ROC analysis and evaluation metrics for machine learning

Peter.Flach@bristol.ac.uk

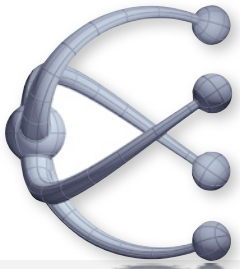
Peter A. Flach

www.cs.bris.ac.uk/~flach/

Department of Computer Science

Intelligent Systems Laboratory

University of Bristol



Summary

- After this tutorial, you will be able to
  - [model evaluation] produce ROC plots for categorical and ranking classifiers and calculate their AUC; apply cross-validation in doing so;
  - [model selection] use the ROC convex hull method to select among categorical classifiers; determine the optimal decision threshold for a ranking classifier;
  - [metrics] analyse a variety of machine learning metrics by means of ROC isometrics; understand fundamental properties such as skew-sensitivity and equivalence between metrics;
  - [model construction] appreciate that one model can be many models from a ROC perspective; use ROC analysis to improve a model's AUC;
  - [multi-class ROC] understand multi-class approximations such as the MAUC metric and calibration of multi-class probability estimators.

Summary

- After this tutorial, you will be able to
  - [model evaluation] produce ROC plots for categorical and ranking classifiers and calculate their AUC; apply cross-validation in doing so; a ROC perspective; use ROC analysis to improve a model's AUC;
  - [multi-class ROC] understand multi-class approximations such as the MAUC metric and calibration of multi-class probability estimators.

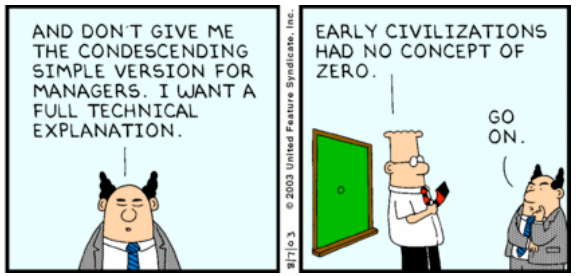
Take-home messages

- It is almost always a good idea to distinguish performance between classes.
- ROC analysis is not just about ‘cost-sensitive learning’, but more generally about how to properly take account of operating conditions.
- Ranking is a more fundamental notion than classification.
- Different metrics say different things about performance, but can be translated into expected loss as a ‘common currency’.

Outline

- Part I: Fundamentals
  - categorical classification: ROC plots, random selection between models, the ROC convex hull, iso-accuracy lines
  - ranking: ROC curves, concavities, the AUC metric, turning rankers into classifiers, calibration, averaging
  - alternatives: PN plots, precision-recall curves, cost curves
- Part II: A broader view
  - understanding ML metrics: isometrics, basic types of linear isometric plots, linear metrics and equivalences between them, non-linear metrics, skew-sensitivity
  - model manipulation: obtaining new models without re-training, ordering decision tree branches and rules, repairing concavities, locally adjusting rankings
  - multi-class ROC: multi-objective optimisation and the Pareto front, calibrating multi-class probability estimators
- Part III: Comparing machine learning metrics
  - Brier score, threshold selection methods, expected loss, ROL plots, rate-driven cost curve

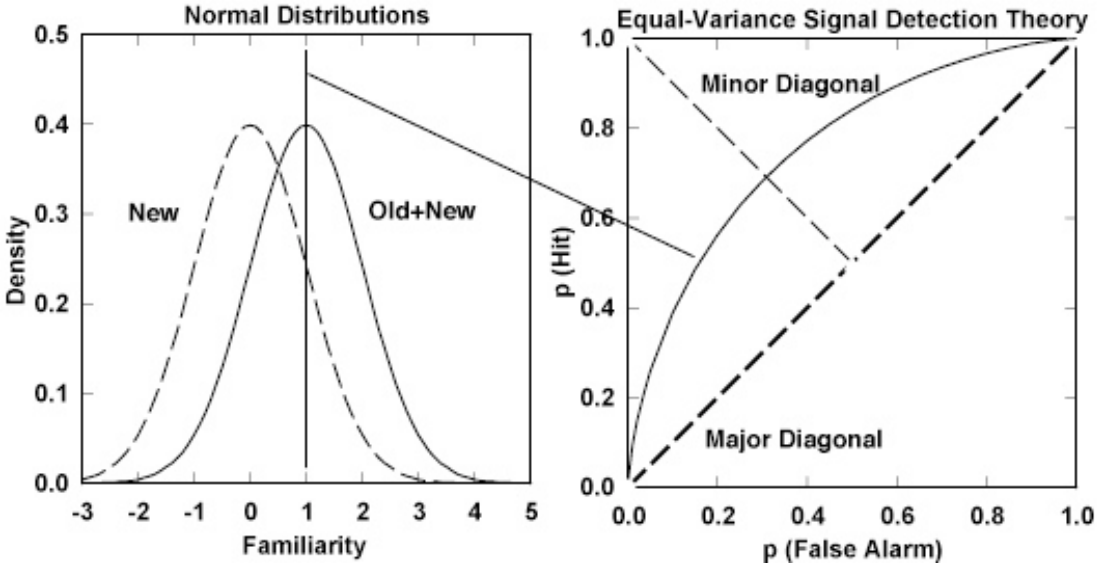
Part I: Fundamentals



- Categorical classification:
  - ROC plots
  - random selection between models
  - the ROC convex hull
  - iso-accuracy lines
- Ranking:
  - ROC curves
  - the AUC metric
  - turning rankers into classifiers
  - calibration
- Alternatives:
  - PN plots
  - precision-recall curves
  - cost curves

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

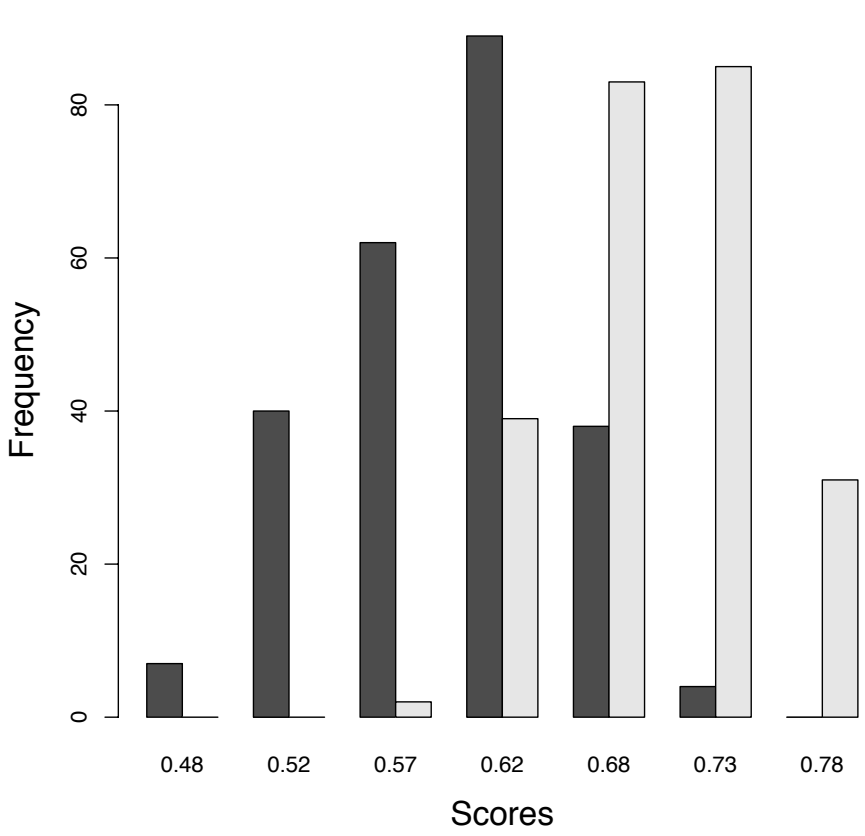
- slope of ROC curve is equal to likelihood ratio $\mathrm{LR}(x) = \dfrac{P(x|\text{signal})}{P(x|\text{noise})}$
  - for equal variances the Gaussian model gives

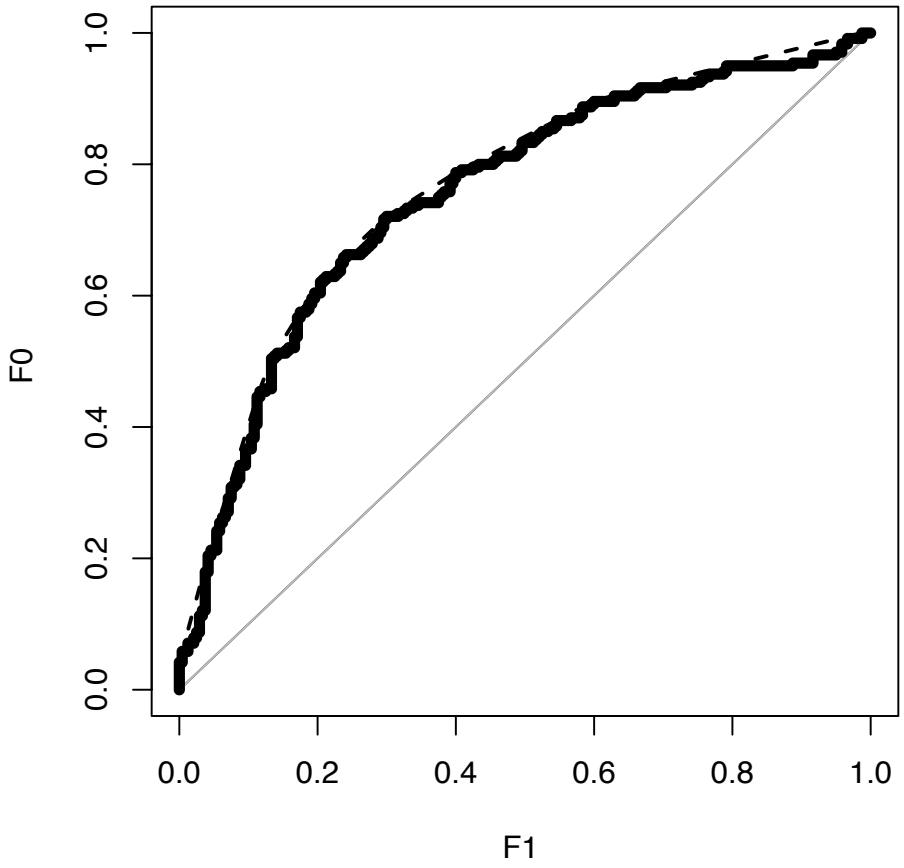
$$\mathrm{LR}(x) = \exp\left(\gamma(x - x_0)\right) \qquad \gamma = \frac{\mu_{\text{signal}} - \mu_{\text{noise}}}{\sigma^2} \qquad x_0 = \frac{\mu_{\text{signal}} + \mu_{\text{noise}}}{2}$$

  - so LR(x) increases monotonically with x and ROC curve is convex
  - optimal decision threshold is t such that $\mathrm{LR}(t) = \dfrac{P(\text{noise})}{P(\text{signal})}$
  - for uniform prior this gives t = x_0 which means this threshold picks the point where the ROC curve intersects with the descending diagonal.

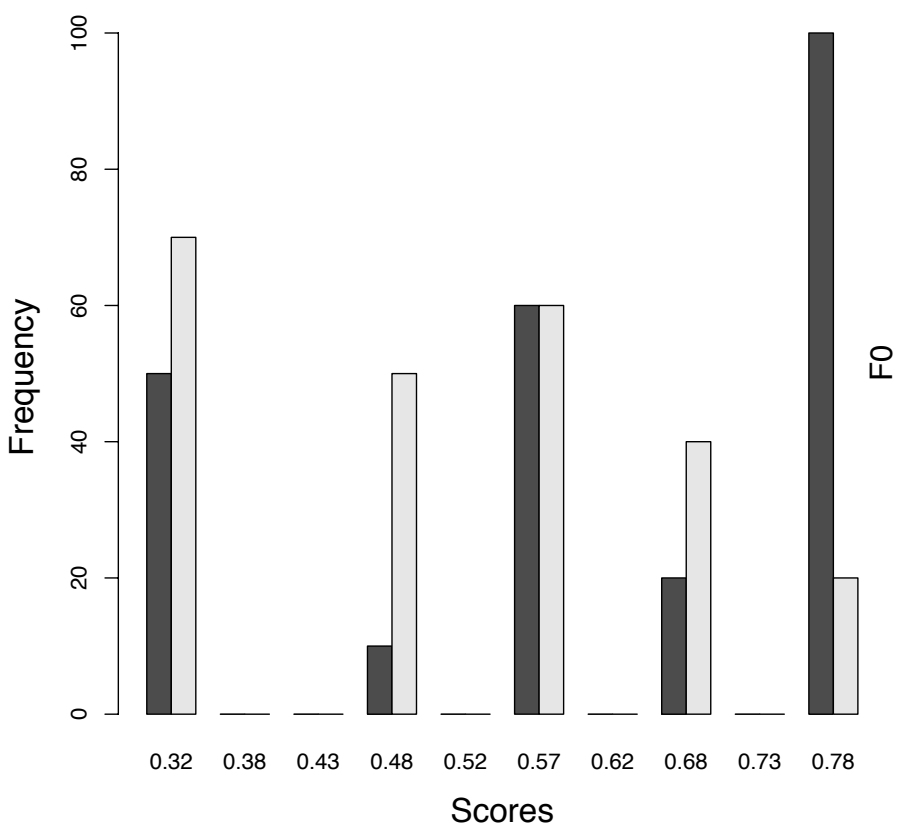
- concavities occur with unequal variances

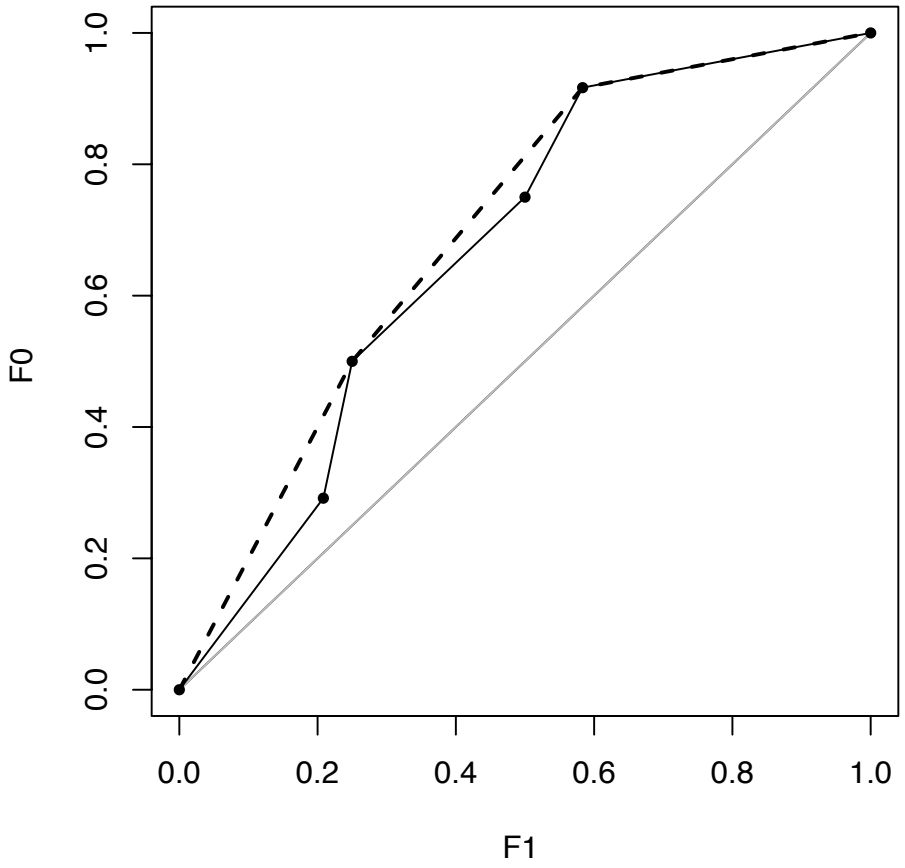
From score distributions to ROC curves





From score distributions to ROC curves





ROC analysis for fixed-threshold classifiers

- Based on contingency table or confusion matrix

	Predicted ⊕	*Predicted* ⊖	
Actual ⊕	**# true positives**	**# false negatives**	# positives
Actual ⊖	**# false positives**	**# true negatives**	# negatives
	# positive predictions	# negative predictions	

- 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

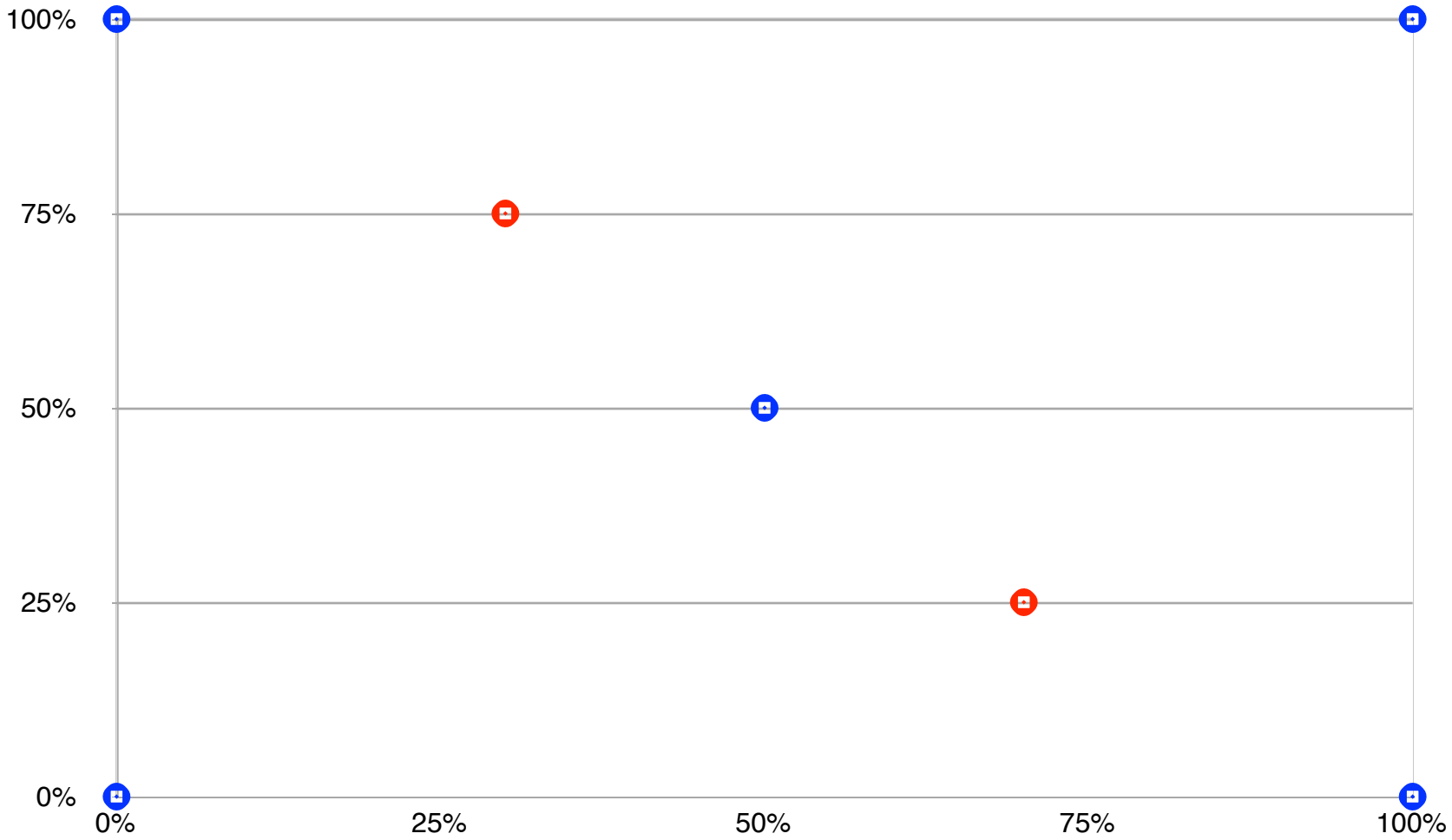
- True positive rate tpr = TP/Pos = TP/(TP+FN)
  - fraction of positives correctly predicted
- False positive rate fpr = FP/Neg = FP/(FP+TN)
  - fraction of negatives incorrectly predicted
  - = 1 – true negative rate TN/(FP+TN)
- Accuracy

$$acc = \frac{TP + TN}{Pos + Neg}$$
$$= \frac{TP}{Pos}\frac{Pos}{Pos + Neg} + \frac{TN}{Neg}\frac{Neg}{Pos + Neg}$$
$$= pos \cdot tpr + neg \cdot (1 - fpr)$$

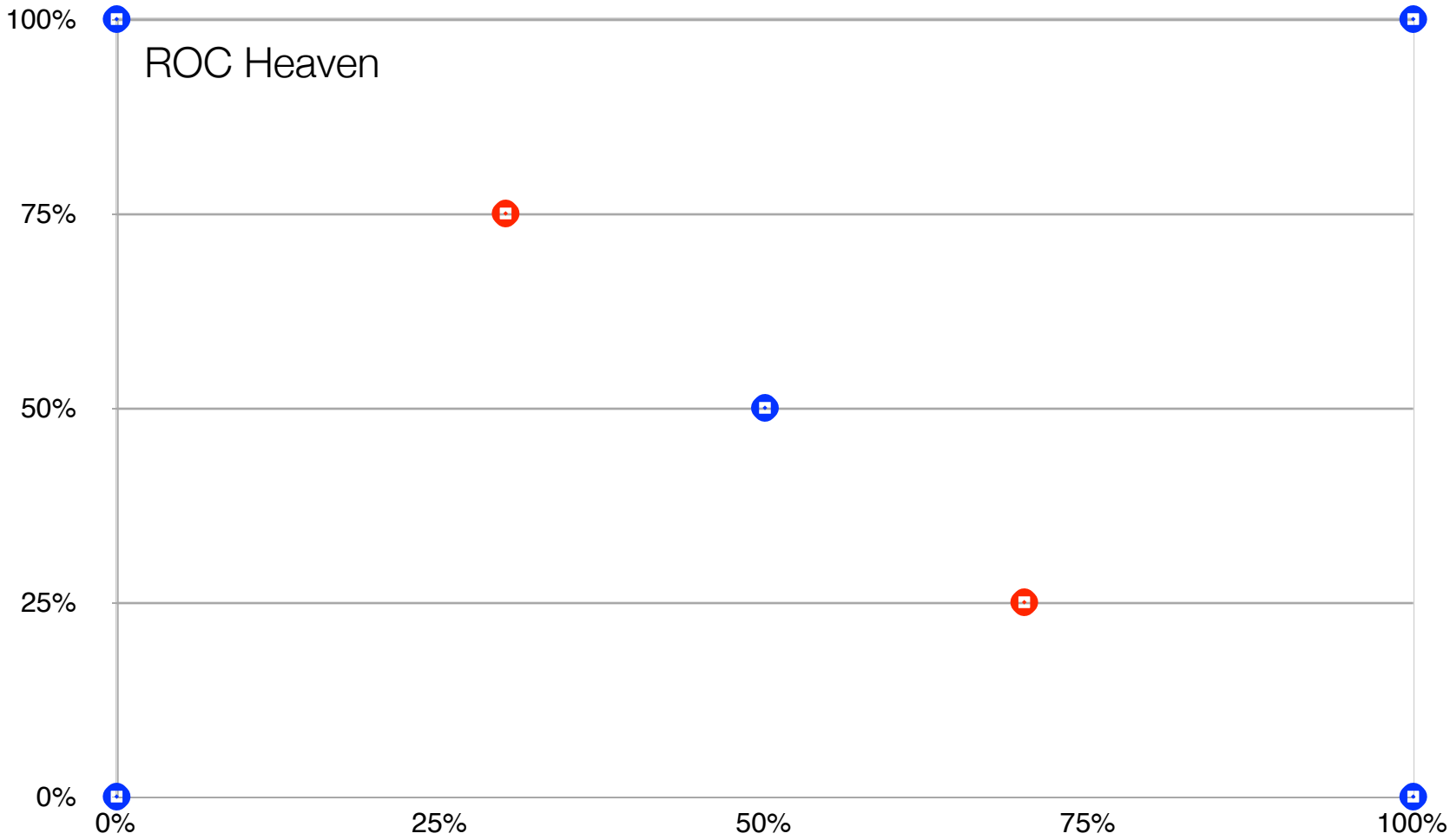
  - weighted average of true positive and true negative rates

	Predicted ⊕	*Predicted* ⊖	
Actual ⊕	**TP**	**FN**	Pos
Actual ⊖	**FP**	**TN**	Neg
	PPos	PNeg	

