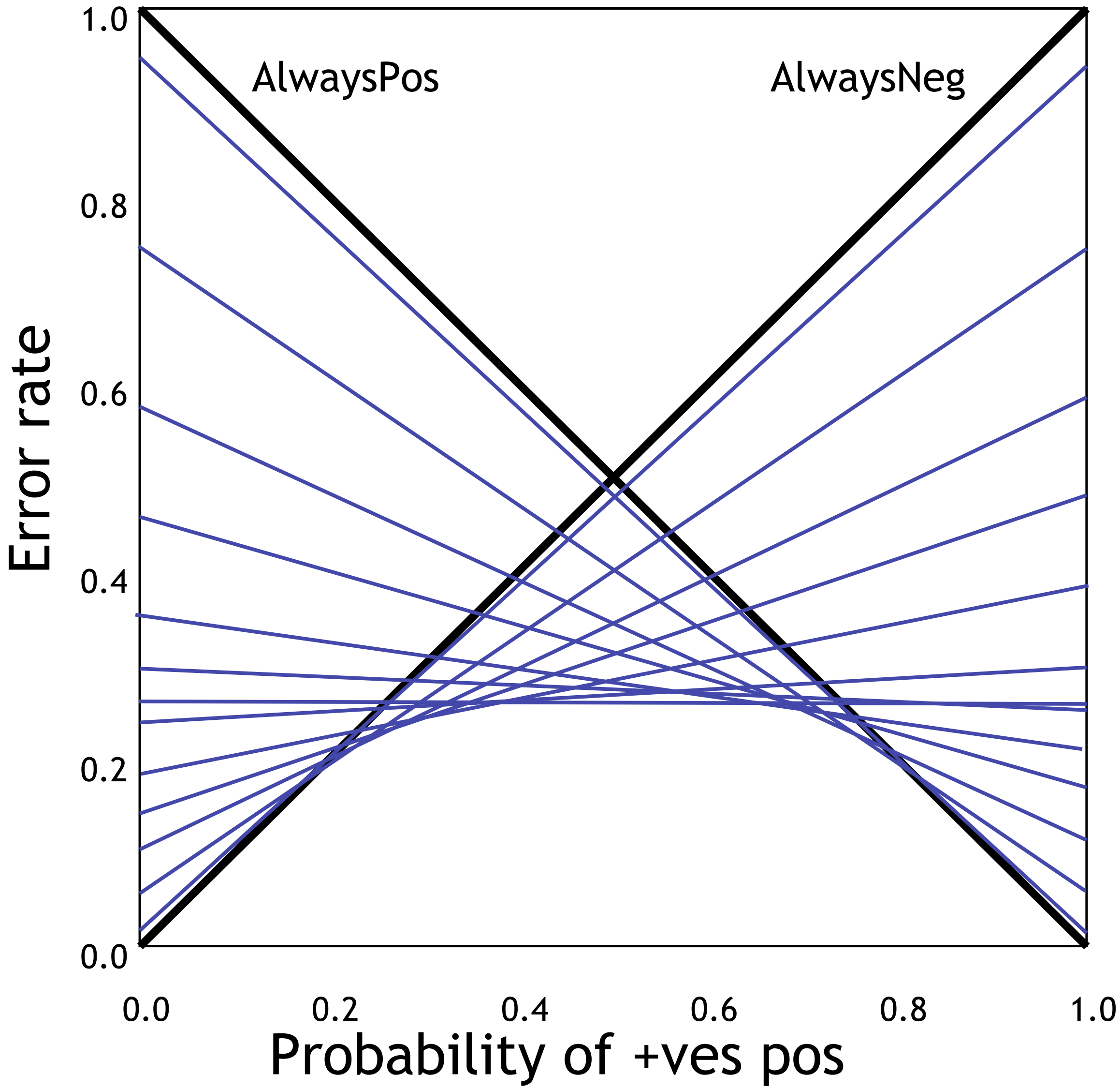
- Two ROC curves
- Corresponding PR curves





From (Fawcett, 2004)

Cost curves (Drummond & Holte, 2006)



Taking costs into account

- Error rate is err = (1–tpr)*pos + fpr*(1–pos)

- Define probability cost function as

$$pcf = \frac{pos \cdot C(-|+)}{pos \cdot C(-|+) + neg \cdot C(+|-)}$$

- Normalised expected cost is

  nec = (1–tpr)*pcf + fpr*(1–pcf)

Concluding remarks

- ROC analysis for model evaluation and selection
  - key idea: separate performance on classes
  - think rankers, not classifiers!
  - information in ROC curves not easily captured by statistics
- ROC analysis for use within ML algorithms
  - one classifier can be many classifiers!
  - separate skew-insensitive parts of learning…
    - probabilistic model, unlabelled tree
  - …from skew-sensitive parts
    - selecting thresholds or class weights, labelling and pruning

Outlook

- Several issues not covered in this tutorial
  - optimising AUC rather than accuracy when training
    - e.g. RankBoost optimises AUC (Cortes & Mohri, 2003)
- Many open problems remain
  - ROC analysis in rule learning
    - overlapping rules
  - relation between training skew and testing skew
  - multi-class ROC analysis

References

C. Cortes and M. Mohri (2003). AUC optimization vs. error rate minimization. In Advances in Neural Information Processing Systems (NIPS'03). MIT Press.

C. Drummond and R.C. Holte (2006). Cost curves: an improved method for visualizing classifier performance. Machine Learning, 65(1): 95-130.

T. Fawcett (2004). ROC graphs: Notes and practical considerations for data mining researchers. Technical report HPL-2003-4, HP Laboratories, Palo Alto, CA, USA. Revised March 16, 2004. Available at http://www.purl.org/NET/tfawcett/papers/ROC101.pdf.

T. Fawcett and A. Niculescu-Mizil (2007). PAV and the ROC convex hull. Machine Learning, 68(1): 97-106

C. Ferri, P.A. Flach, and J. Hernández-Orallo (2002). Learning Decision Trees Using the Area Under the ROC Curve. In C. Sammut and A. Hoffmann, editors, Proc. 19th International Conference on Machine Learning (ICML'02), pp. 139–146. Morgan Kaufmann.

P.A. Flach (2003). The geometry of ROC space: understanding machine learning metrics through ROC isometrics. In T. Fawcett and N. Mishra, editors, Proc. 20th International Conference on Machine Learning (ICML'03), pp. 194–201. AAAI Press.

P.A. Flach and S. Wu (2003). Reparing concavities in ROC curves. In Proc. 19th International Joint Conference on Artificial Intelligence (IJCAI'05), pp. 702-707.

P.A. Flach and E.T. Matsubara (2007). A simple lexicographic ranker and probability estimator. In Proc. 18th European Conference on Machine Learning (ECML'07). Springer.

J. Fürnkranz and P.A. Flach (2003). An analysis of rule evaluation metrics. In T. Fawcett and N. Mishra, editors, Proc. 20th International Conference on Machine Learning (ICML'03), pp. 202–209. AAAI Press.

J. Fürnkranz and P.A. Flach (2005). ROC 'n' rule learning — towards a better understanding of covering algorithms. Machine Learning, 58(1): 39-77.

D.J. Hand and R.J. Till (2001). A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems, Machine Learning, 45: 171-186.

N. Lachiche and P.A. Flach (2003). Improving accuracy and cost of two-class and multi-class probabilistic classifiers using ROC curves. In T. Fawcett and N. Mishra, editors, Proc. 20th International Conference on Machine Learning (ICML'03), pp. 416–423. AAAI Press.

D. Mossman (1999). Three-way ROCs. Medical Decision Making 1999(19): 78-89.

F. Provost and T. Fawcett (2001). Robust classification for imprecise environments. Machine Learning, 42: 203–231.

F. Provost and P. Domingos (2003). Tree induction for probability-based rankings. Machine Learning 52(3).

A. Srinivasan (1999). Note on the location of optimal classifiers in n-dimensional ROC space. Technical Report PRG-TR-2-99, Oxford University Computing Laboratory.

B. Zadrozny and C. Elkan (2001). Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In C. Brodley and A.P. Danyluk, editors, Proc. 18th International Conference on Machine Learning (ICML'01), pp. 609-616. Morgan Kaufmann.

See also http://splweb.bwh.harvard.edu:8000/pages/ppl/zou/roc.html for pointers to ROC literature.

Acknowledgements

- Many thanks to:
  - Johannes Fürnkranz, Cèsar Ferri, José Hernández-Orallo, Nicolas Lachiche, Edson Matsubara & Shaomin Wu for joint work on ROC analysis
  - Jim Farrand, Ronaldo Prati, Tarek Abudawood & Edson Matsubara for ROC visualisation software
  - Chris Drummond, Rob Holte, Tom Fawcett & Rich Roberts for additional material