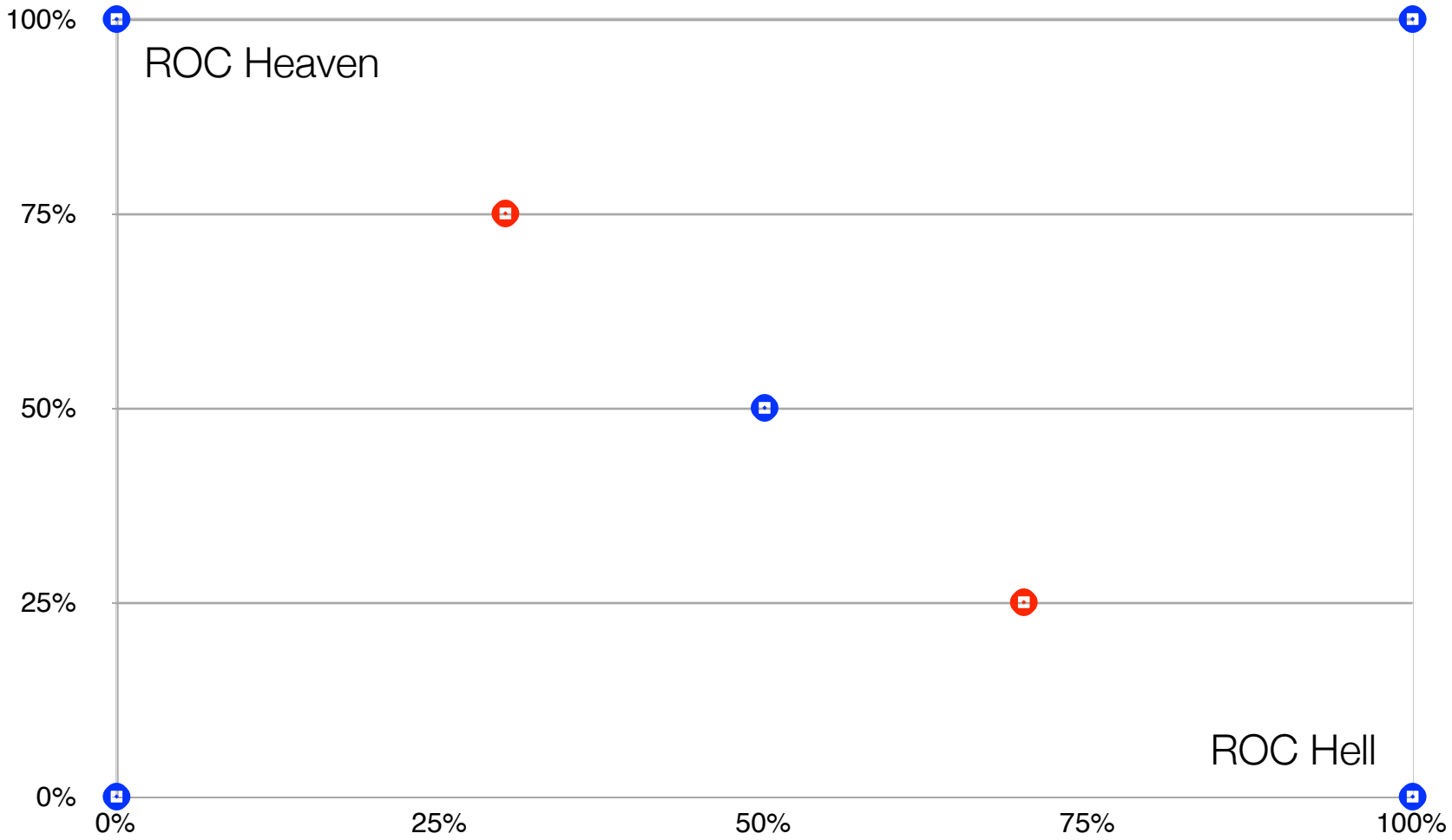
A closer look at ROC space



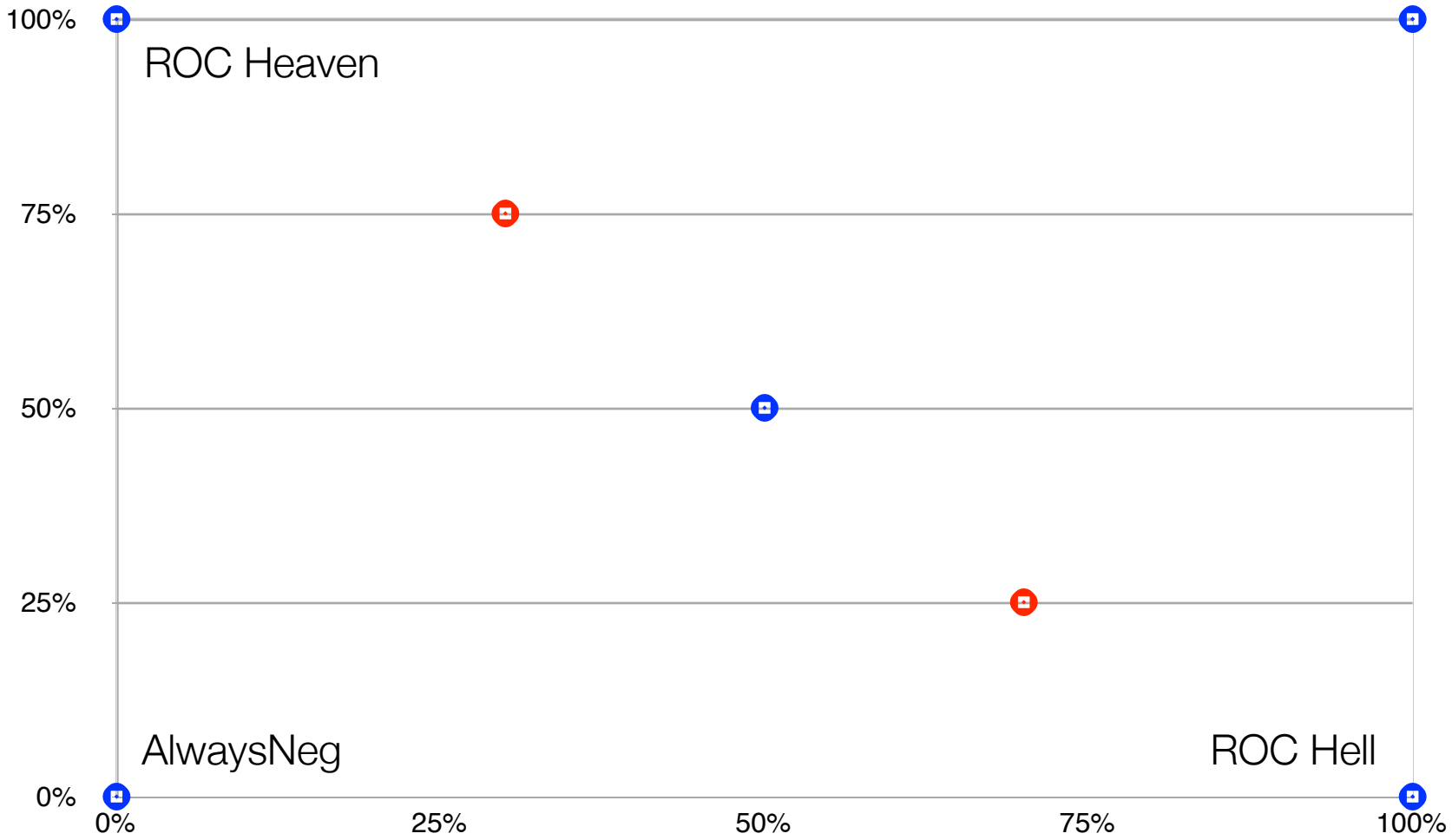
A closer look at ROC space



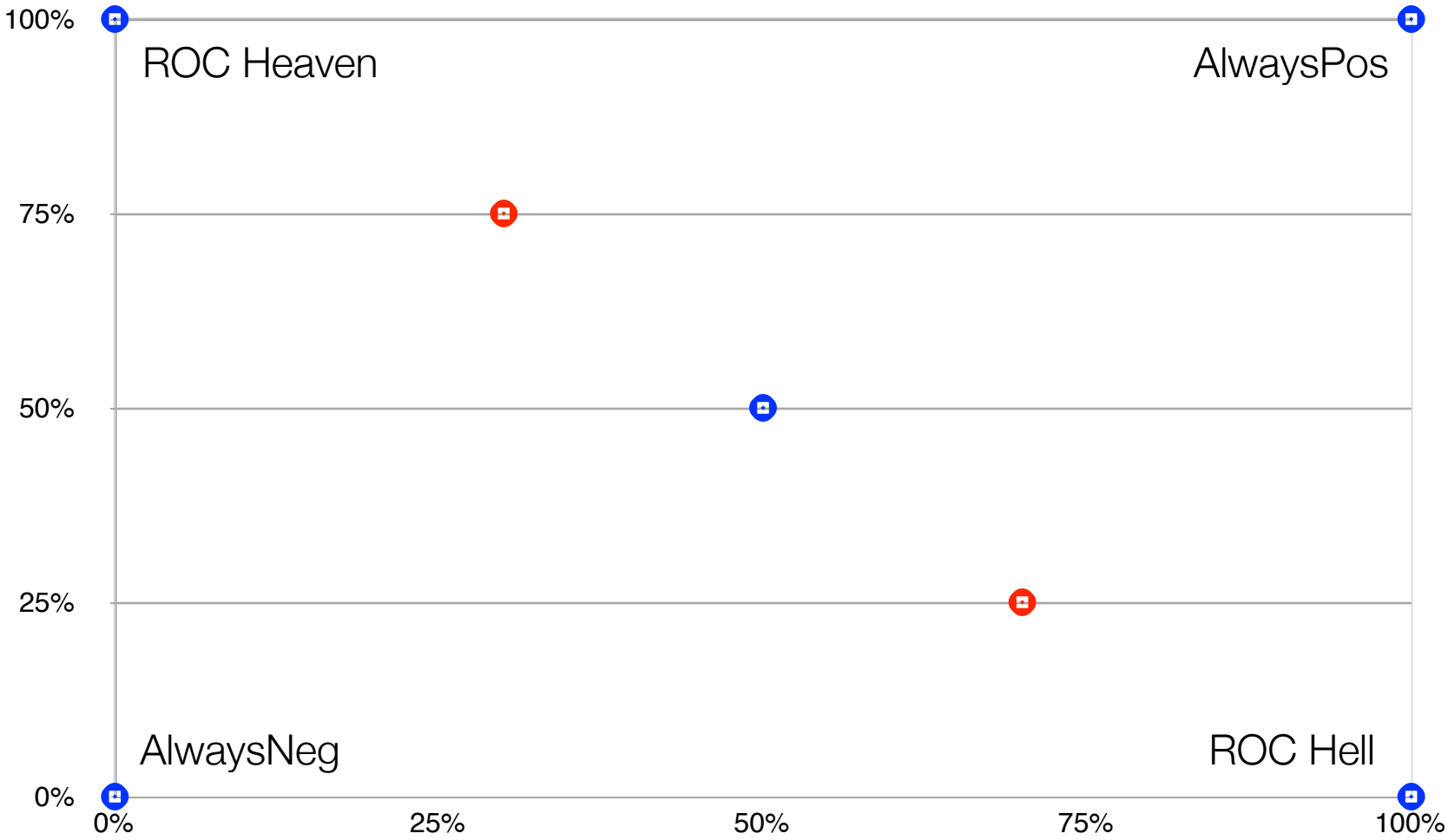
A closer look at ROC space



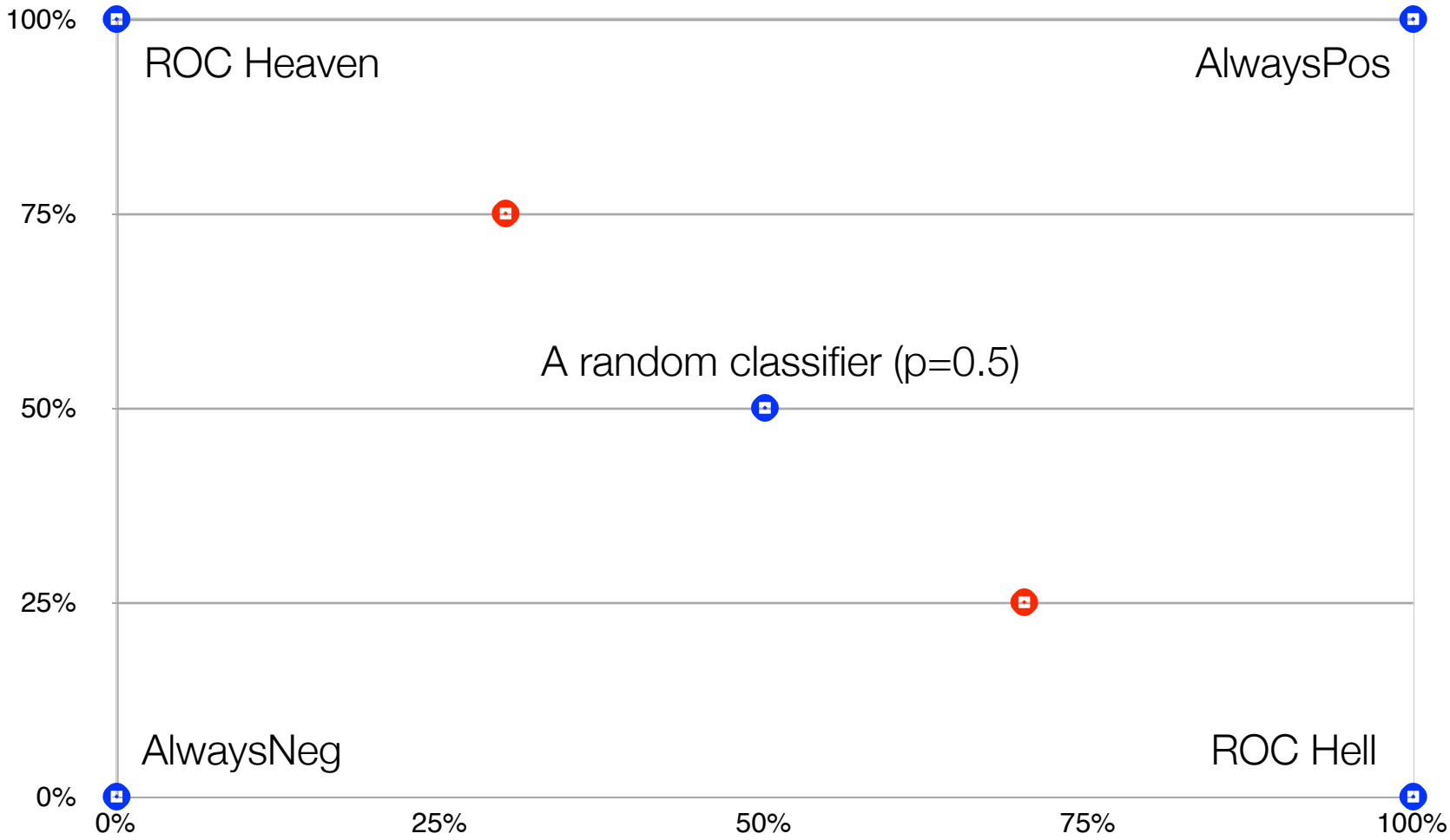
A closer look at ROC space



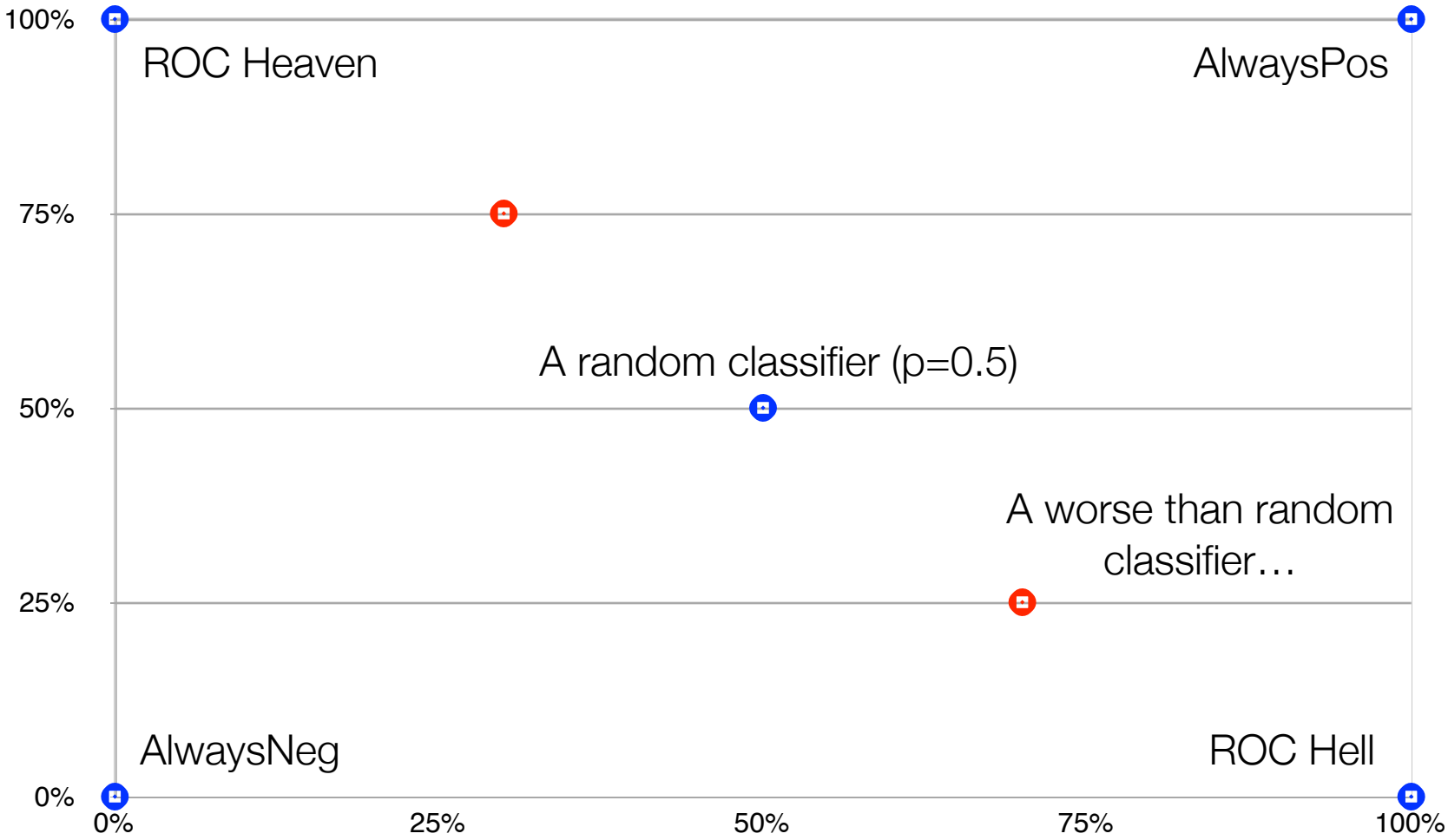
A closer look at ROC space



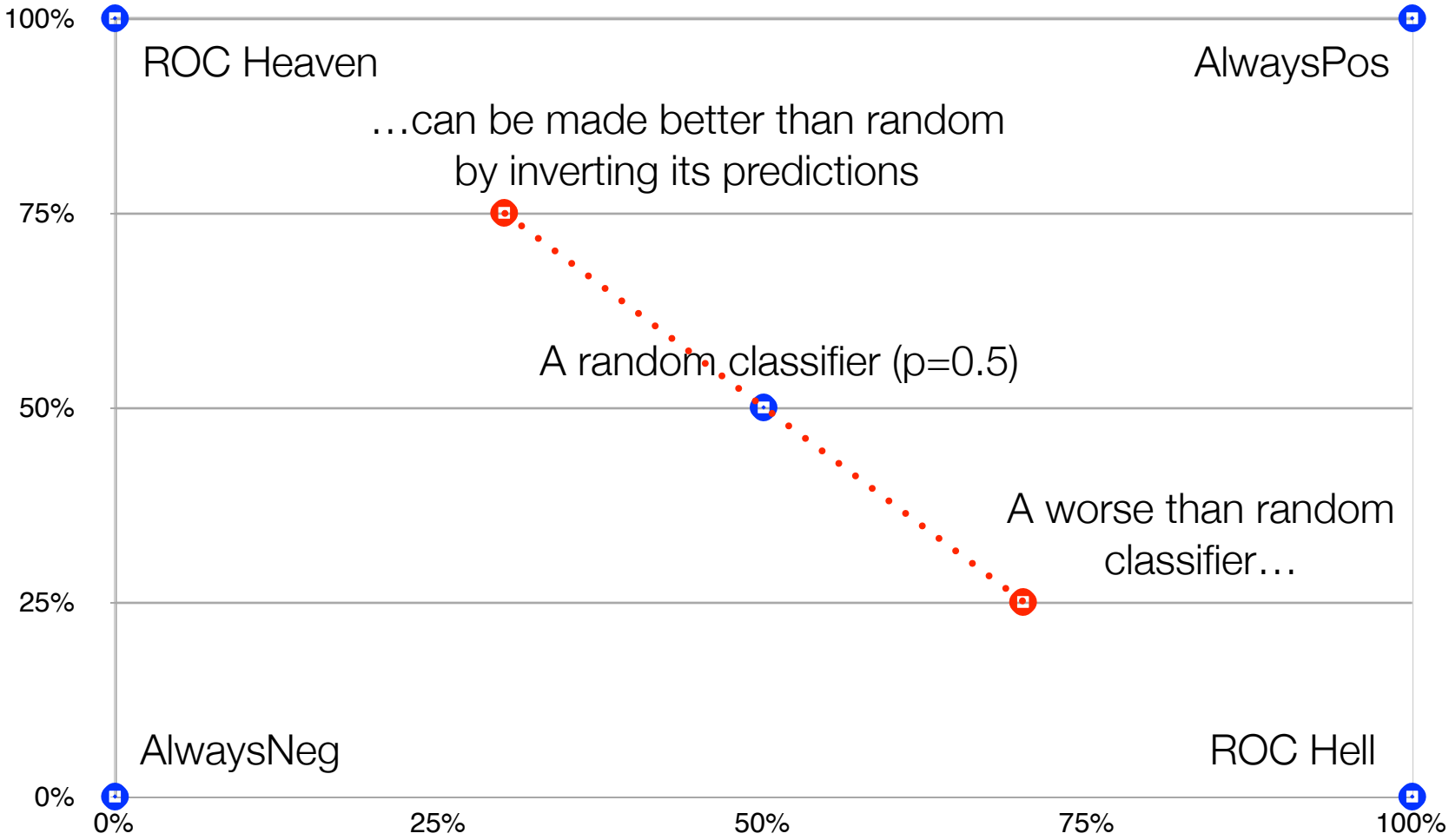
A closer look at ROC space



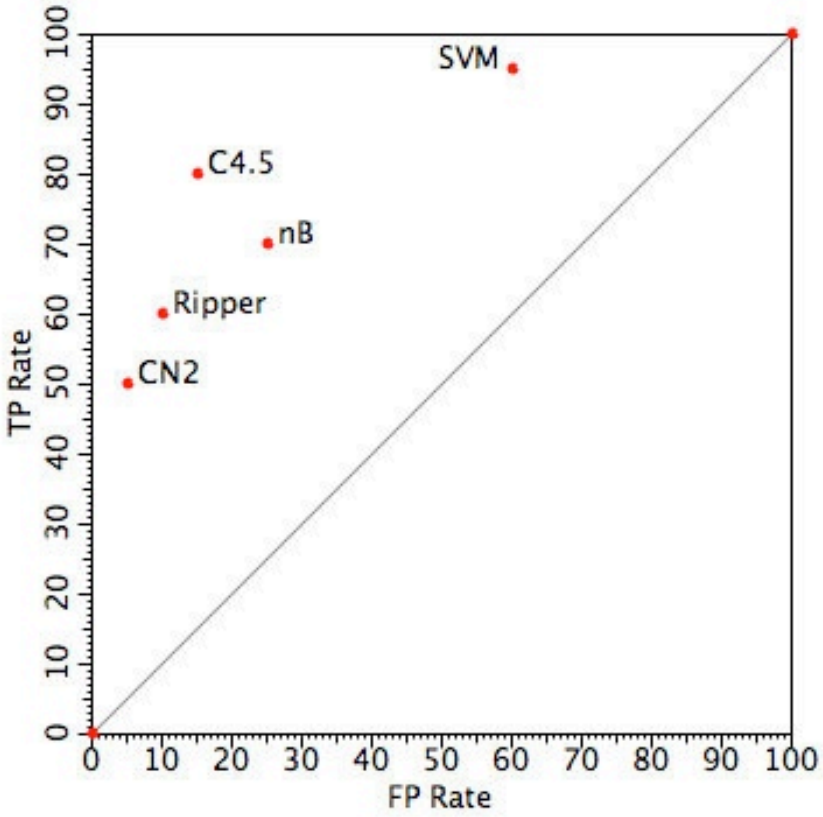
A closer look at ROC space



A closer look at ROC space

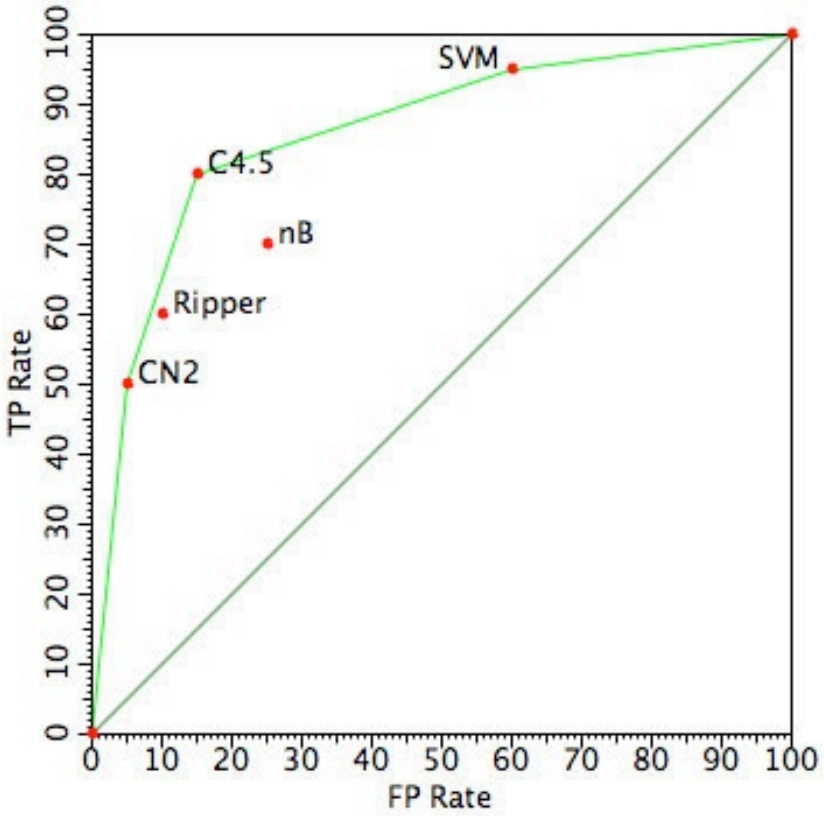


Example ROC plot



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.

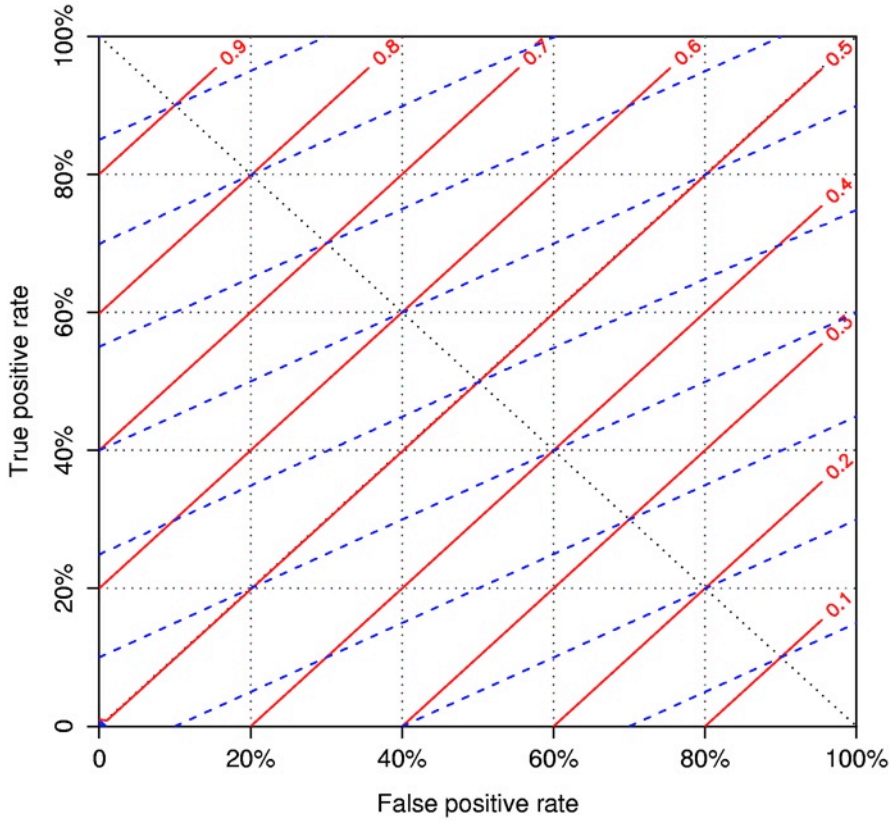
Classifiers below the convex hull are always sub-optimal

Why is the convex hull a curve?

- Any performance on a line segment connecting two ROC points can be achieved by randomly choosing between them
  - the ascending default performance diagonal is just a special case
- The classifiers on the ROC convex hull can be combined to form the ROCCH-hybrid (Provost & Fawcett, 2001)
  - ordered sequence of classifiers
  - can be turned into a ranker
    - as with decision trees, see later

Iso-accuracy lines

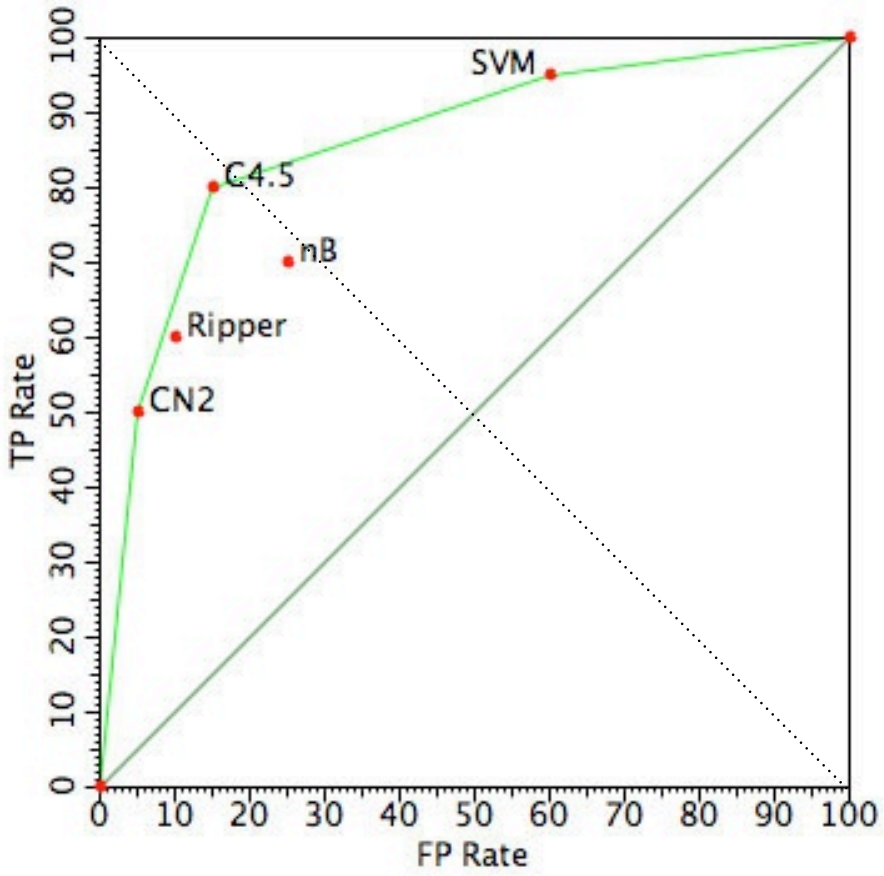
- Iso-accuracy line connects ROC points with the same accuracy
  - $pos \cdot tpr + neg \cdot (1 - fpr) = a$
  - $tpr = \frac{a - neg}{pos} + \frac{neg}{pos} \cdot fpr$
- Parallel ascending lines with slope neg/pos
  - higher lines are better
  - on descending diagonal, tpr = a



Iso-accuracy & convex hull

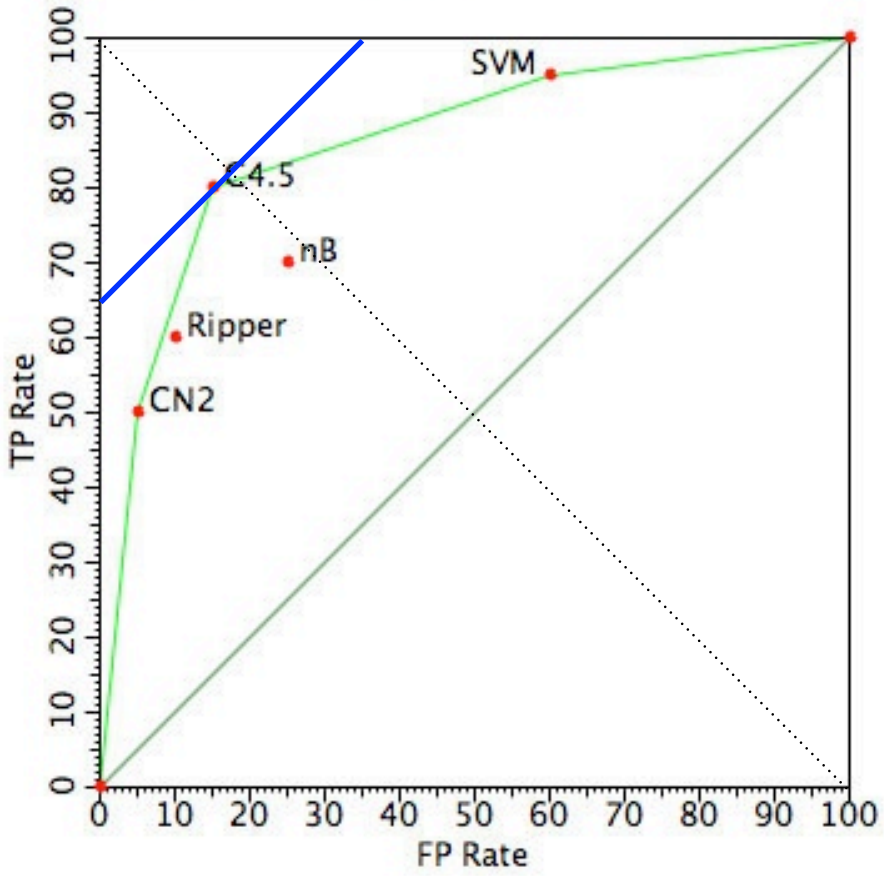
- Each line segment on the convex hull is an iso-accuracy line for a particular class distribution
  - under that distribution, the two classifiers on the end-points achieve the same accuracy
  - for distributions skewed towards negatives (steeper slope), the left one is better
  - for distributions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



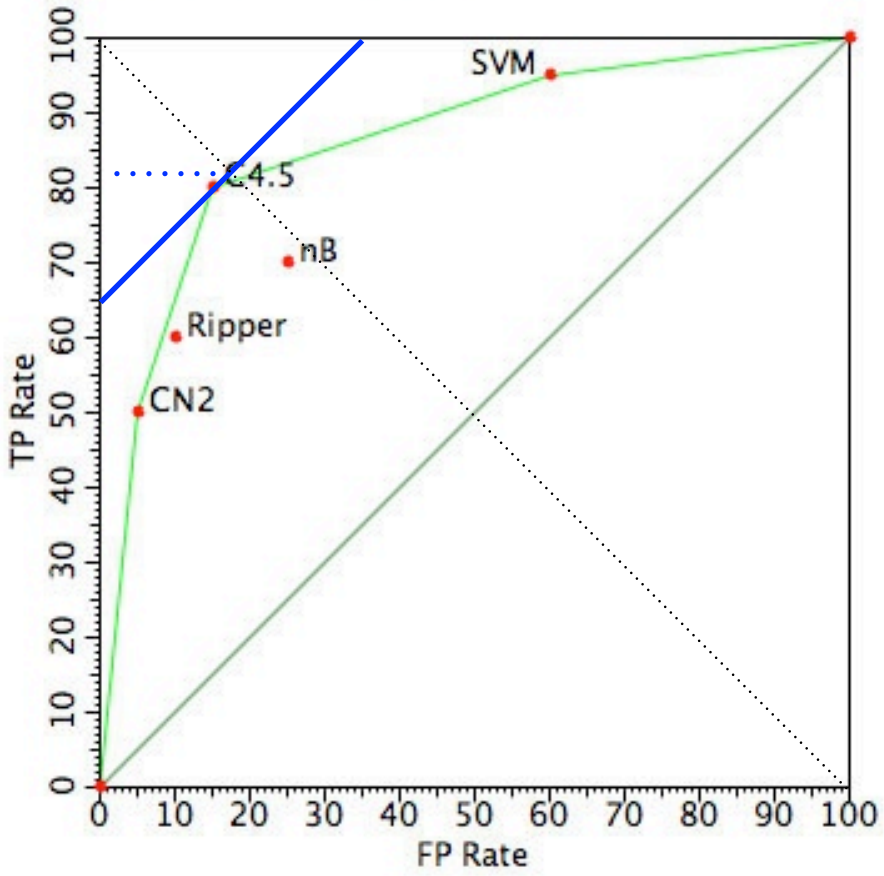
For uniform class distribution, C4.5 is optimal and achieves about 82% accuracy.

Selecting the optimal classifier



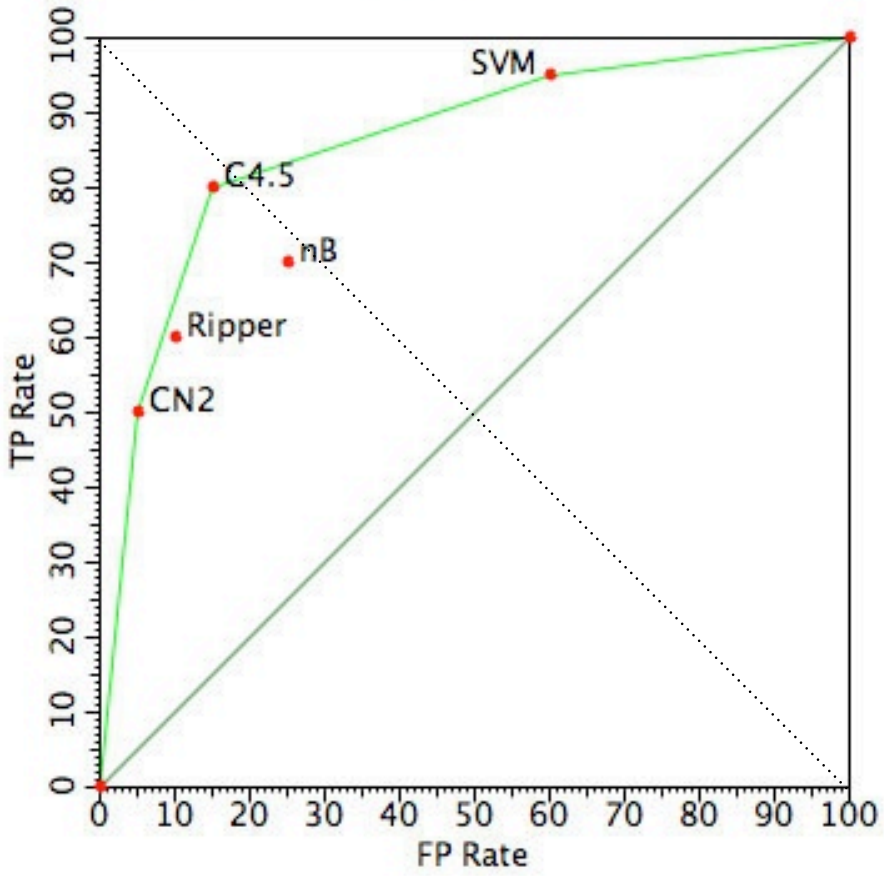
For uniform class distribution, C4.5 is optimal and achieves about 82% accuracy.

Selecting the optimal classifier



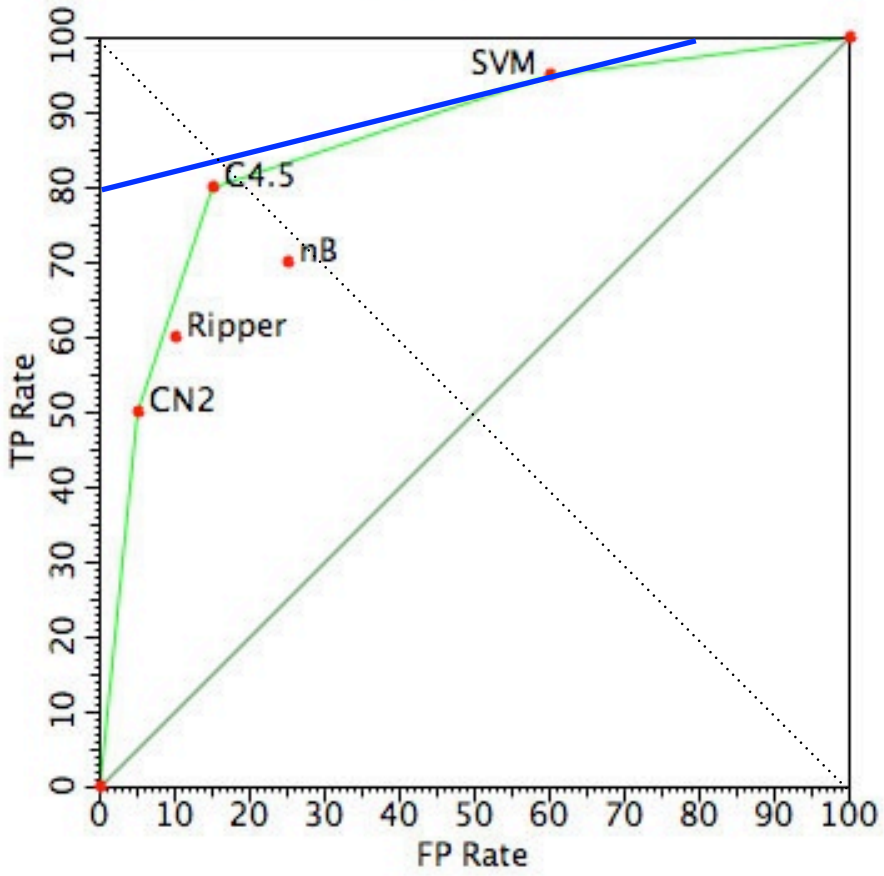
For uniform class distribution, C4.5 is optimal and achieves about 82% accuracy.

Selecting the optimal classifier



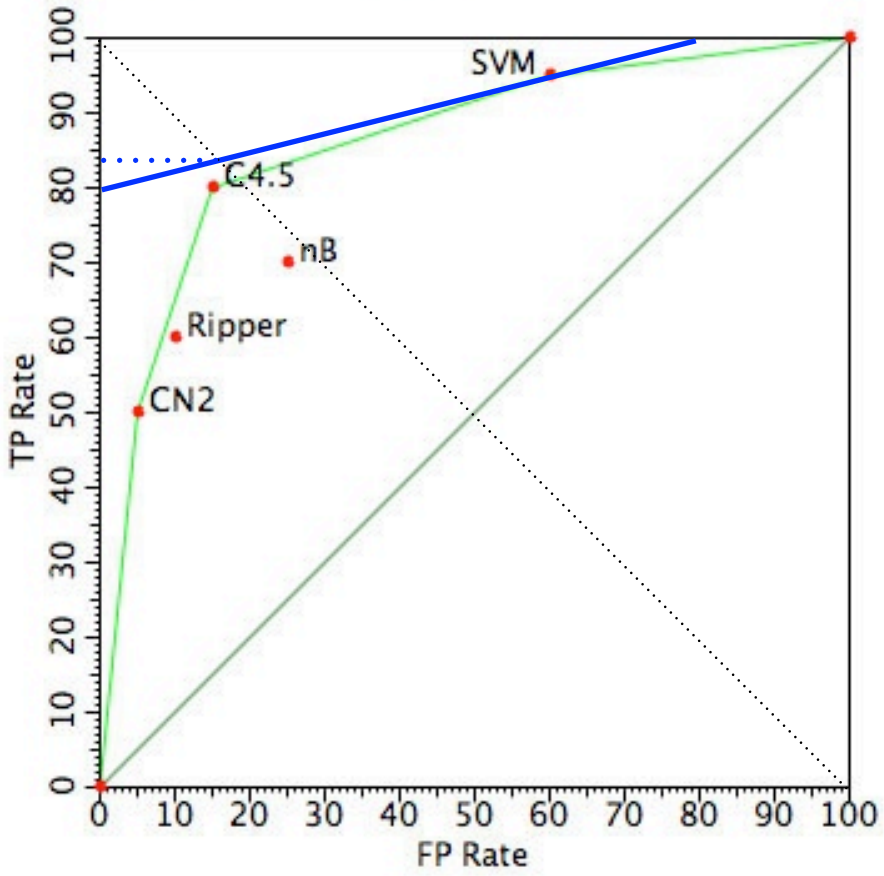
With four times as many +ves as –ves, SVM is optimal and achieves about 84% accuracy.

Selecting the optimal classifier



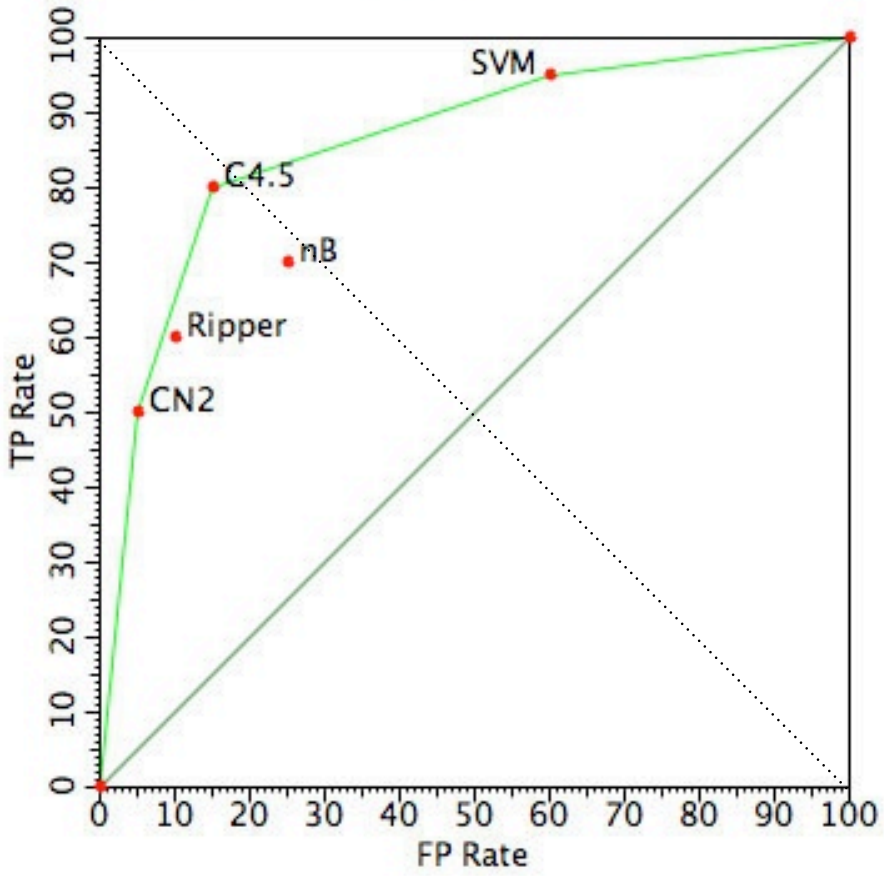
With four times as many +ves as –ves, SVM is optimal and achieves about 84% accuracy.

Selecting the optimal classifier



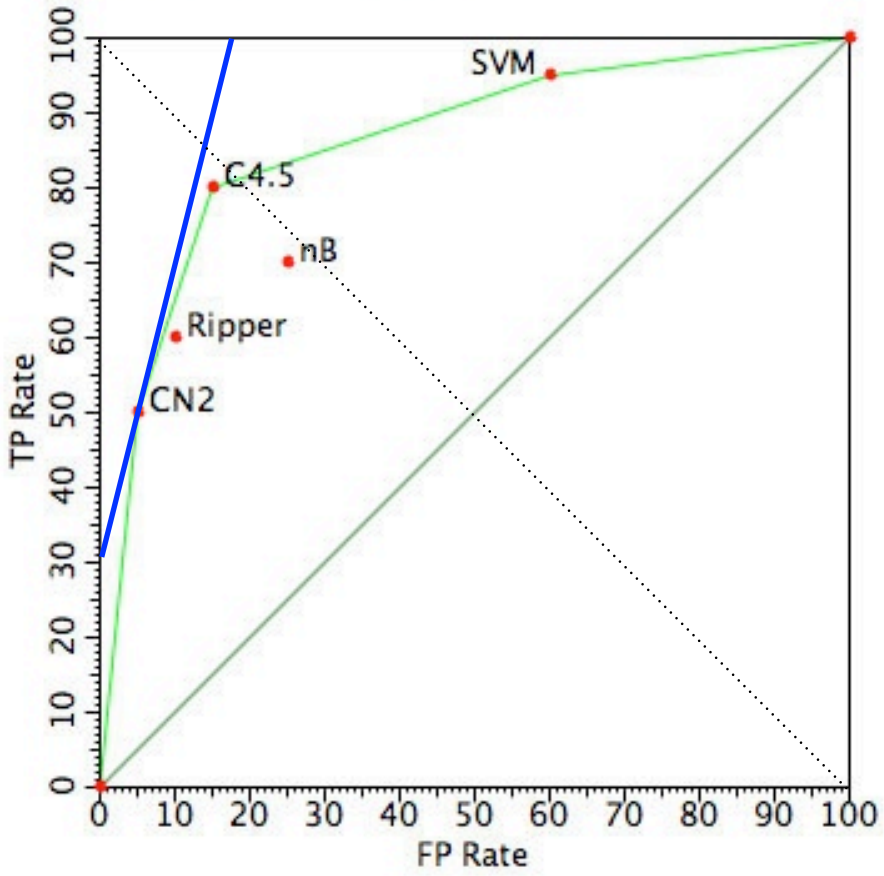
With four times as many +ves as –ves, SVM is optimal and achieves about 84% accuracy.

Selecting the optimal classifier



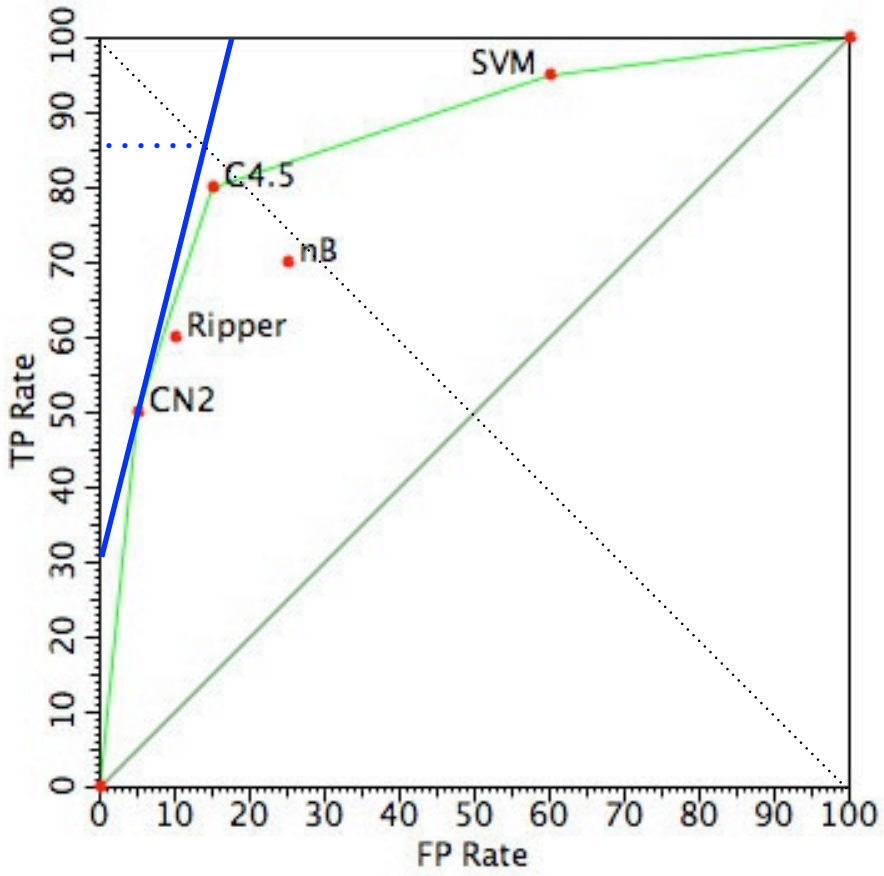
With four times as many –ves as +ves, CN2 is optimal and achieves about 86% accuracy

Selecting the optimal classifier



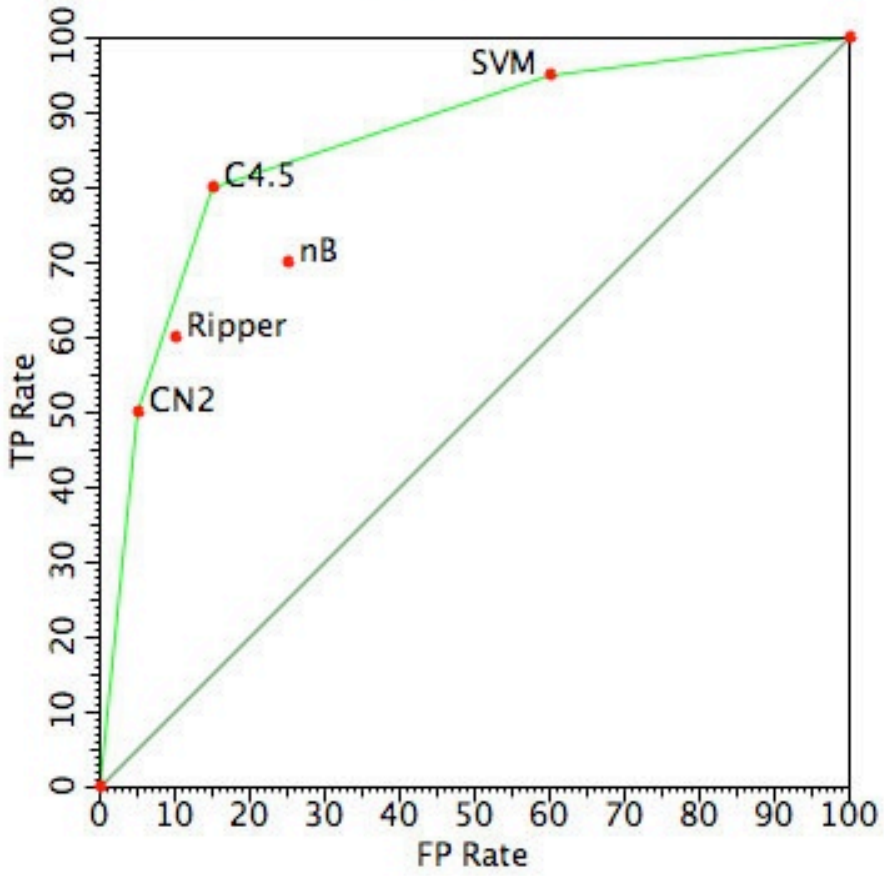
With four times as many –ves as +ves, CN2 is optimal and achieves about 86% accuracy

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 (negative costs):
  - 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

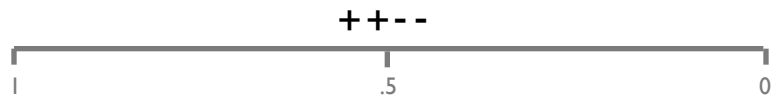
- From a decision-making perspective, the cost matrix has one degree of freedom
  - need full cost matrix to determine absolute yield
- There is no reason to distinguish between cost skew and class skew
  - skew ratio expresses relative importance of negatives vs. positives
- ROC analysis deals with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

ROC analysis for scoring classifiers

- A scoring classifier outputs scores f(x,+) and f(x,–) for each class
  - e.g. estimate class-conditional 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. likelihood ratio P(x|+)/P(x|–)
- Rankers can be turned into classifiers by setting a threshold on f(x)

Classification ≠ ranking ≠ probability estimation

- Better probabilities ≠ better ranking



  - no ranking errors, mean squared error ≈ 0.25

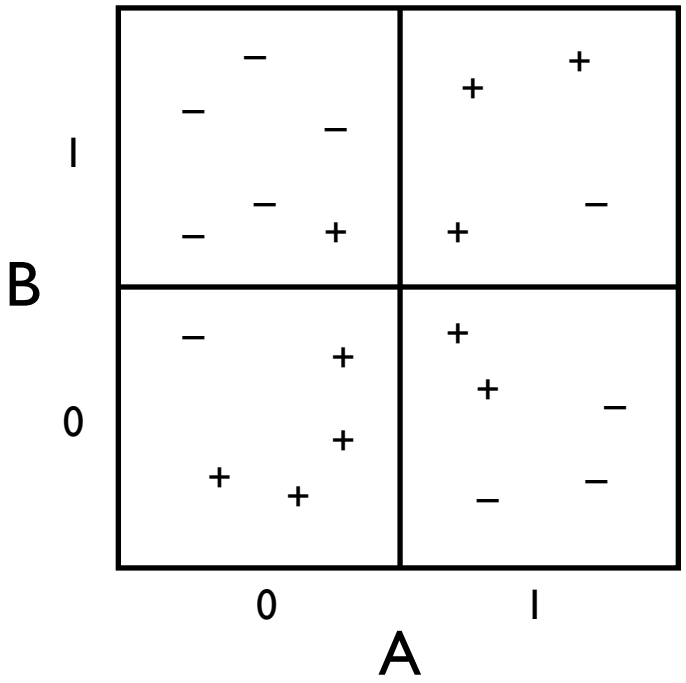
  - 1 ranking error (worse), mean squared error ≈ 0.13 (better)

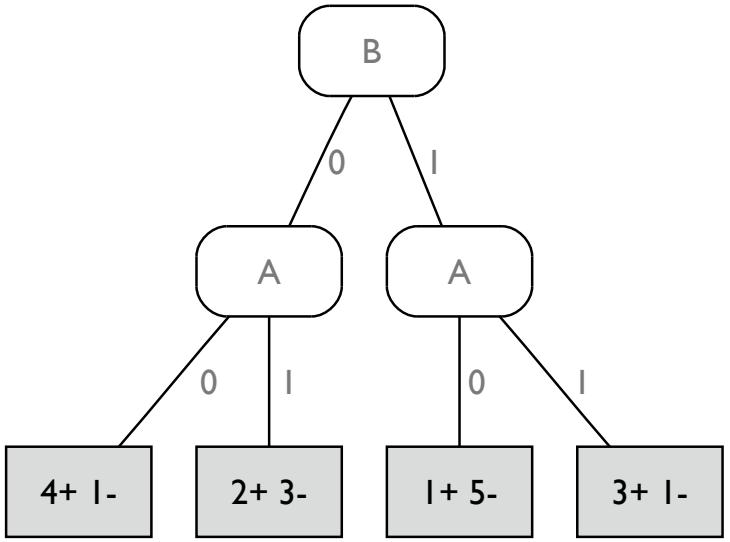
- Better classification ≠ better ranking

  - 4.5 ranking errors, 3 classification errors

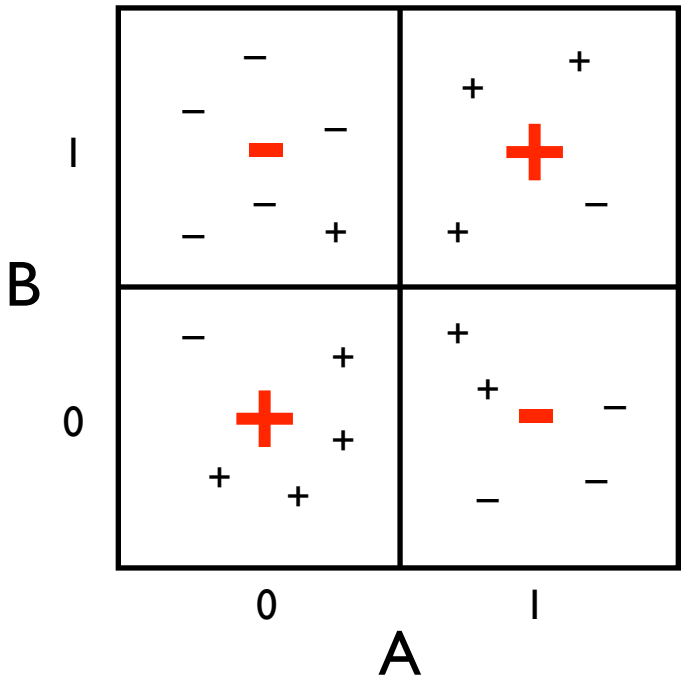
  - 6 ranking errors (worse), 2 classification errors (better)

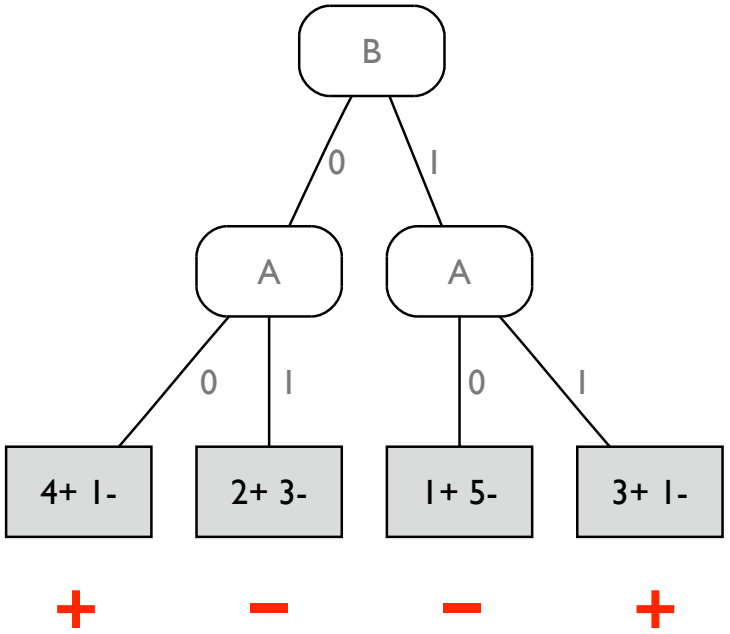
Decision tree classifier





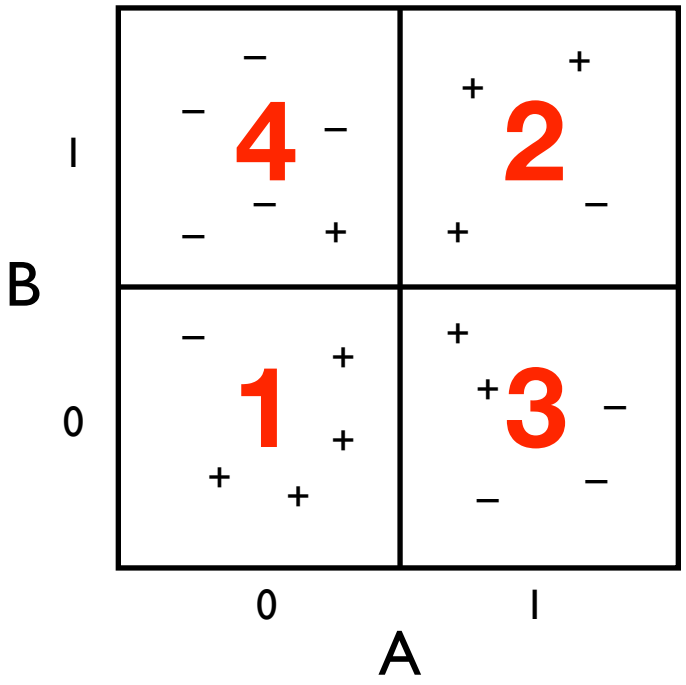
Decision tree classifier

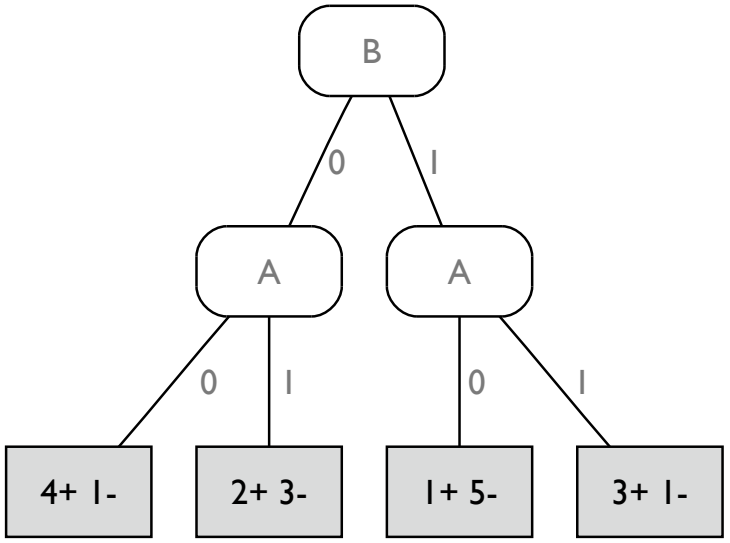




Labels obtained by majority vote decision rule.

Decision tree ranker

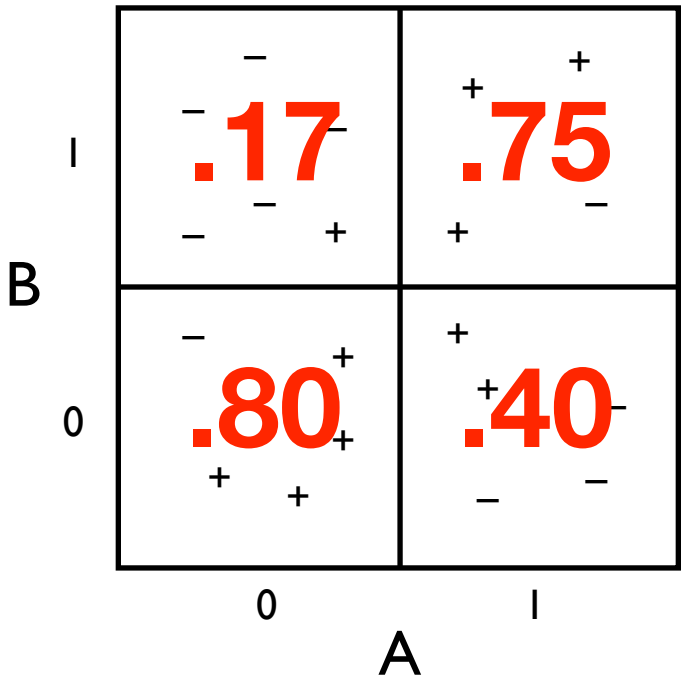


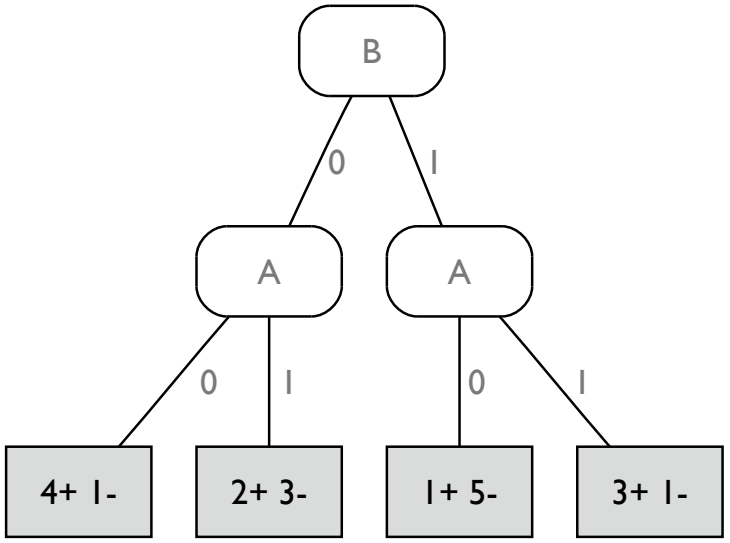


1 3 4 2

4/1 2/3 1/5 3/1

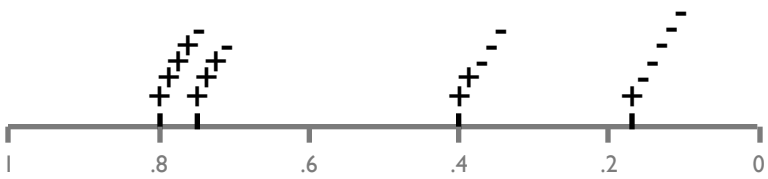
Decision tree probability estimator



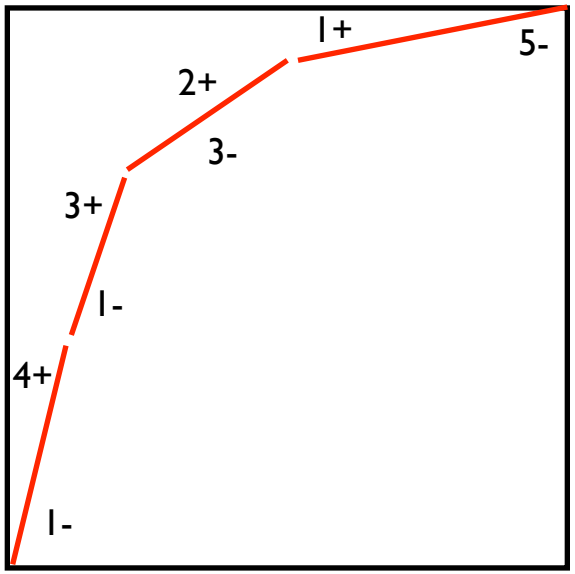


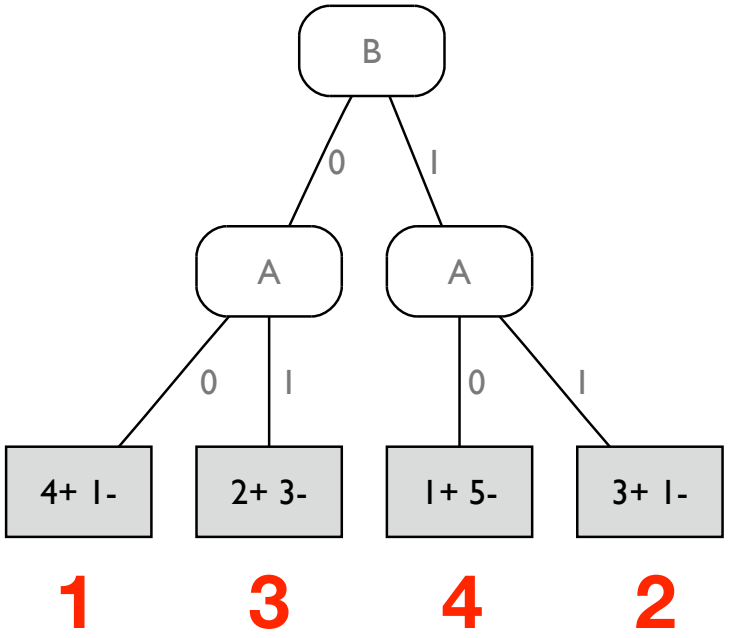
.80 **.40** **.17** **.75**

4/1 **2/3** **1/5** **3/1**



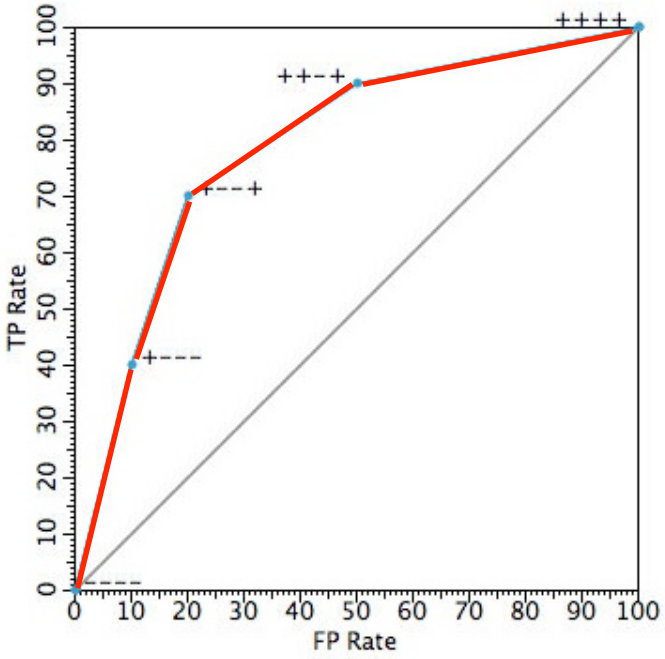
Visualising ranking performance

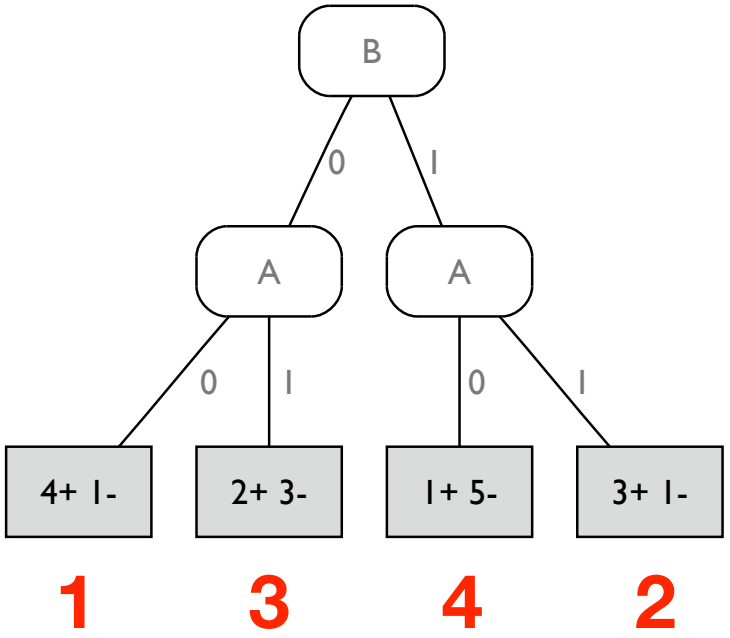




Each leaf is visualised by a line segment; by stacking these line segments in the ranking order we can keep track of cumulative performance (aka Lorenz curve or ROC curve).

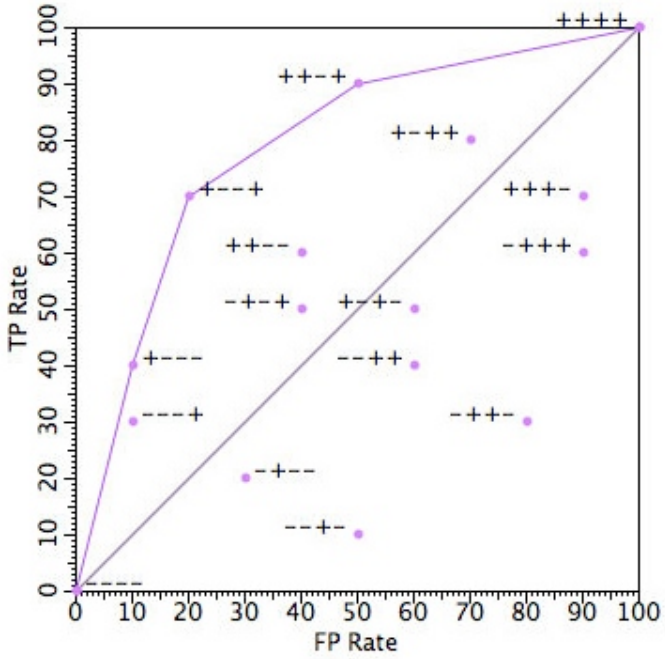
Visualising ranking performance (2)

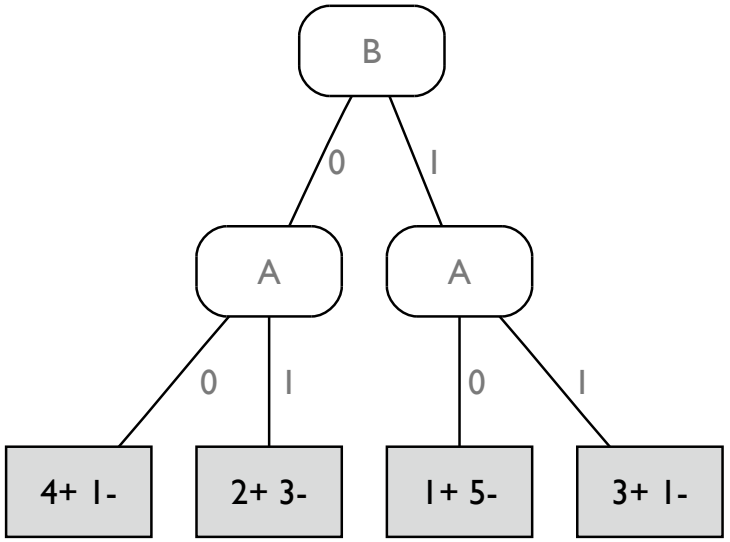




Counts on the axes mean that slopes represent *posterior odds*; normalising these by the number of positives/negatives means that slopes represent *likelihood ratios* instead.

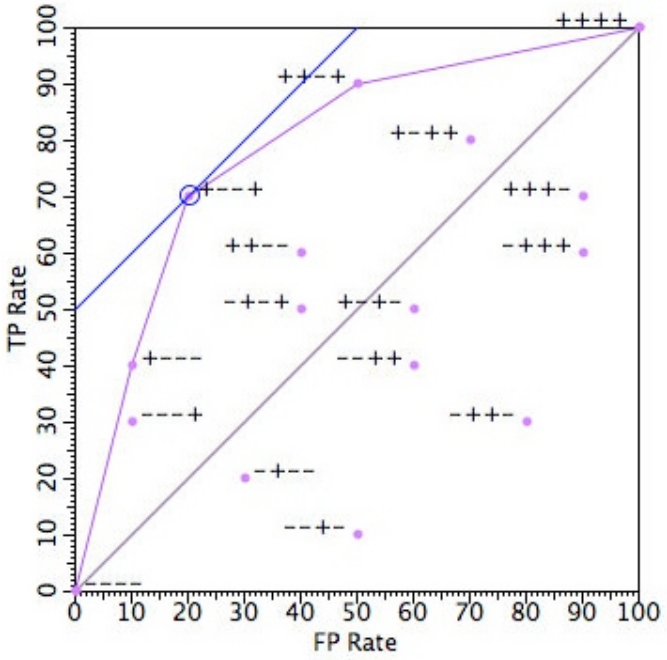
All possible tree labellings

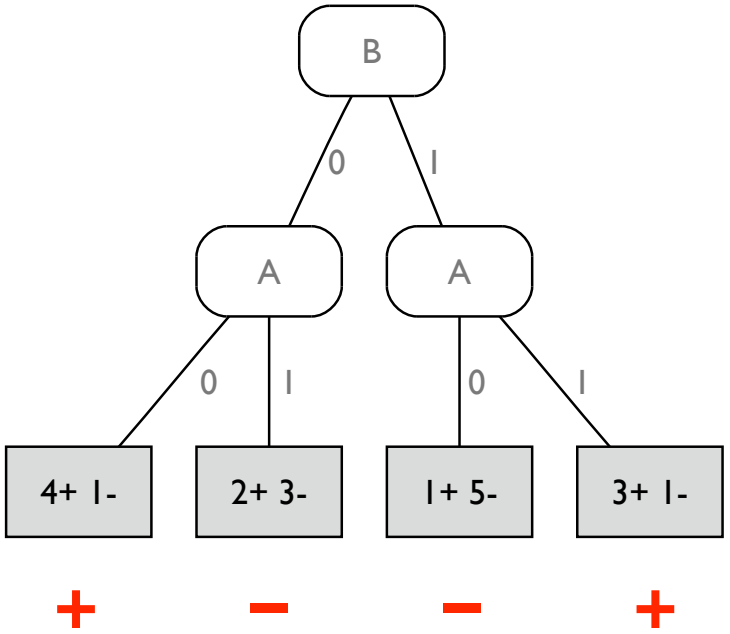




A tree with *n* leaves has 2^n possible labellings, which summarise all possible model behaviours. Notice that a labelling and its opposite (e.g., +—–+ and –++–) are each other's mirror image in ROC space (through (1/2,1/2)).

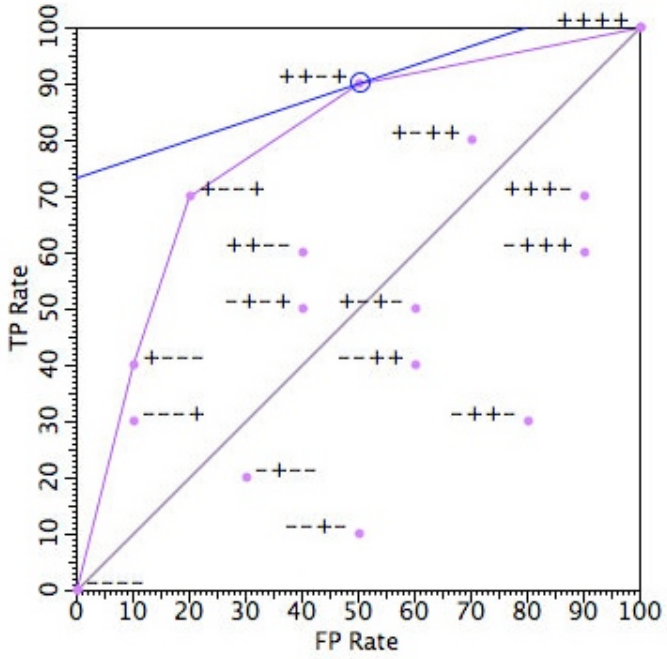
Choosing the optimal labelling

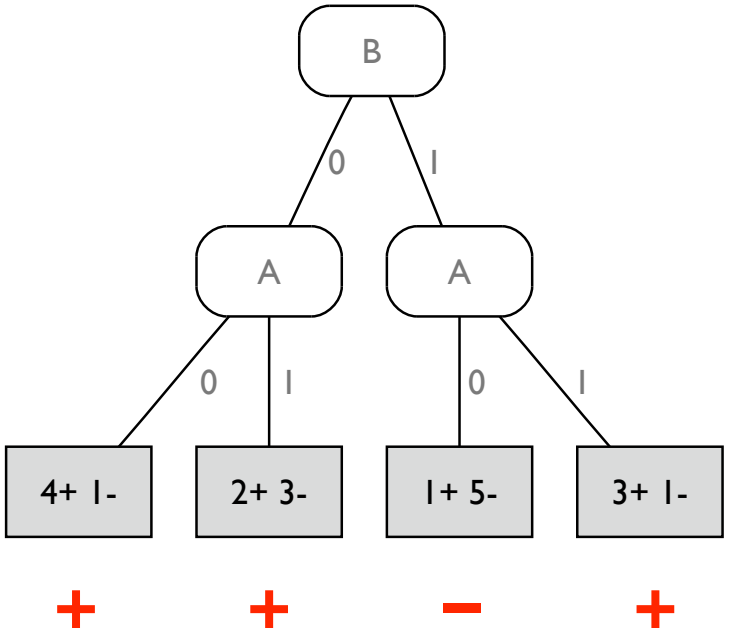




The above labelling is optimal for uniform prior odds (i.e., positives and negatives are equally prevalent/important)

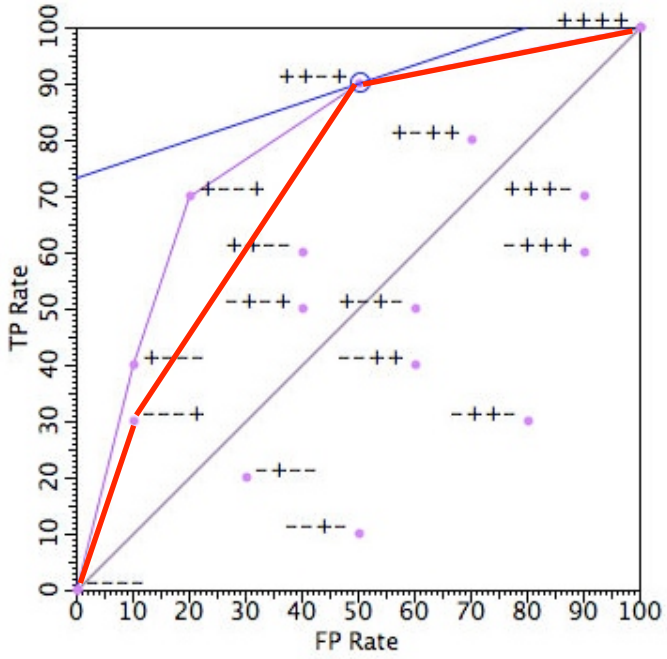
Choosing the optimal labelling (2)

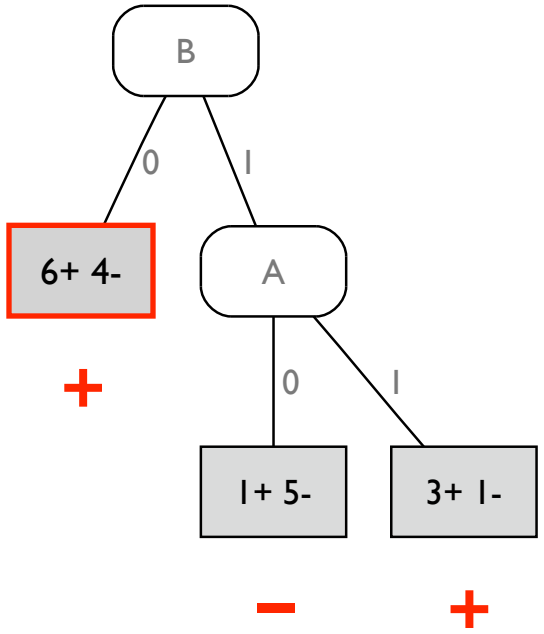




The second leaf is relabelled + if positives are three times as prevalent/important as negatives; notice that this effectively prunes the left subtree.

(aside) Pruning considered harmful...





However, notice that pruning *decreases* ranking performance, as measured by the area under the curve (AUC, see later).

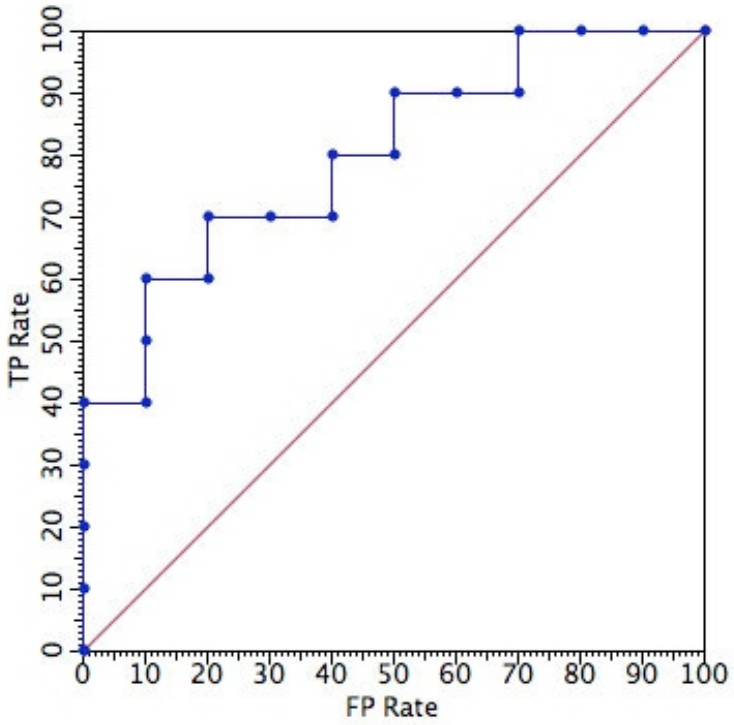
From a ranking to a ROC curve

+ + + + - + + - + - - + - + - - + - - -

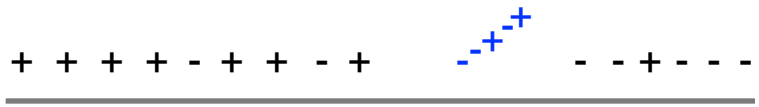
start in (0,0)

get the next instance in the ranking

- if it is positive, move 1/Pos up
- if it is negative, move 1/Neg right



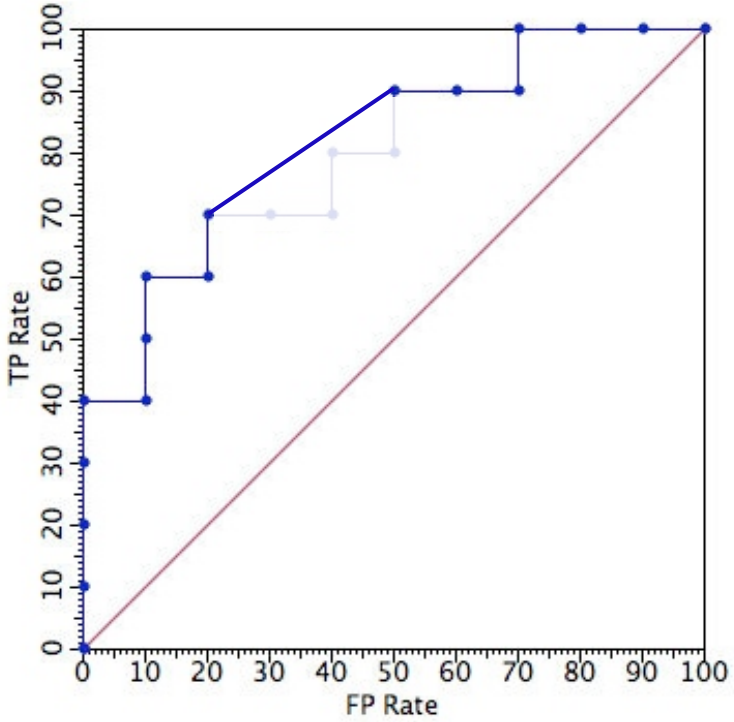
From a ranking to a ROC curve



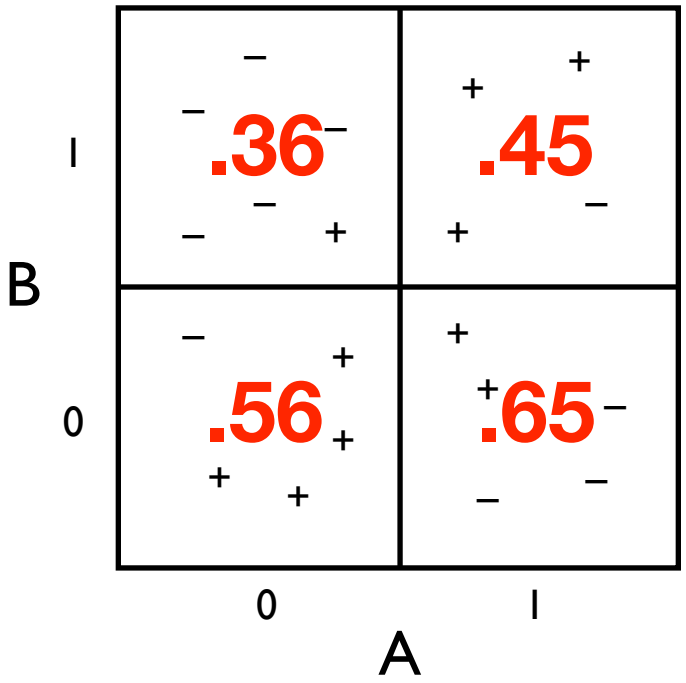
start in (0,0)

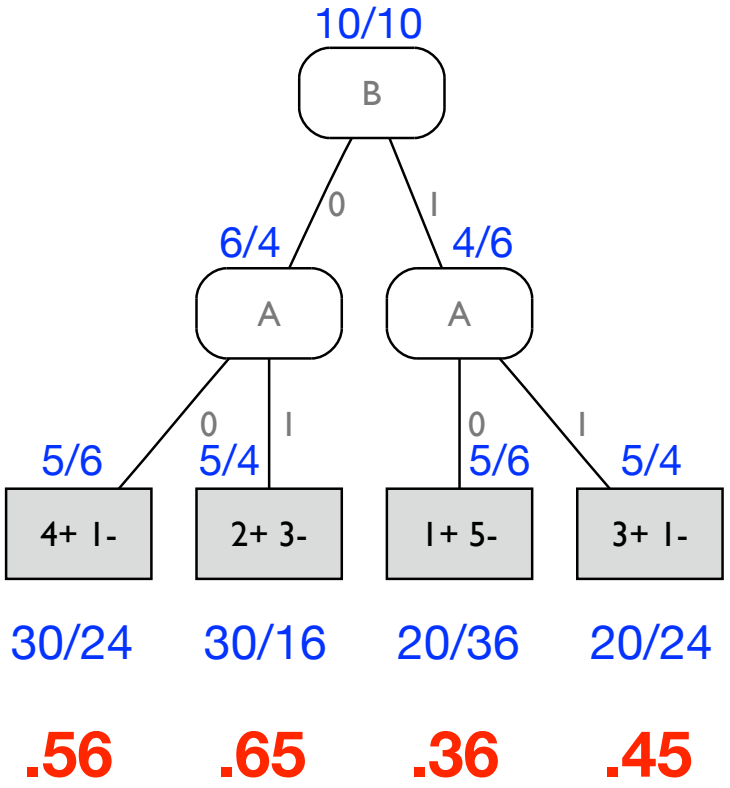
get the next instance in the ranking

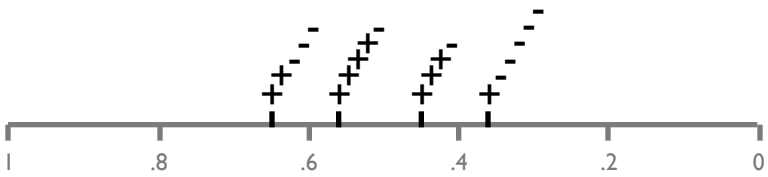
- if it is positive, move 1/Pos up
- if it is negative. move 1/Neg right
- make diagonal move in case of ties



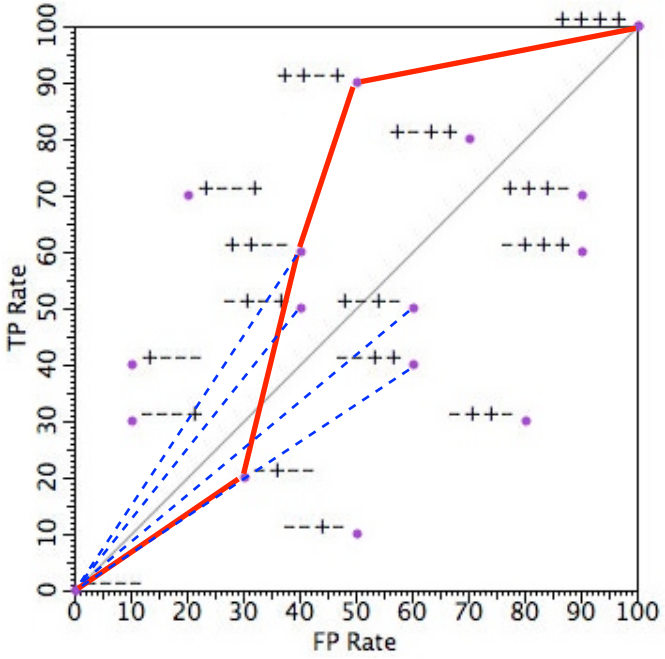
Naive Bayes probability estimator

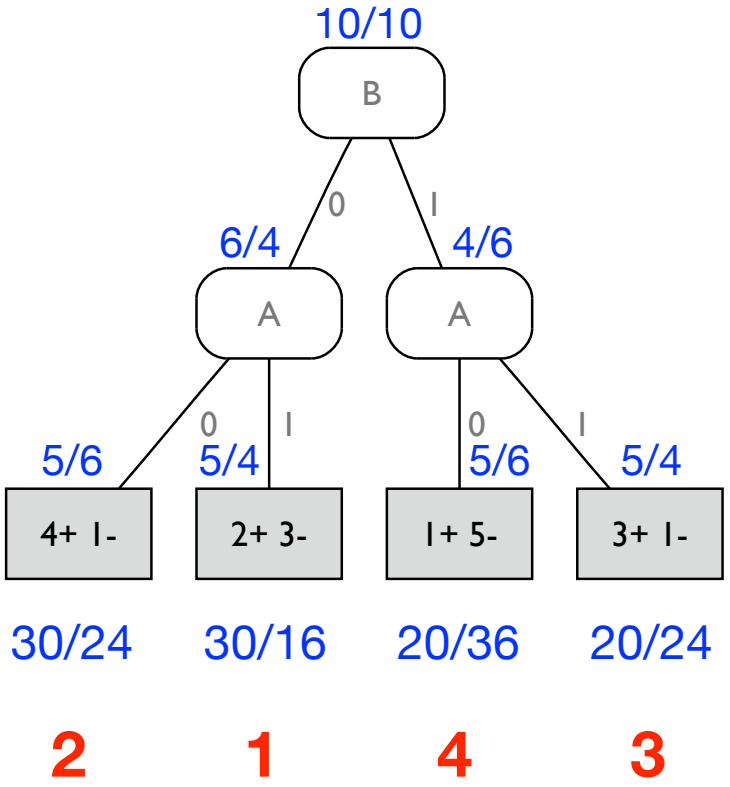






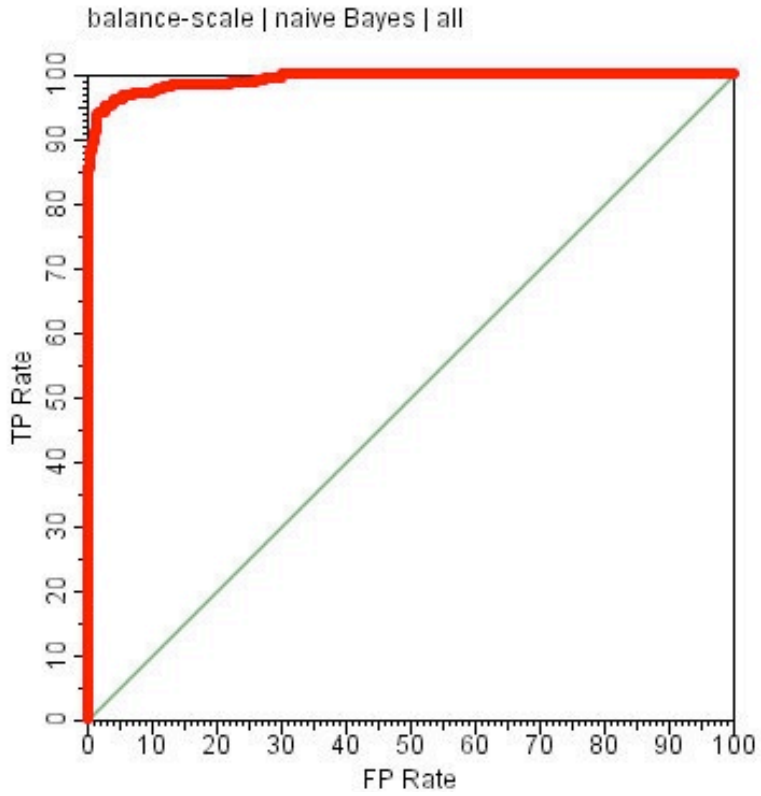
Naive Bayes ROC curve





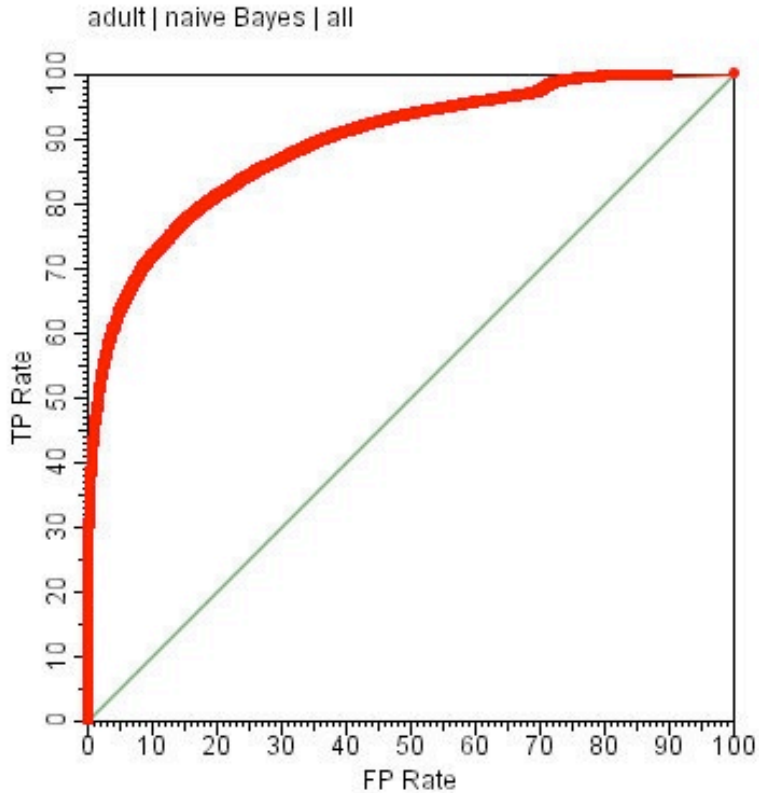
The concavity is caused by misleading marginal probabilities (*cf*. A=1, B=0). Repairing this would require access to the true joint probabilities.

Some example ROC curves



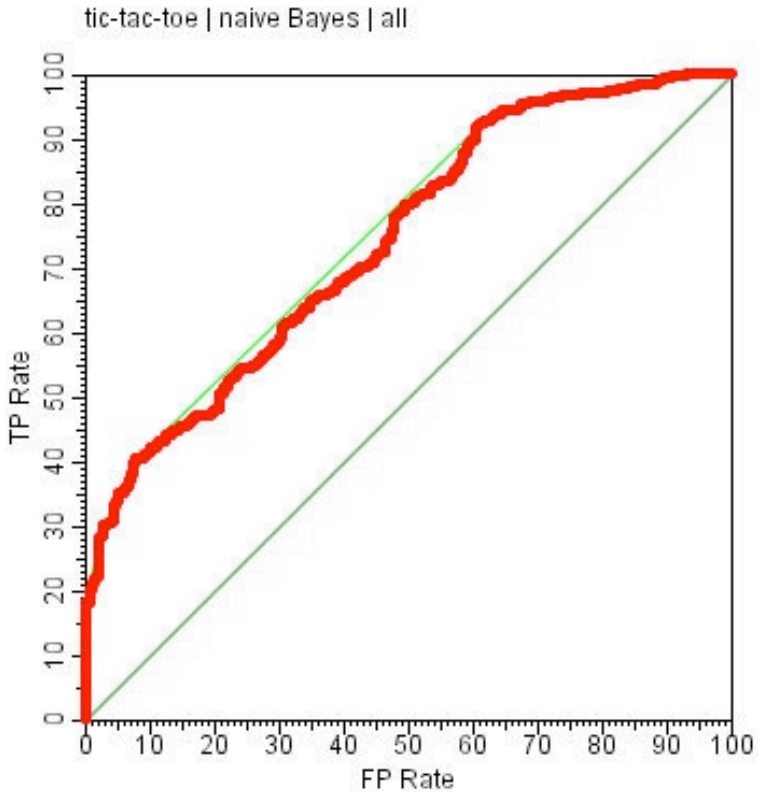
Good separation between classes, convex curve

Some example ROC curves



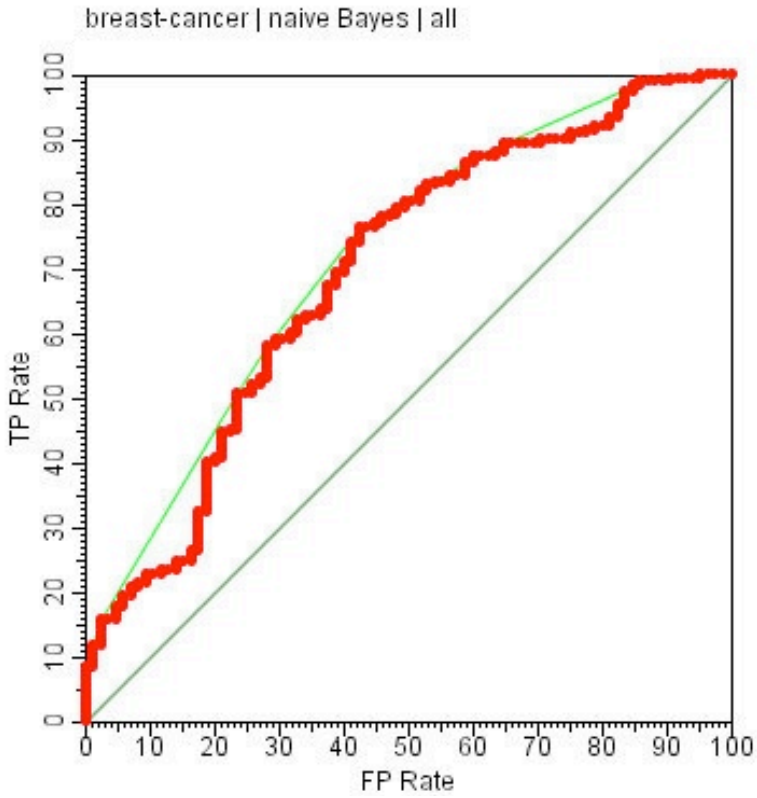
Reasonable separation, mostly convex

Some example ROC curves



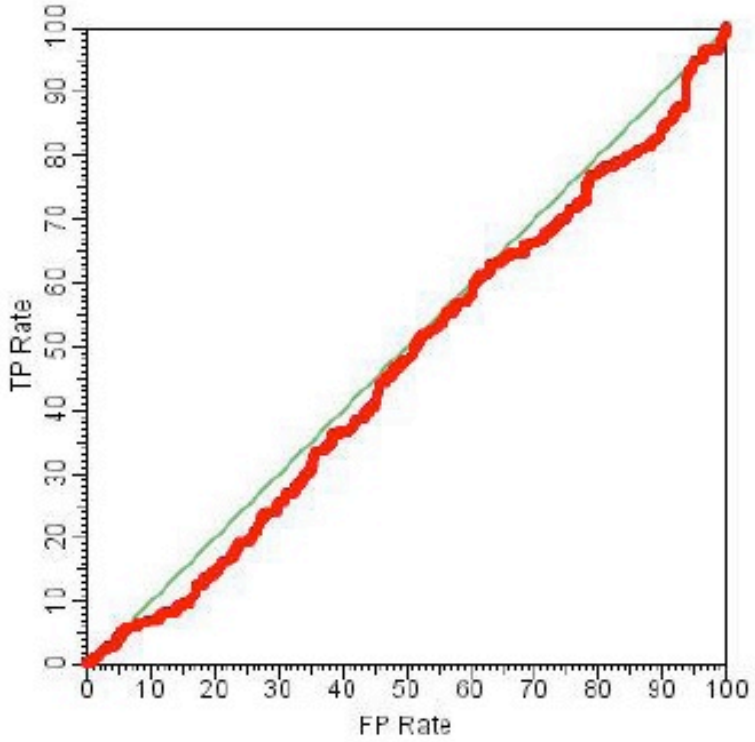
Fairly poor separation, mostly convex

Some example ROC curves



Poor separation, large and small concavities

Some example ROC curves



Random performance

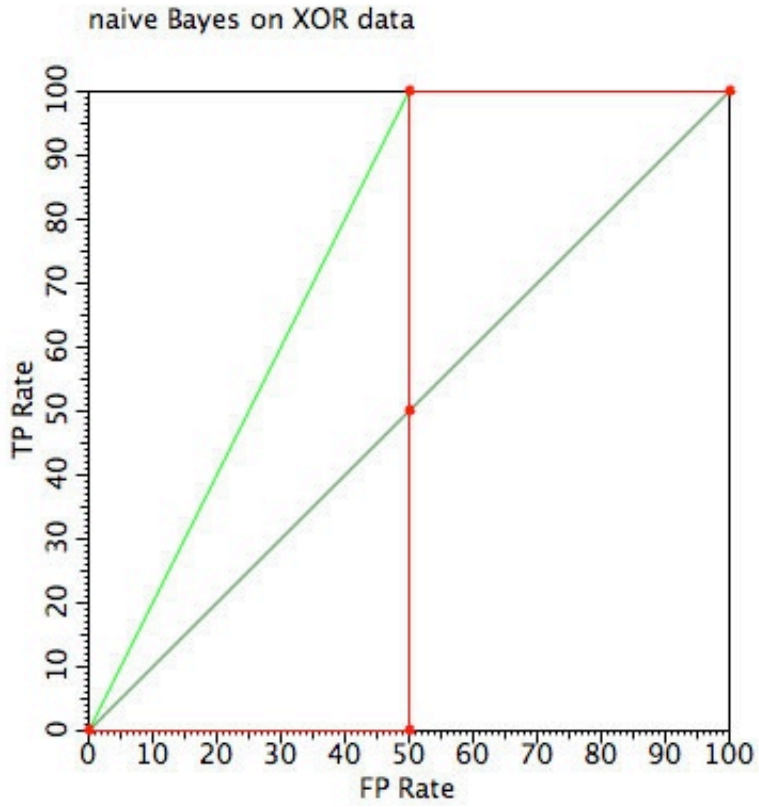
A ROC curve tell a story

- 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 class distribution in that segment of the ranking
  - straight segment -> tied ranking or locally random behaviour
- Concavities indicate locally worse than random behaviour
  - convex hull corresponds to discretising scores
  - can potentially do better: repairing concavities

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 test
  - estimates probability that randomly chosen +ve is ranked before randomly chosen –ve
  - $\frac{S_- - Pos(Pos-1)}{Pos \cdot Neg}$ where S_- is the sum of ranks of –ves
- Gini coefficient = 2*AUC–1 (area between curve and diagonal)
  - NB. not the same as Gini index!

AUC=0.5 not always random



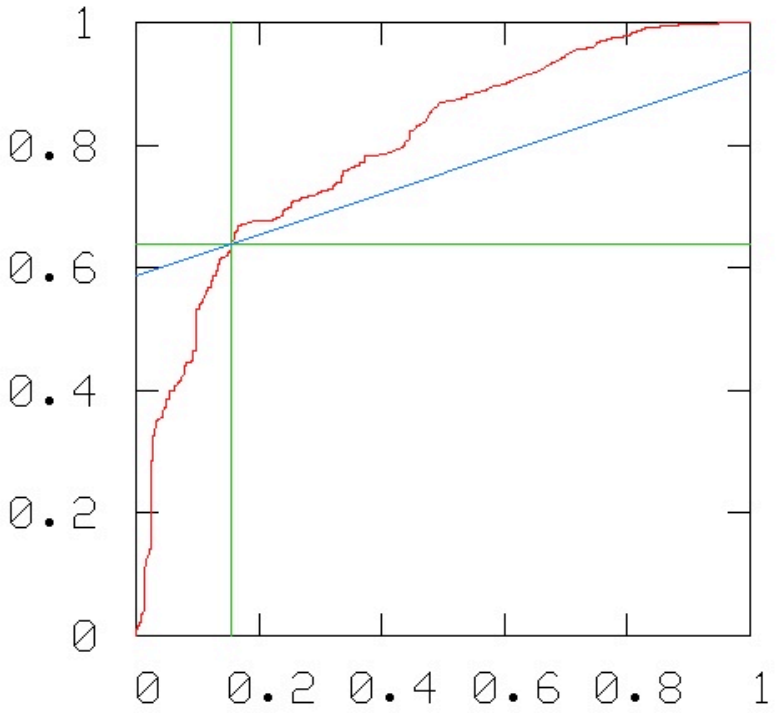
Poor performance because data requires two classification boundaries

Turning rankers into classifiers

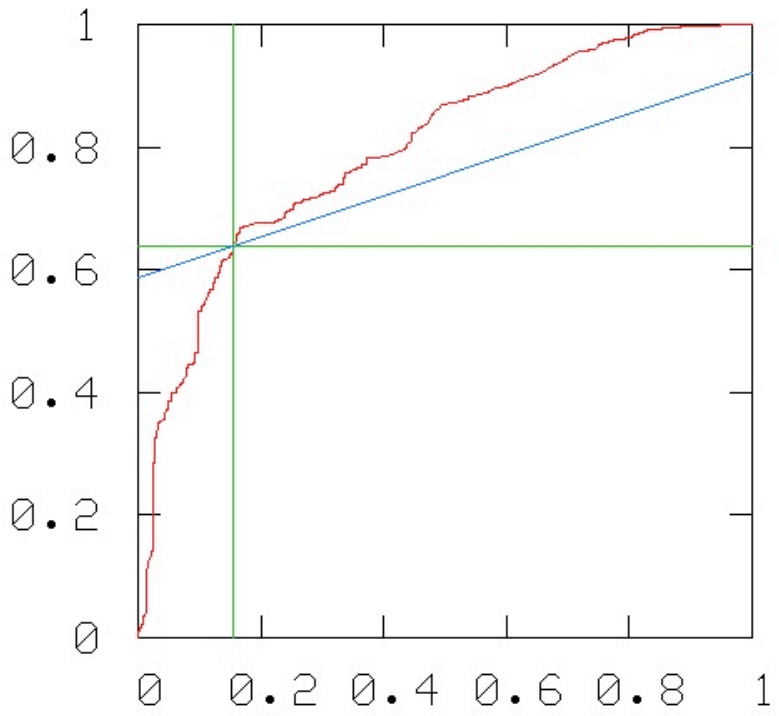
- Requires decision rule, i.e. setting a threshold on the scores f(x)
  - e.g. Bayesian: predict positive if $\frac{P(\oplus|x)}{P(\ominus|x)} = \frac{P(x|\oplus)}{P(x|\ominus)}\frac{P(\oplus)}{P(\ominus)} > 1$
  - equivalently: $\frac{P(x|\oplus)}{P(x|\ominus)} > \frac{P(\ominus)}{P(\oplus)}$

- If scores are calibrated we can use the Bayesian threshold
  - with uncalibrated scores we need to learn the threshold from the data
  - NB. naïve Bayes is uncalibrated
    - i.e. don't use prior, work directly with likelihood ratio

Uncalibrated threshold

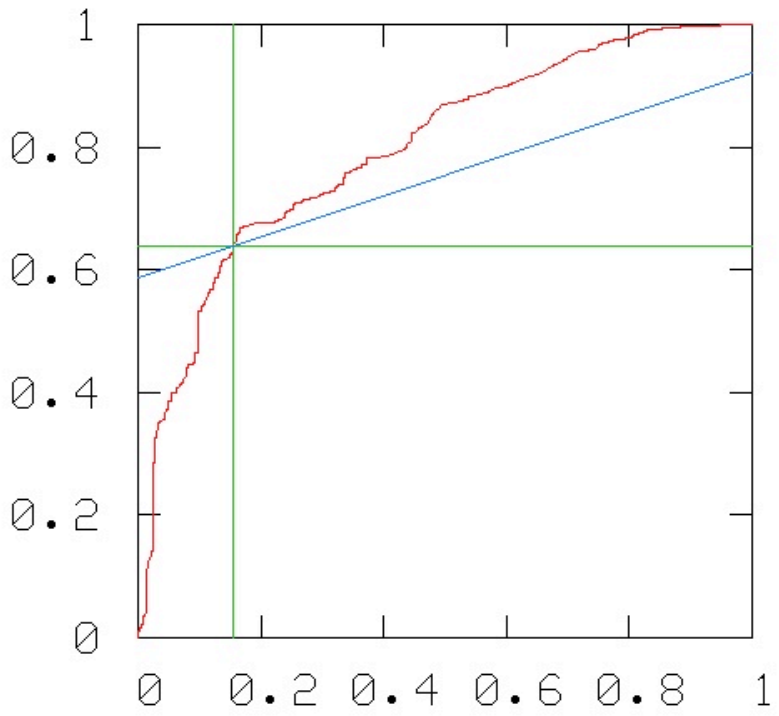


Uncalibrated threshold



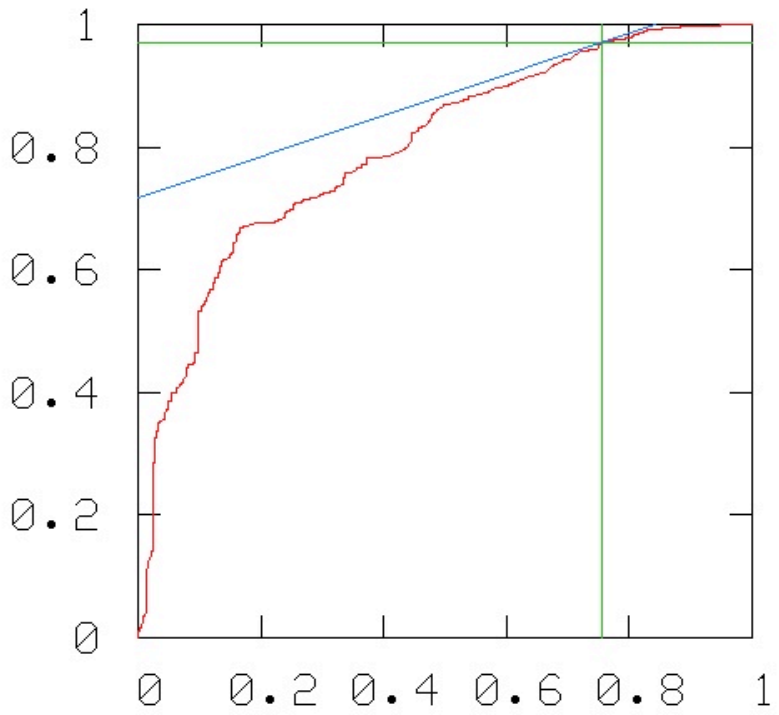
True and false positive rates achieved by default threshold

Uncalibrated threshold



True and false positive rates achieved by default threshold
(NB. worse than majority class!)

Calibrated threshold



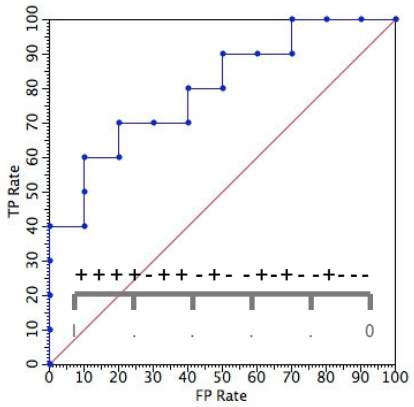
Optimal achievable accuracy

Calibration

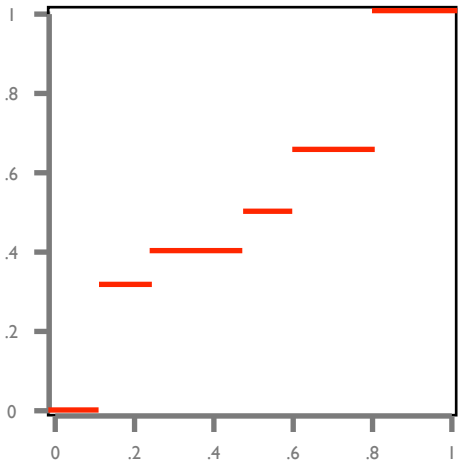
- Well-calibrated class probabilities have the following property:
  - conditioning a test sample on predicted probability *p*, the expected proportion of positives is close to *p*
- Thus, the predicted likelihood ratio approximates the slope of the ROC curve
  - perfect calibration implies convex ROC 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, ICML'01; Fawcett & Niculescu-Mizil, MLj'07; Flach & Matsubara, ECML'07)
    - notice that this is exactly what decision trees do, so they are well-calibrated on the training set

Isotonic calibration = pool adjacent violators

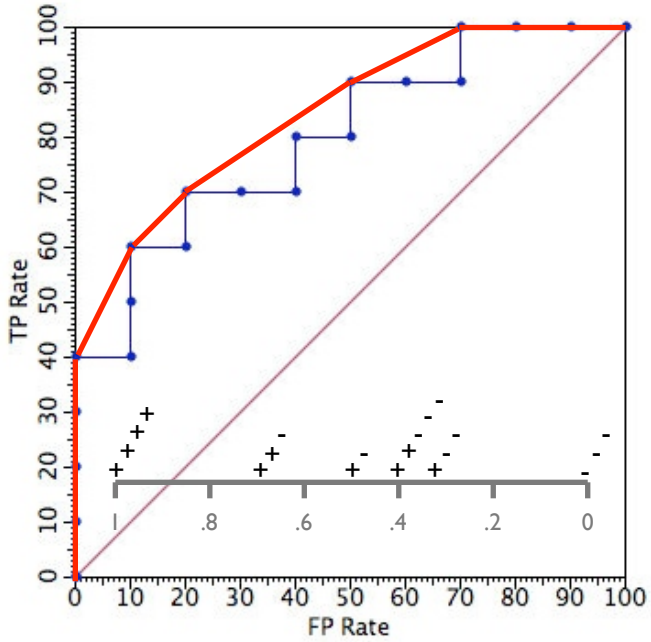
Original scores



Calibration map



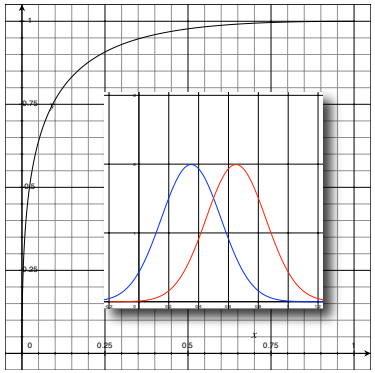
Calibrated scores



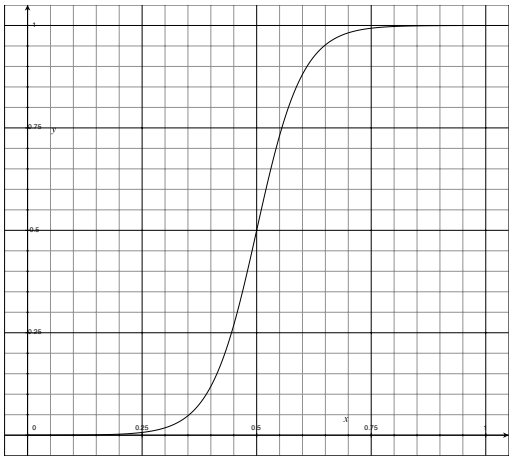
Piecewise constant calibration map leads to more ties in the ranking.

Parametric alternative: logistic calibration

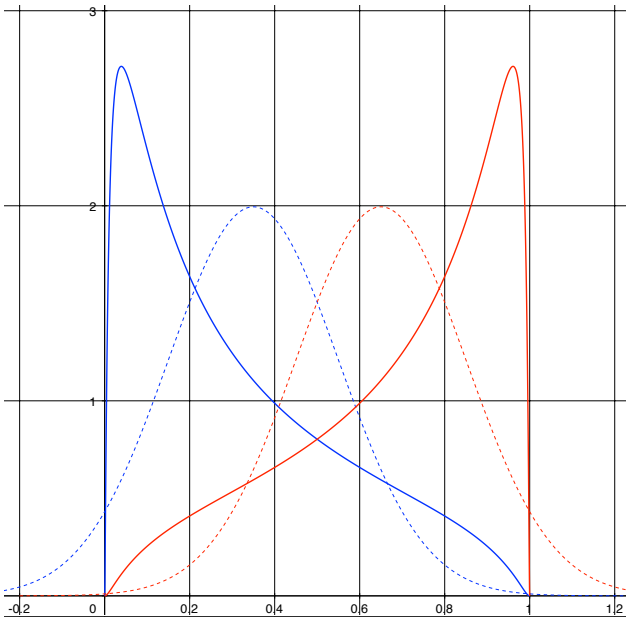
Normally distributed scores



Calibration map: logistic function

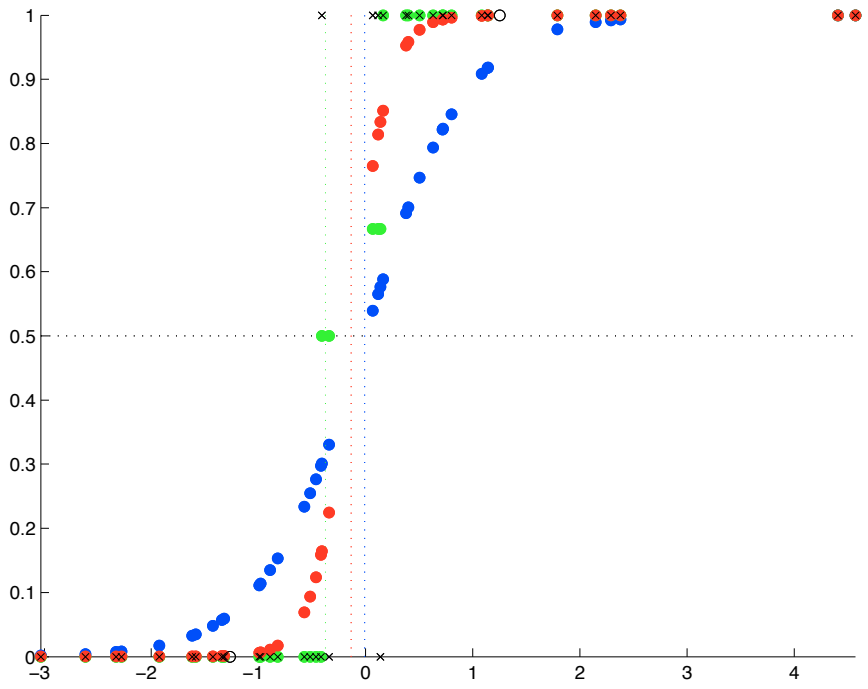


Score distributions after calibration



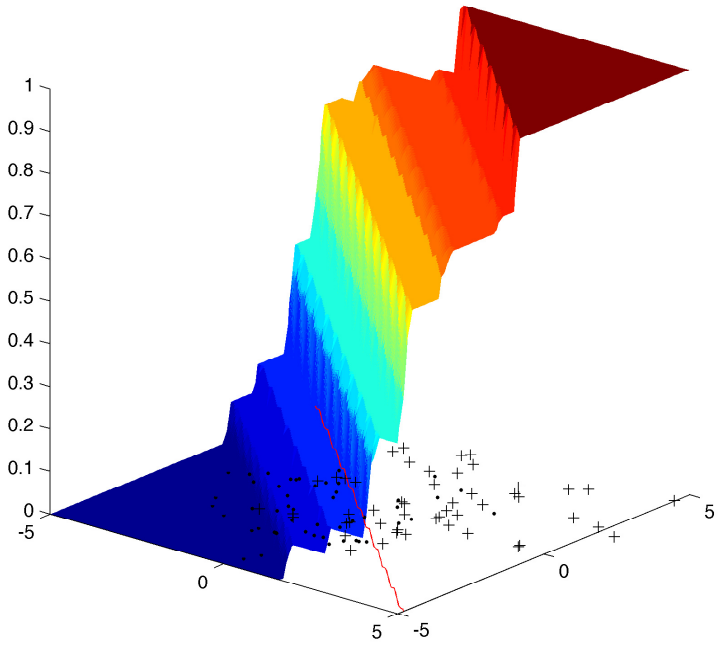
Logistic regression optimises this directly.

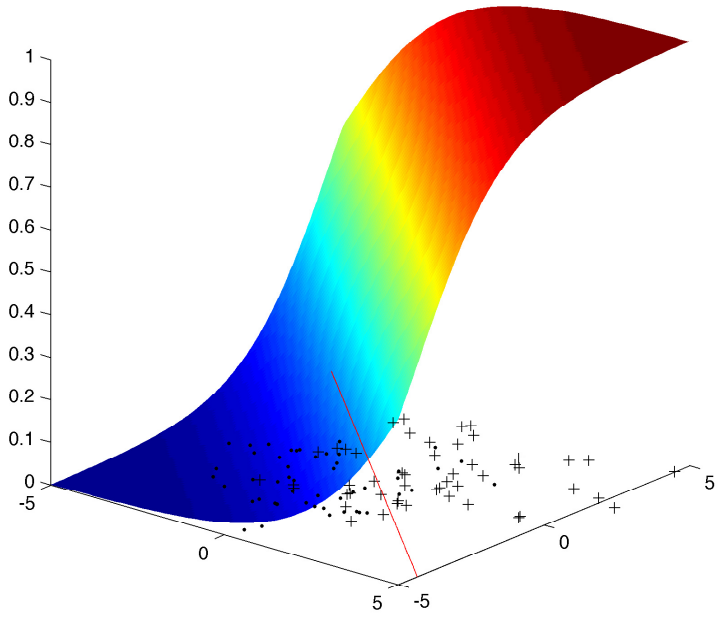
1-D example



Blue: logistically calibrated mean-of-means
Green: isotonically calibrated mean-of-means
Red: logistic regression

2-D example



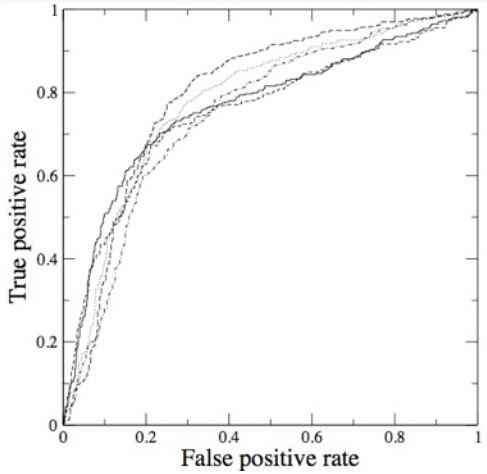


Left: isotonically calibrated difference-between-means classifier
Right: logistically calibrated difference-between-means classifier

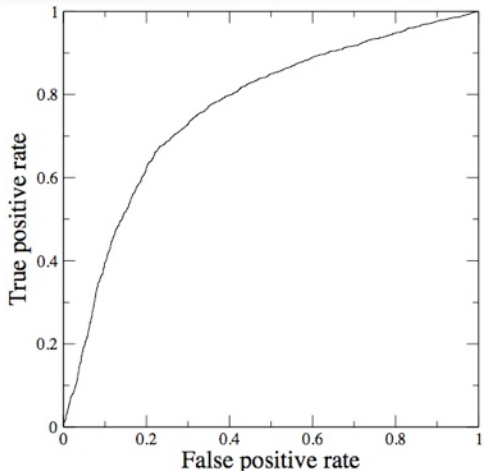
Averaging ROC curves

- To obtain a cross-validated ROC curve
  - just combine all test folds with scores for each instance, and draw a single ROC curve
- To obtain cross-validated AUC estimate with error bounds
  - calculate AUC in each test fold and average
  - or calculate AUC from single cv-ed curve and use bootstrap resampling for error bounds
- To obtain ROC curve with error bars
  - vertical averaging (sample at fixed fpr points)
  - threshold averaging (sample at fixed thresholds)
  - see (Fawcett, 2004)

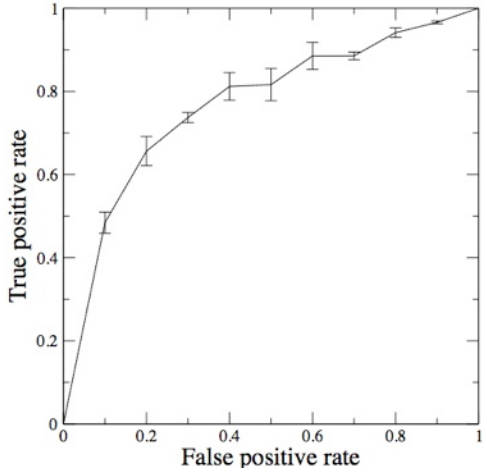
Averaging ROC curves



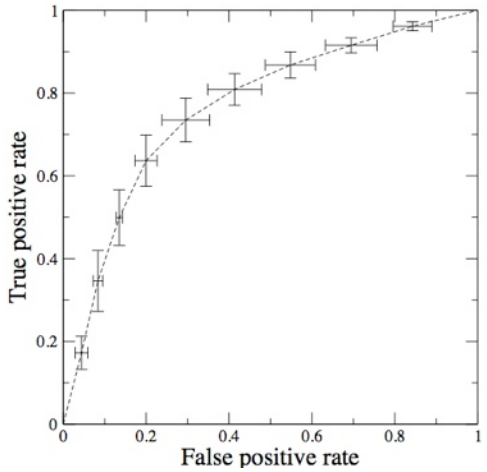
(a) ROC curves from five test samples



(b) ROC curve from combining the samples



(c) Vertical averaging, fixing fpr

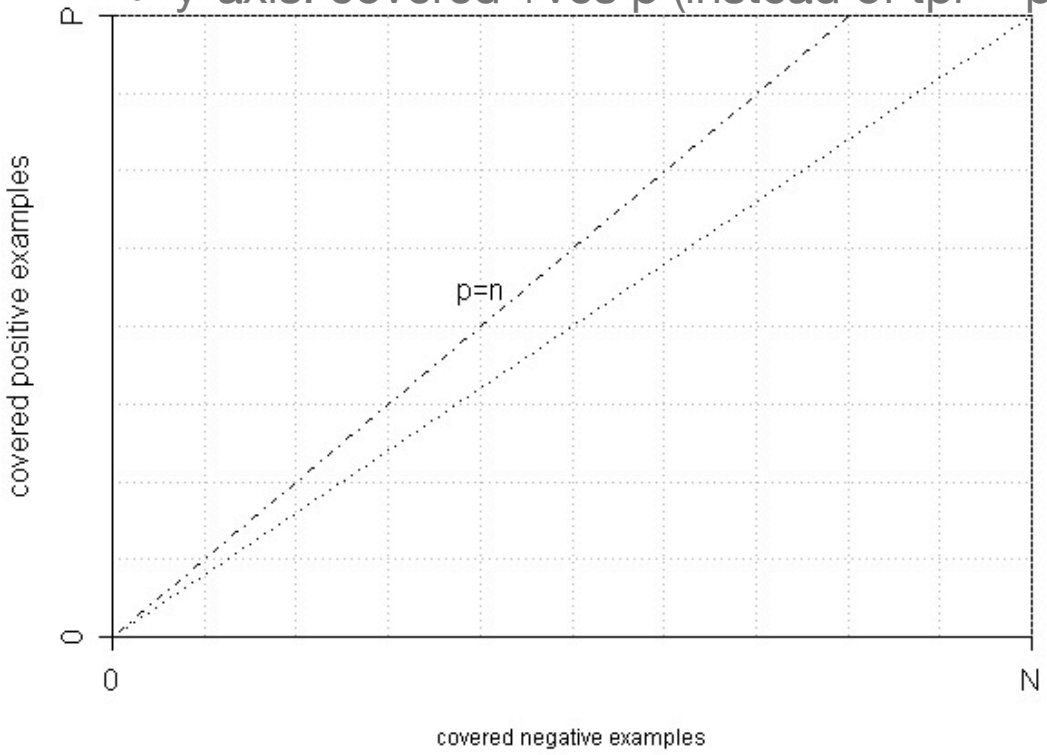


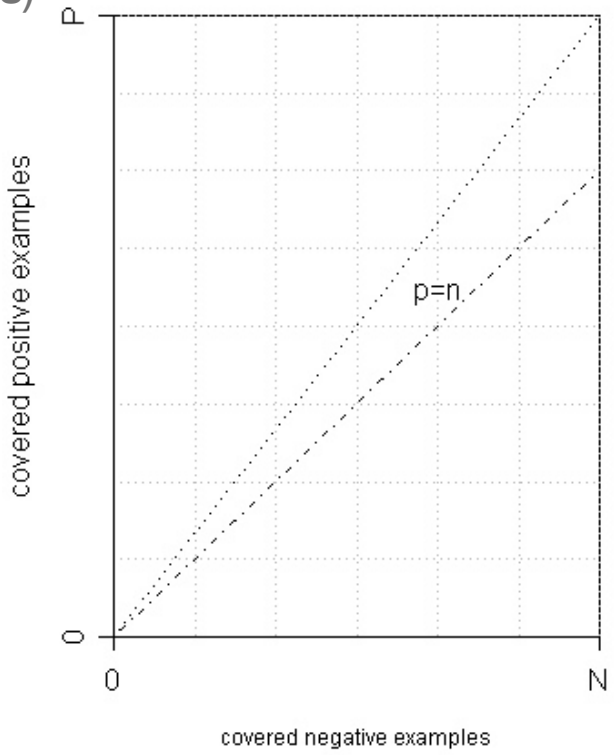
(d) Threshold averaging

From (Fawcett, 2004)

PN spaces

- PN spaces are ROC spaces with non-normalised axes
  - x-axis: covered –ves n (instead of fpr = n/Neg)
  - y-axis: covered +ves p (instead of tpr = p/Pos)

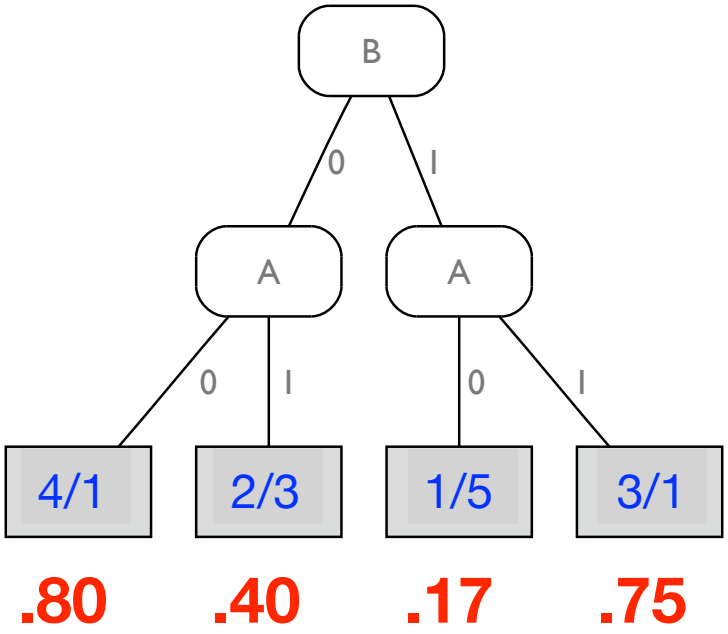




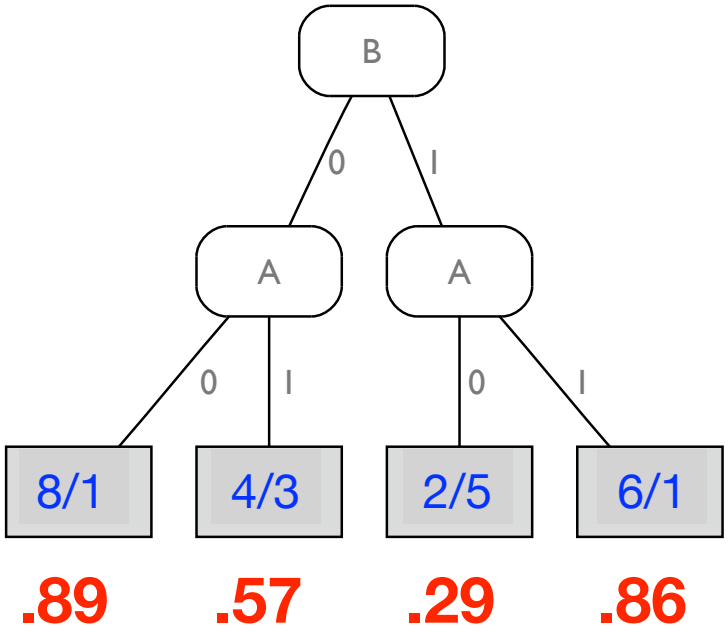
Posterior odds or likelihood ratio?

- In PN plots slopes are posterior odds and the aspect ratio is the prior odds.
  - useful for visualising performance on single data set
- In ROC plots slopes are likelihood ratios; the prior odds is not visible unless you draw accuracy isometrics.
  - useful if class distribution is not fixed
- One way of obtaining likelihood ratios is by rebalancing the classes:
  - posterior odds $po = lr * \pi/(1-\pi)$
  - likelihood ratio $lr = po * (1-\pi)/\pi$

Posterior odds or likelihood ratio (2)



(Re)balanced classes



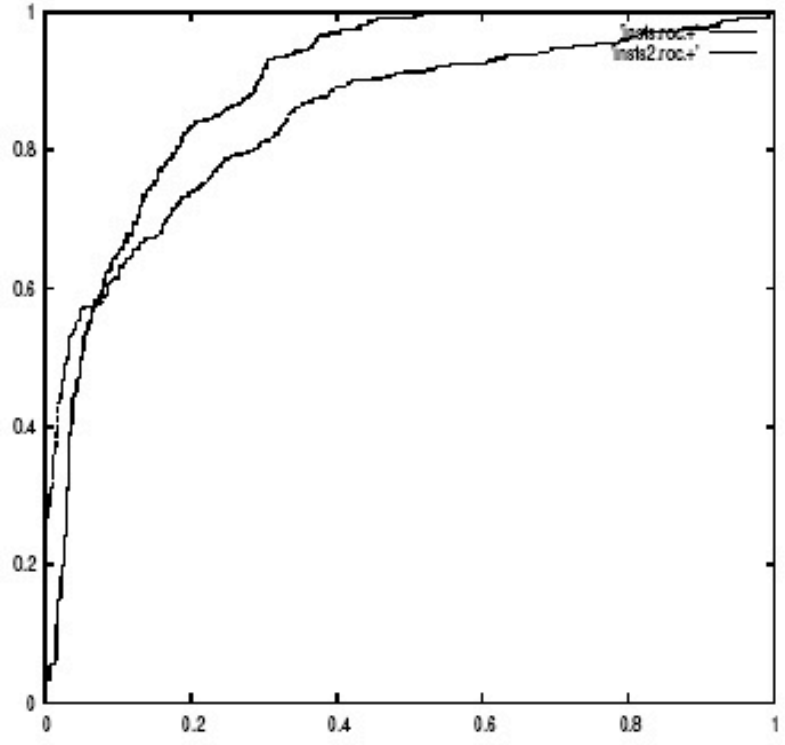
Twice as many +ves as –ves

Precision-recall curves

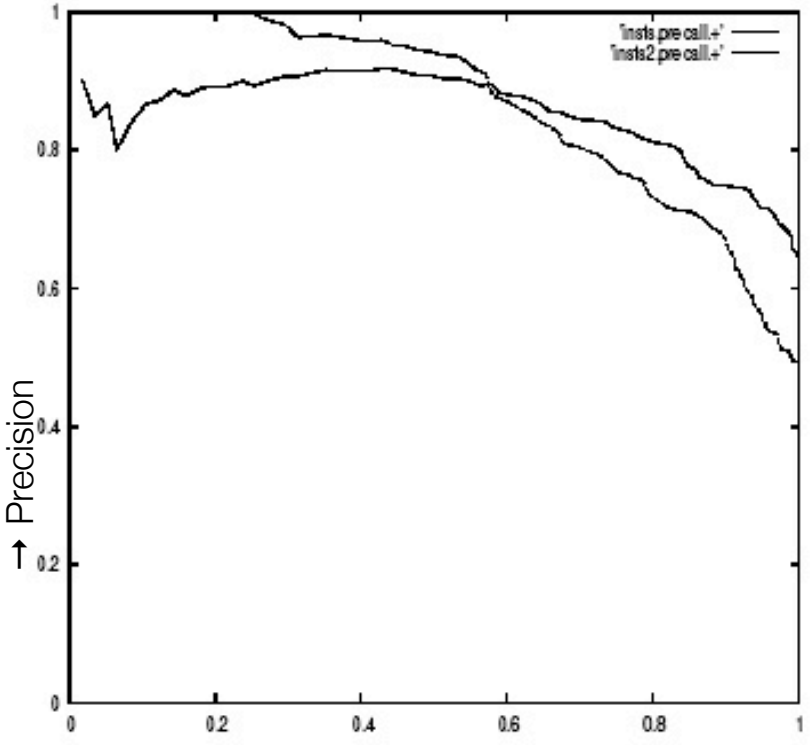
- Precision prec = TP/PPos = TP/TP+FP
  - fraction of positive predictions correct
- Recall rec = tpr = TP/Pos = TP/TP+FN
  - fraction of positives correctly predicted
- Note: neither depends on true negatives
  - makes sense in information retrieval, where true negatives tend to dominate —> low fpr easy
- F-measure is harmonic mean of precision and recall
  - Quiz question: why **harmonic** mean?

	Predicted ⊕	*Predicted* ⊖	
Actual ⊕	**TP**	**FN**	Pos
Actual ⊖	**FP**	**TN**	Neg
	PPos	PNeg	

PR curves vs. ROC curves



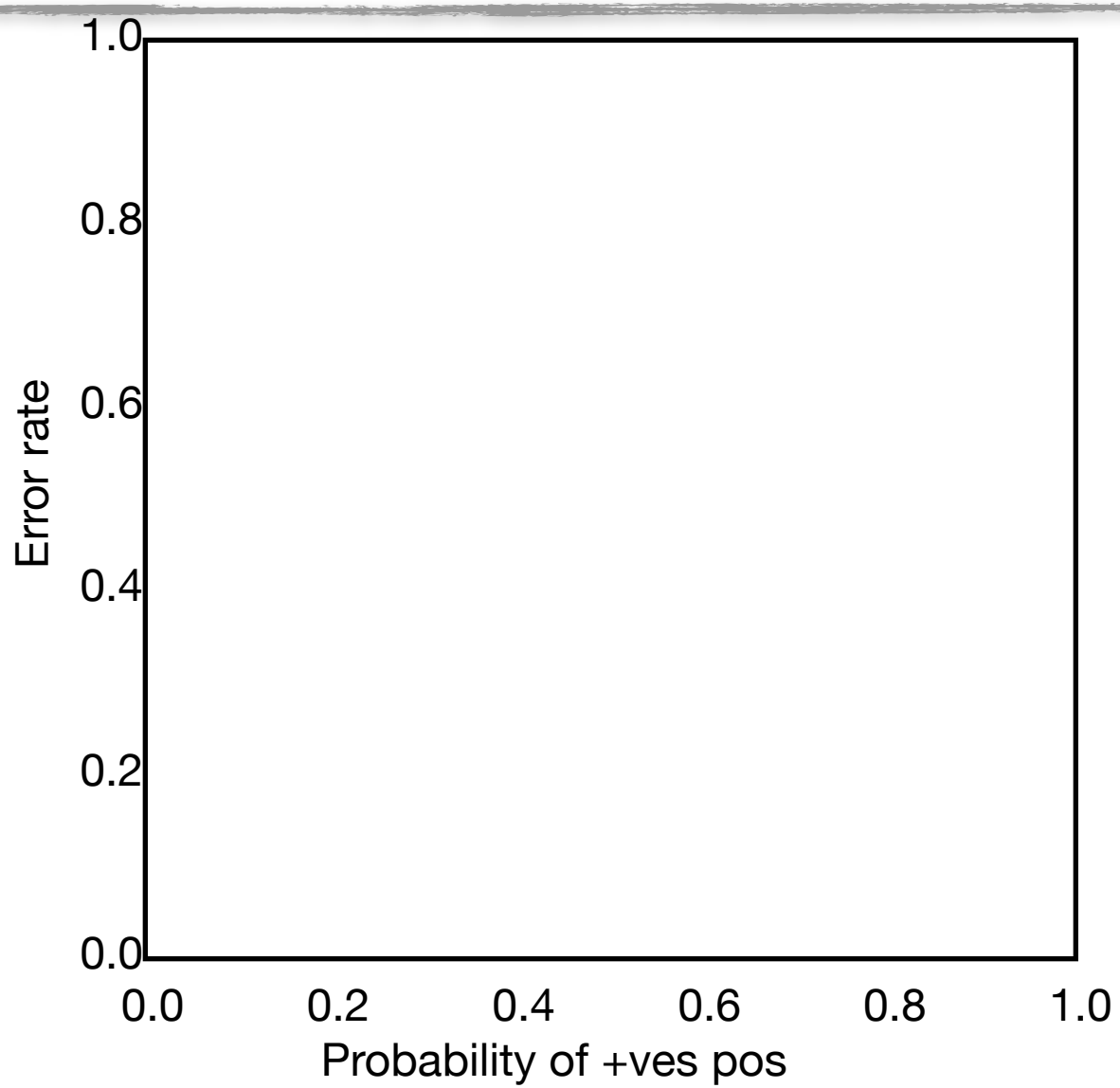
From (Fawcett, 2004)



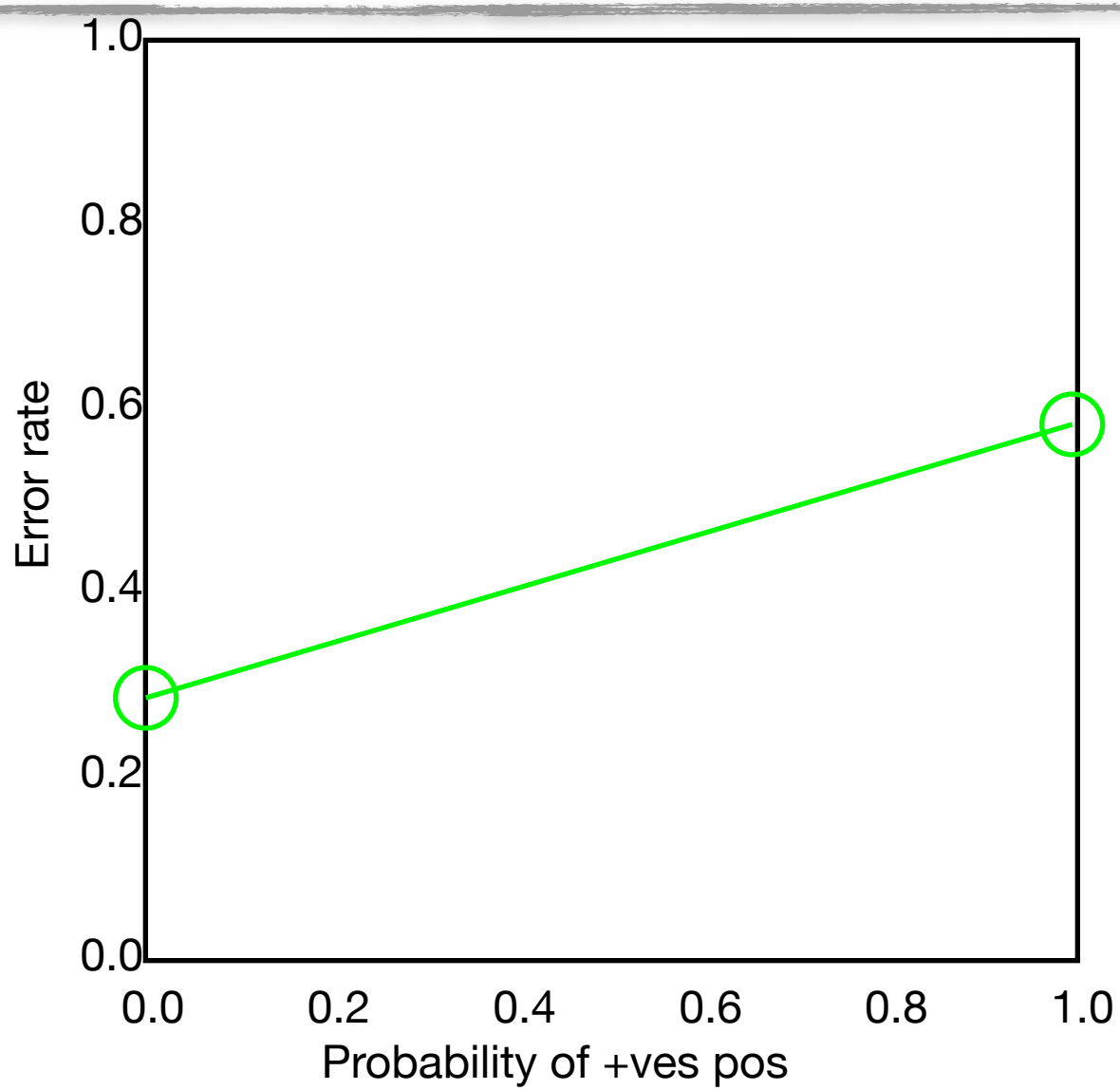
→ Recall

NB. Linear interpolation in ROC space → non-linear interpolation in PR space

Cost curves (Drummond & Holte, 2001)

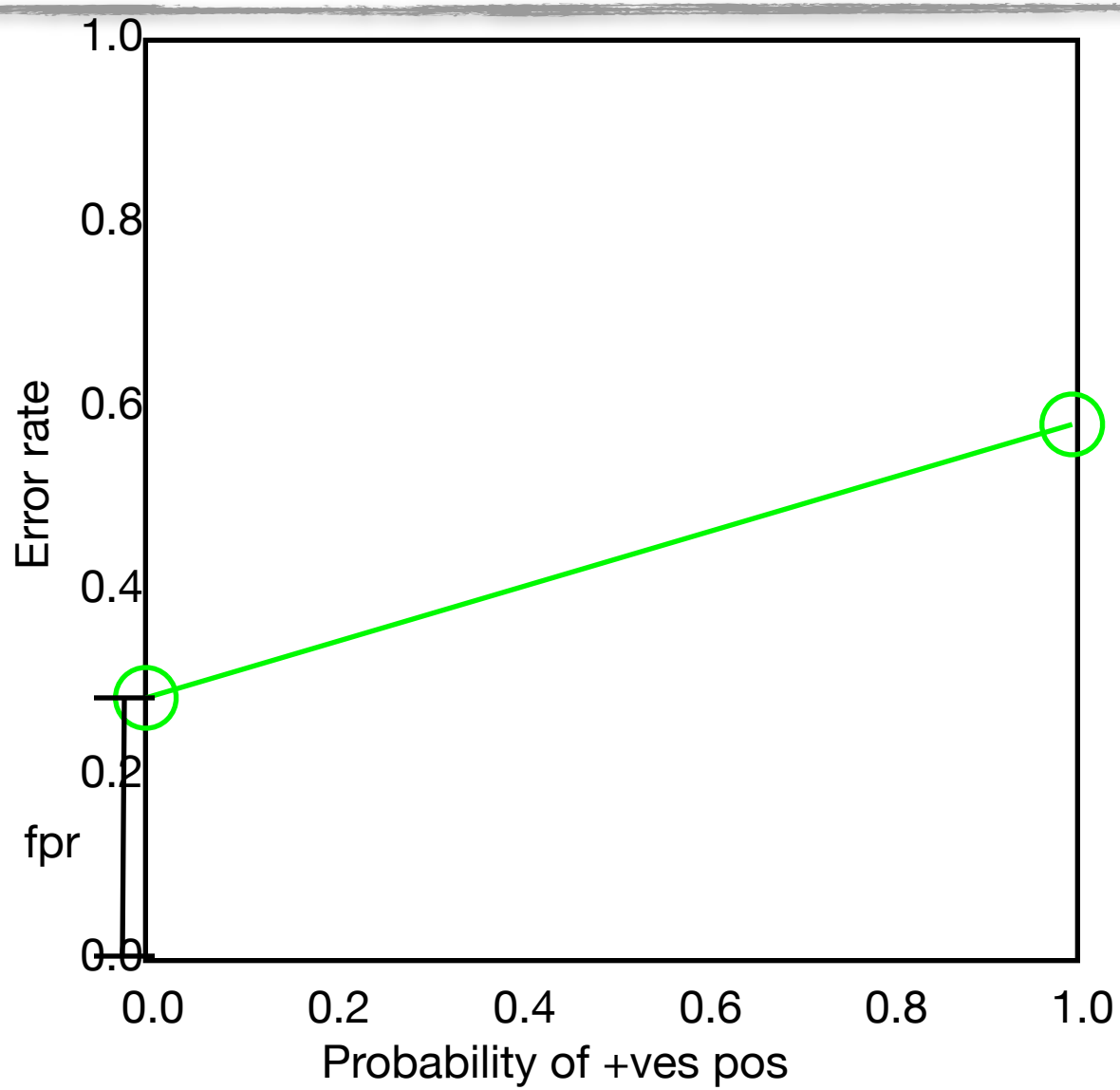


Cost curves (Drummond & Holte, 2001)



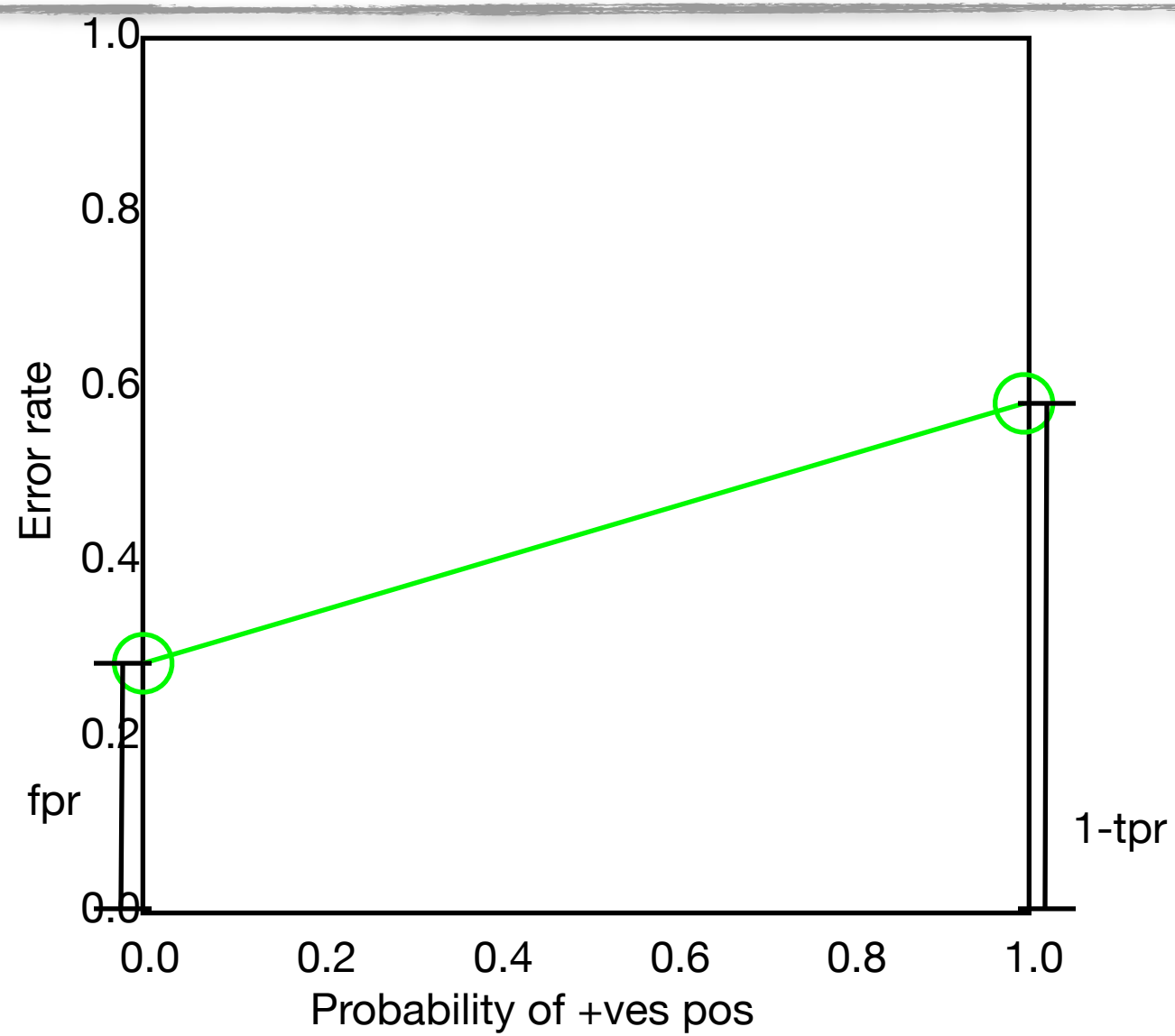
Classifier 1
tpr = 0.4
fpr = 0.3

Cost curves (Drummond & Holte, 2001)

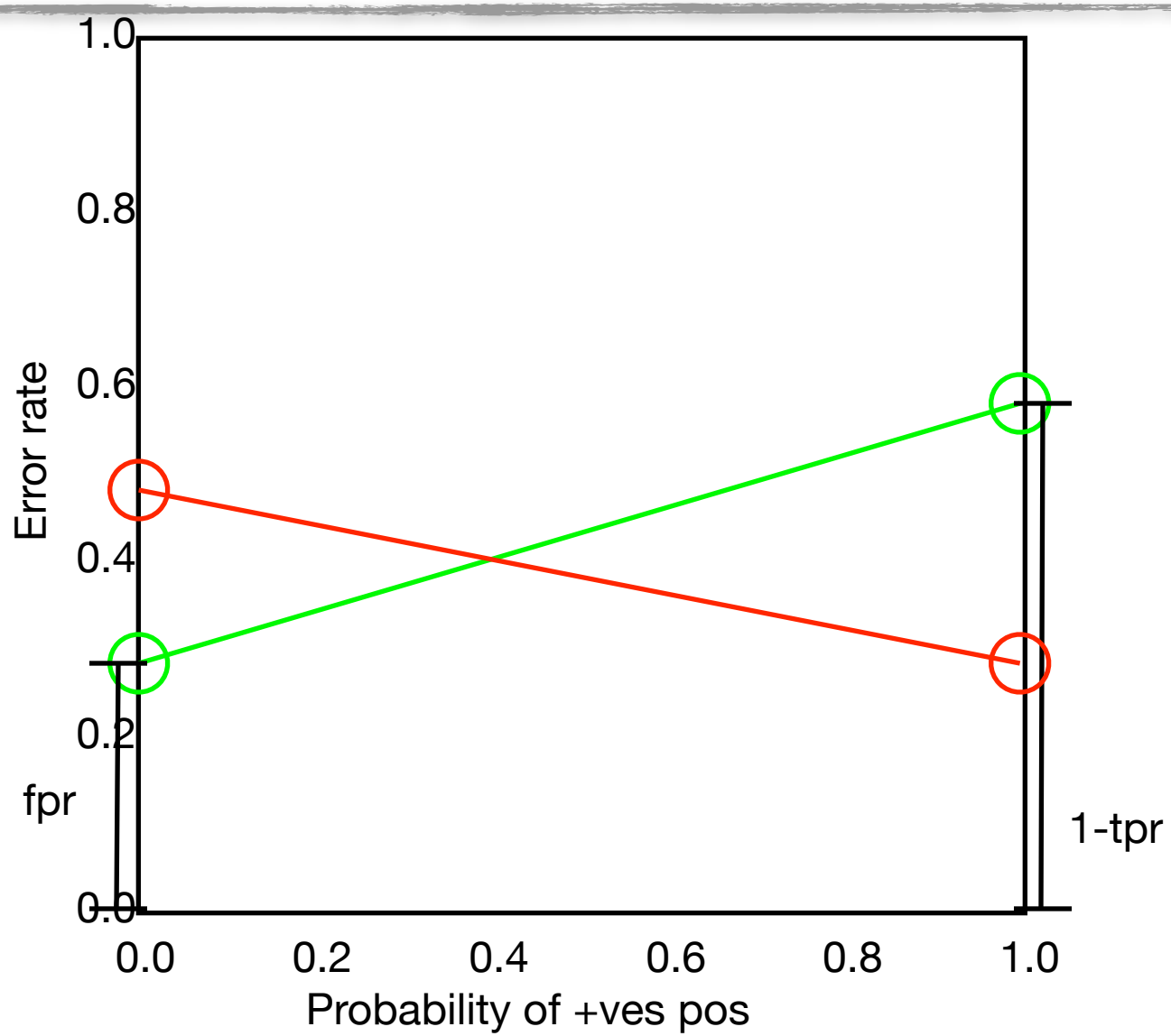


Classifier 1
tpr = 0.4
fpr = 0.3

Cost curves (Drummond & Holte, 2001)



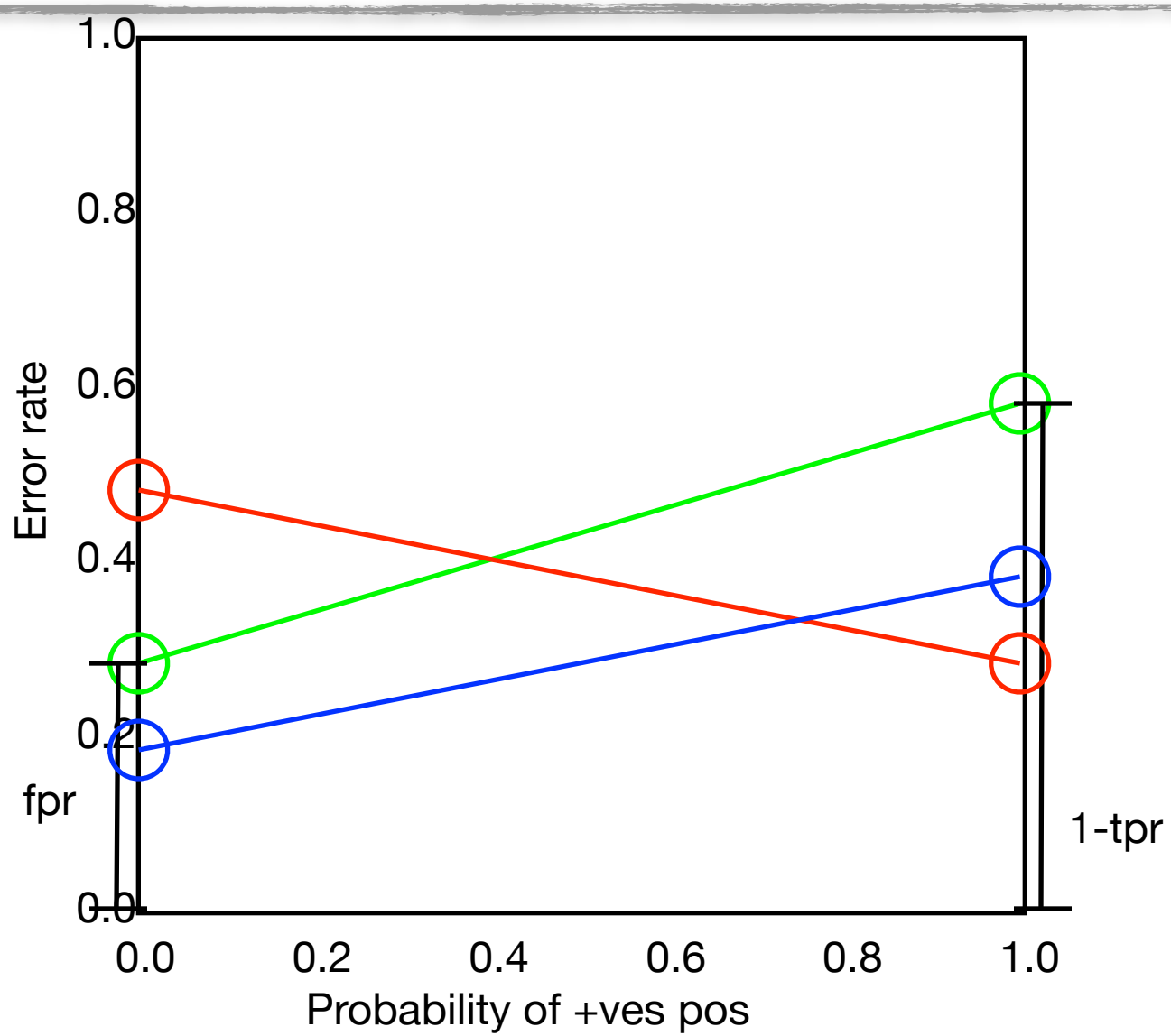
Cost curves (Drummond & Holte, 2001)



Classifier 1
tpr = 0.4
fpr = 0.3

Classifier 2
tpr = 0.7
fpr = 0.5

Cost curves (Drummond & Holte, 2001)

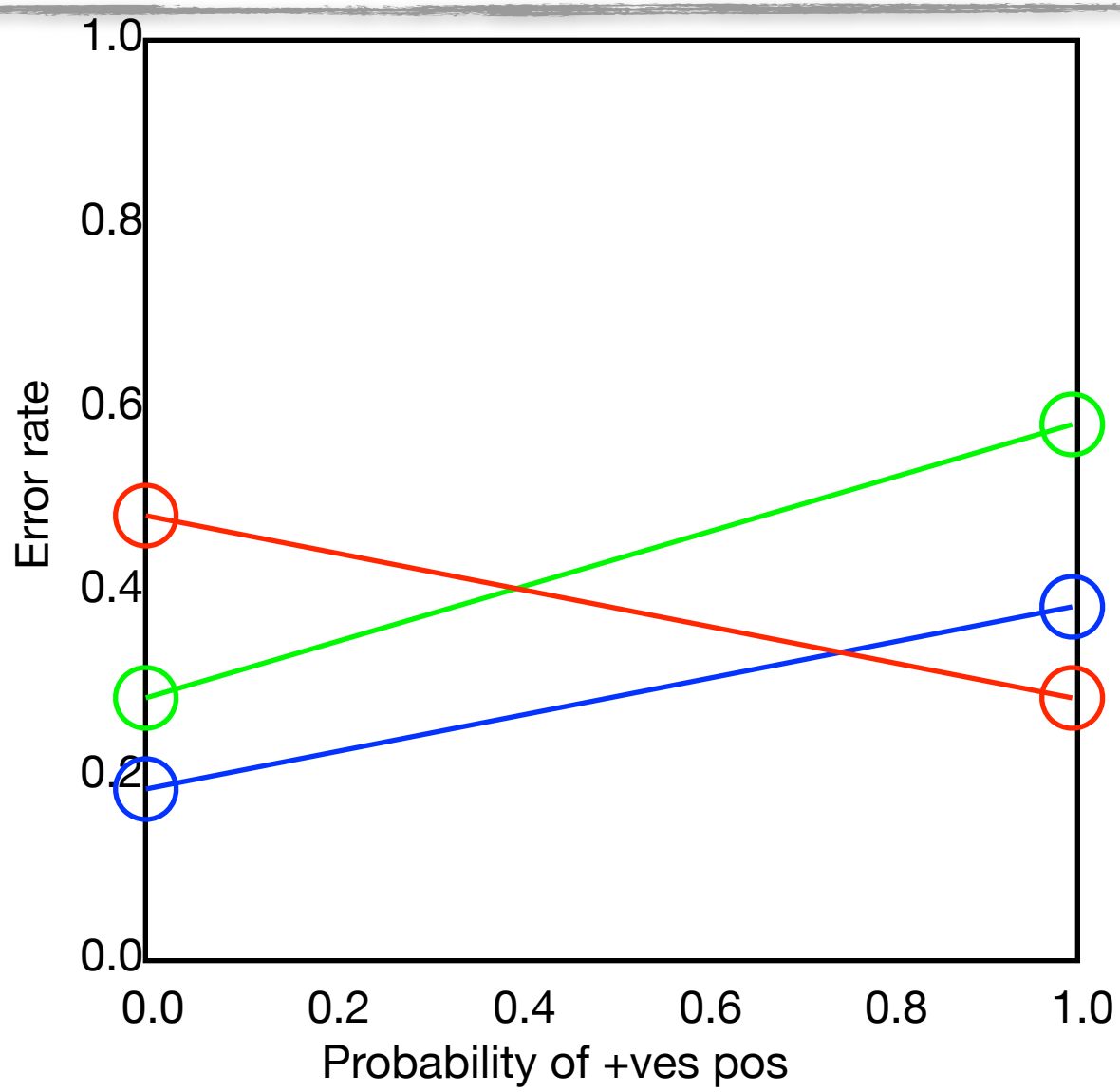


Classifier 1
tpr = 0.4
fpr = 0.3

Classifier 2
tpr = 0.7
fpr = 0.5

Classifier 3
tpr = 0.6
fpr = 0.2

Lower envelope

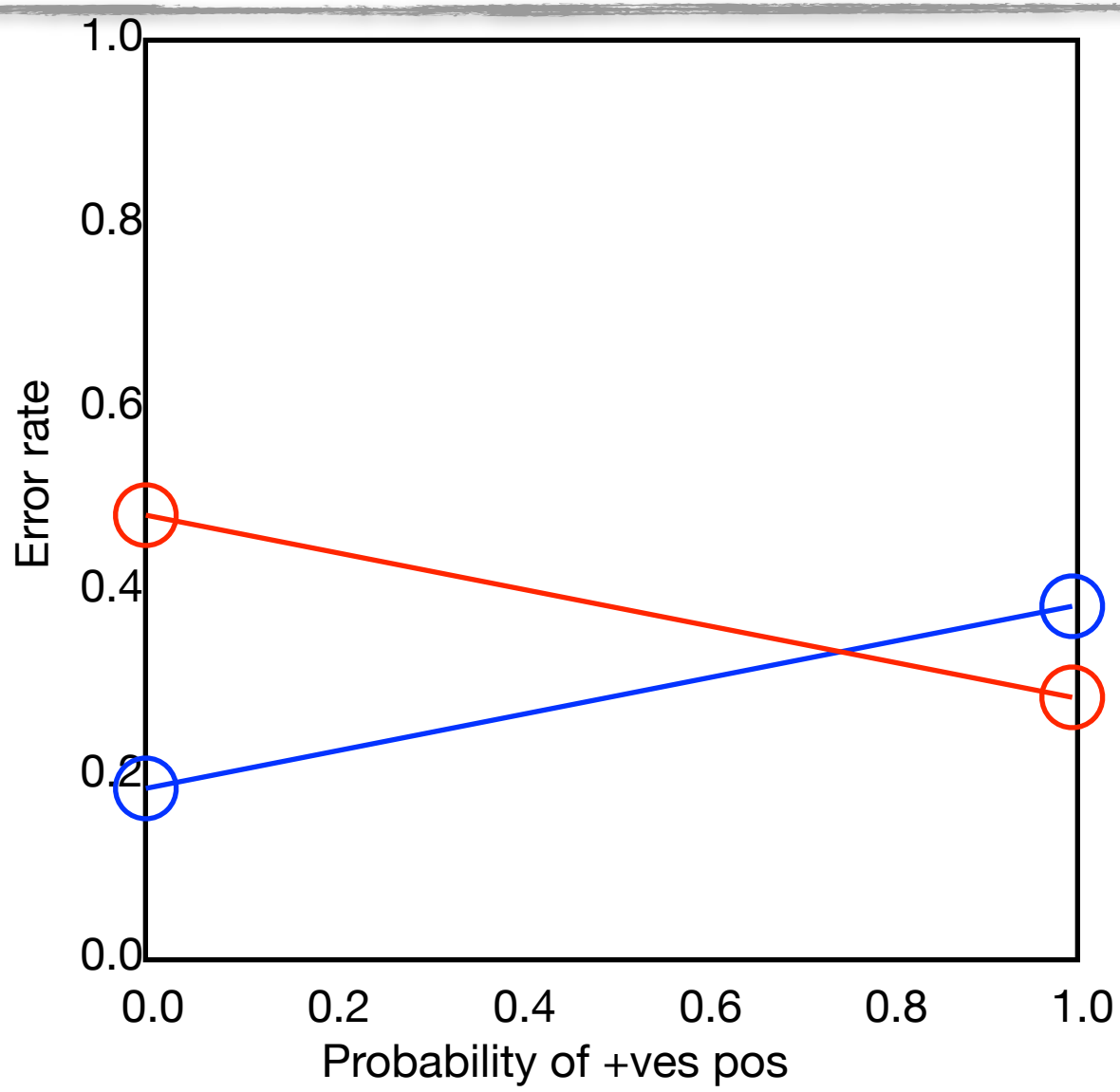


Classifier 1
tpr = 0.4
fpr = 0.3

Classifier 2
tpr = 0.7
fpr = 0.5

Classifier 3
tpr = 0.6
fpr = 0.2

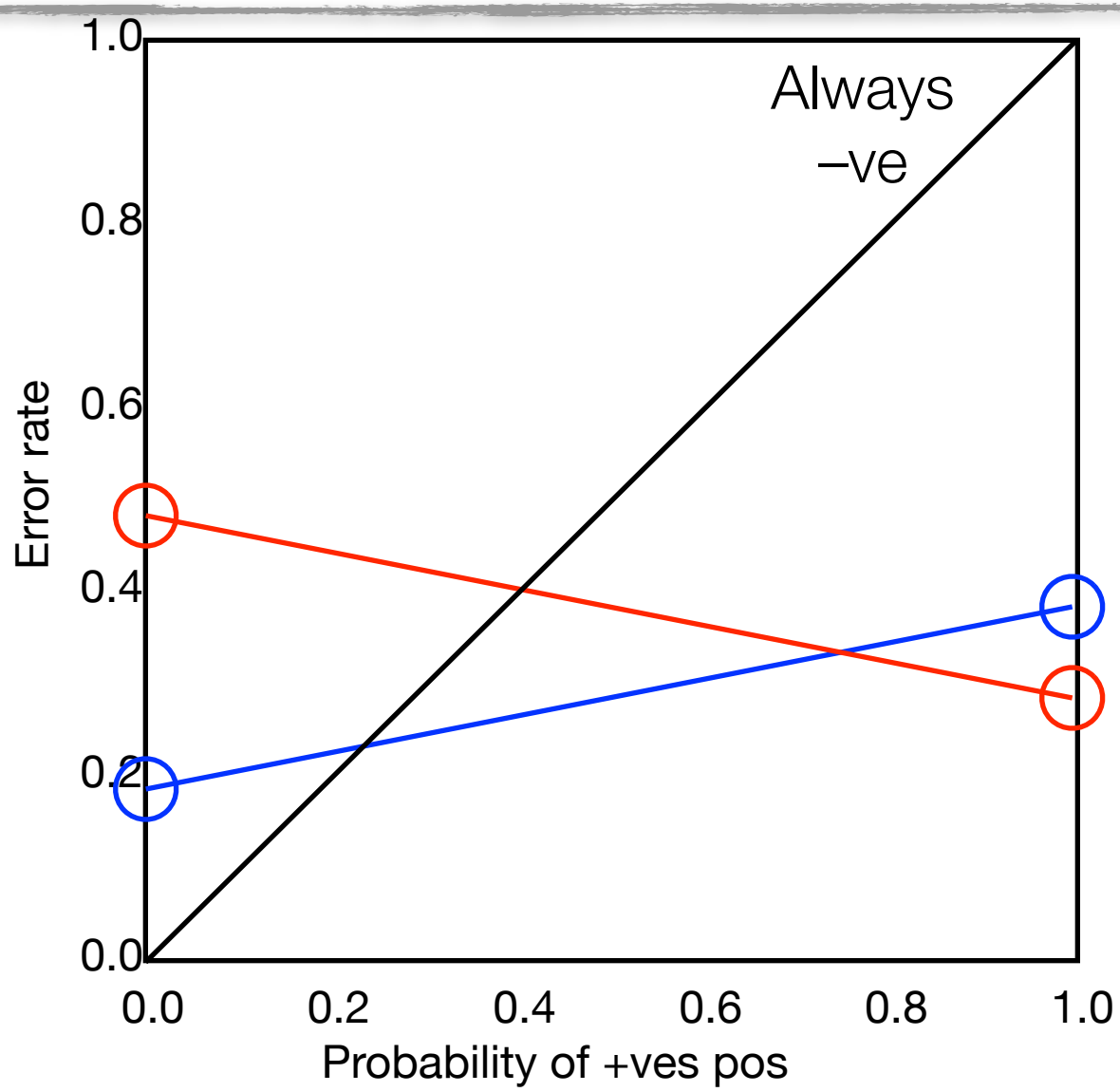
Lower envelope



Classifier 2
tpr = 0.7
fpr = 0.5

Classifier 3
tpr = 0.6
fpr = 0.2

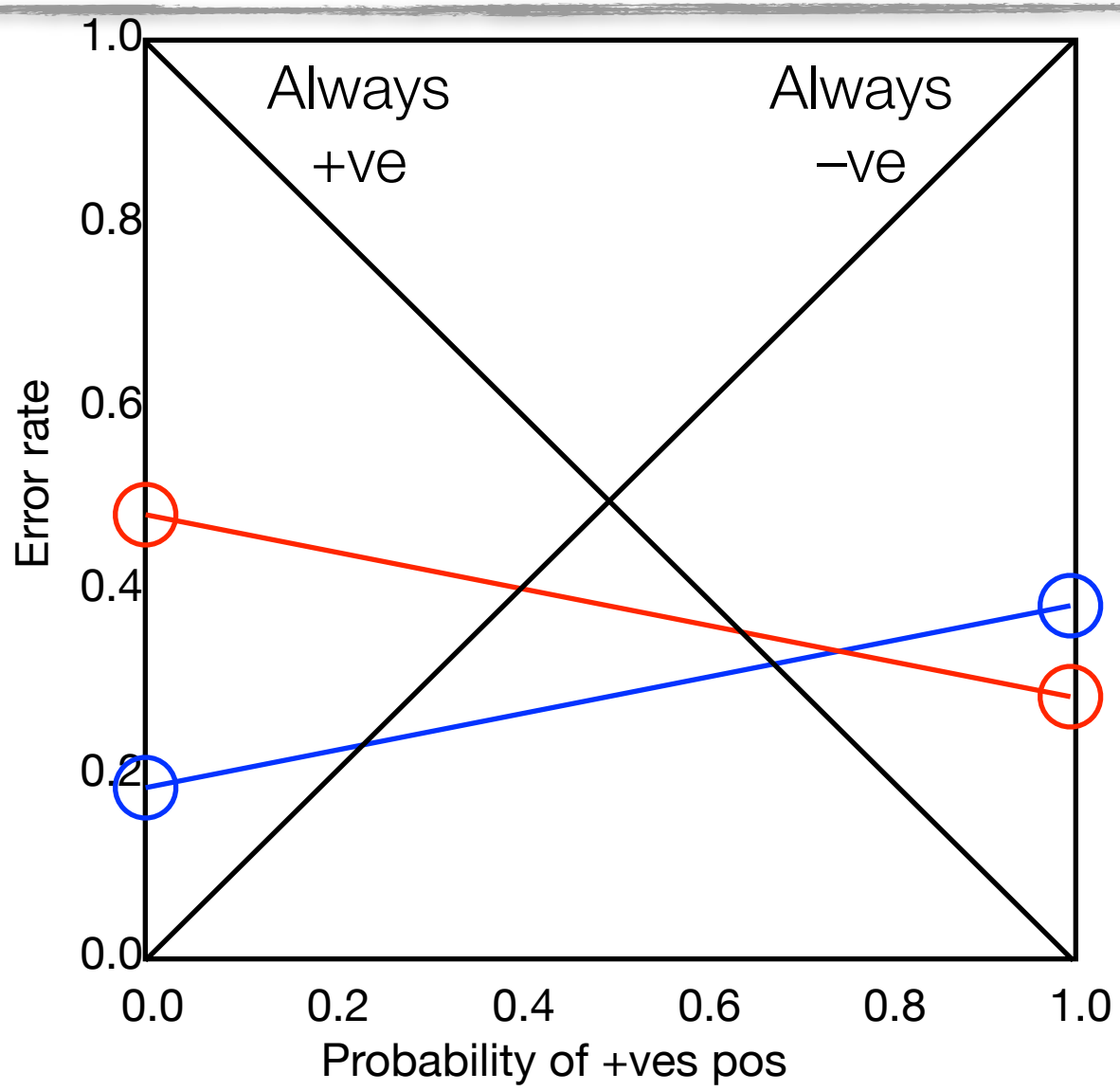
Lower envelope



Classifier 2
tpr = 0.7
fpr = 0.5

Classifier 3
tpr = 0.6
fpr = 0.2

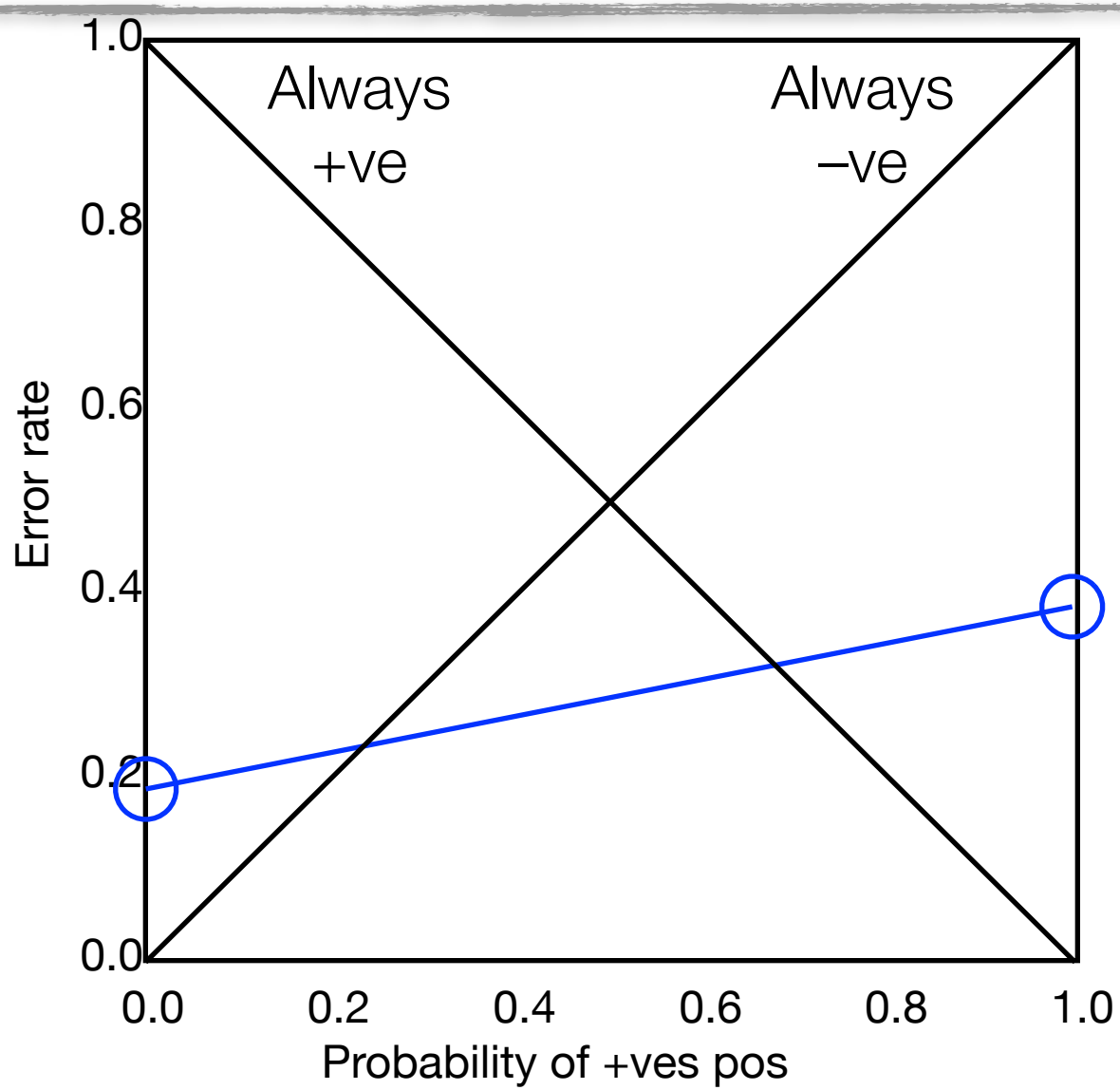
Lower envelope



Classifier 2
tpr = 0.7
fpr = 0.5

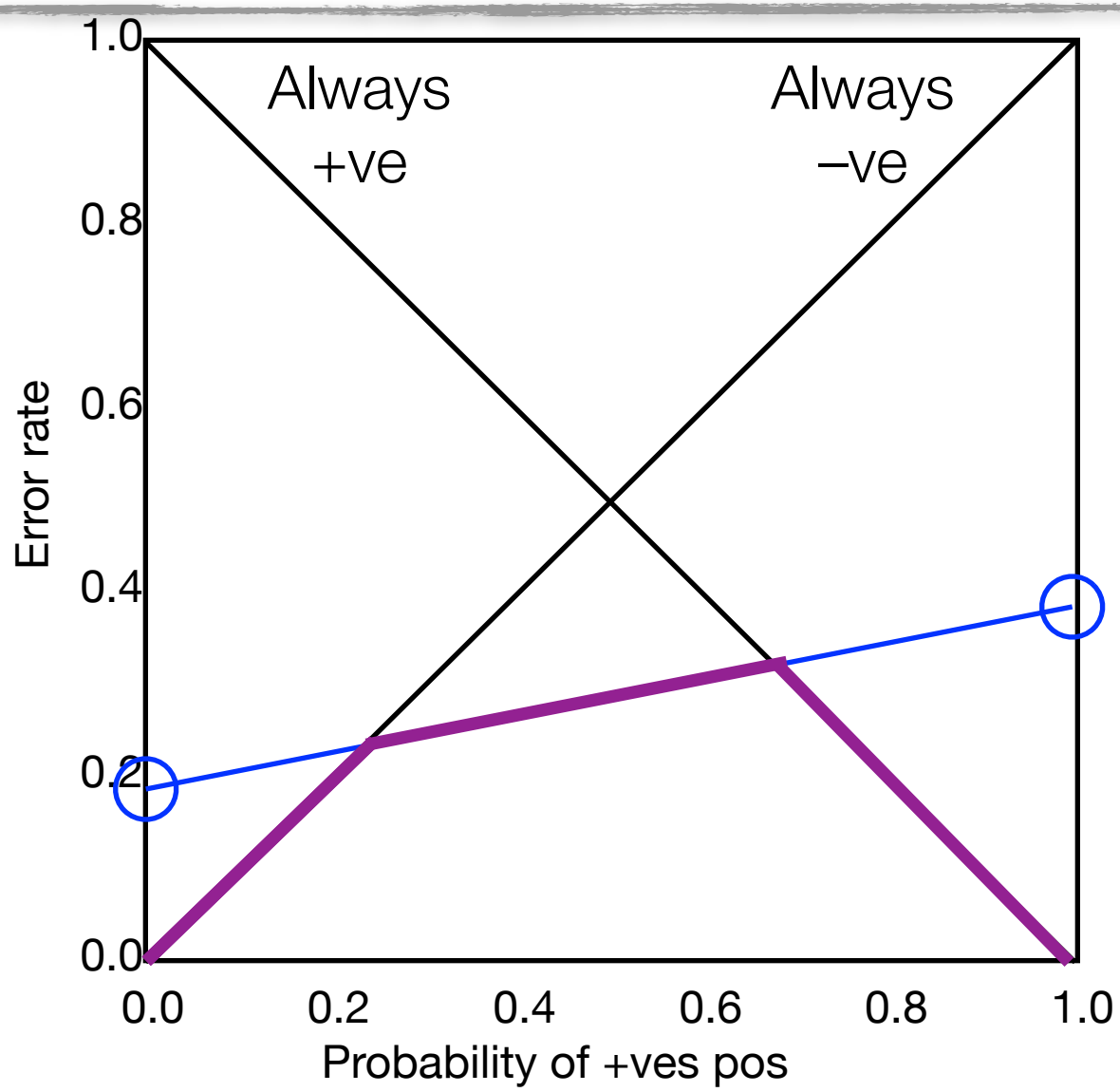
Classifier 3
tpr = 0.6
fpr = 0.2

Lower envelope



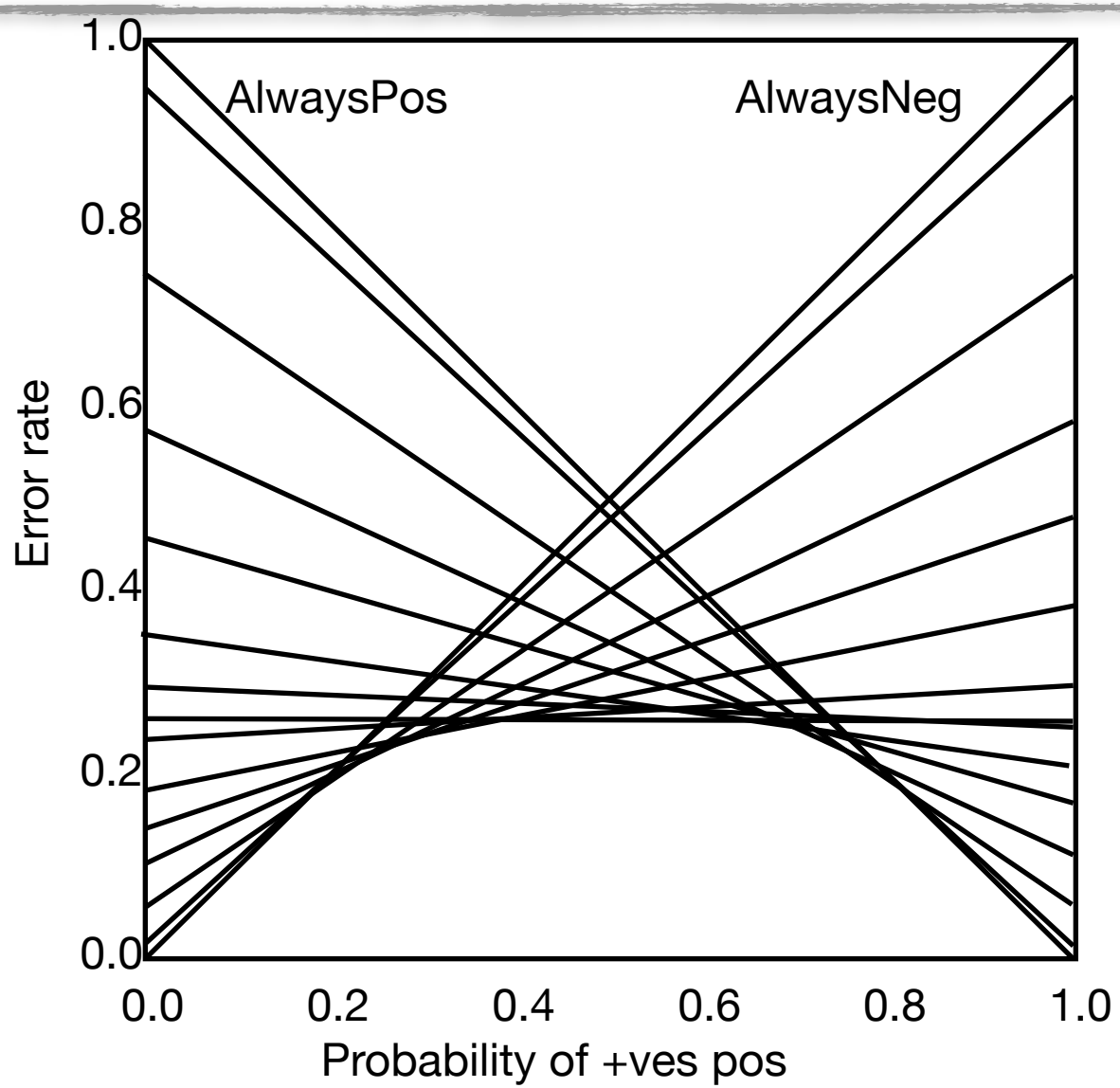
Classifier 3
tpr = 0.6
fpr = 0.2

Lower envelope

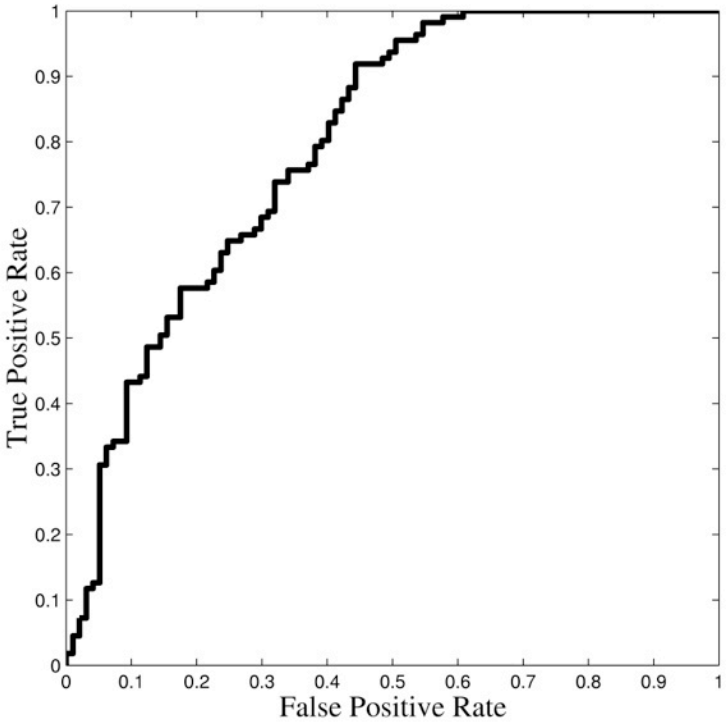


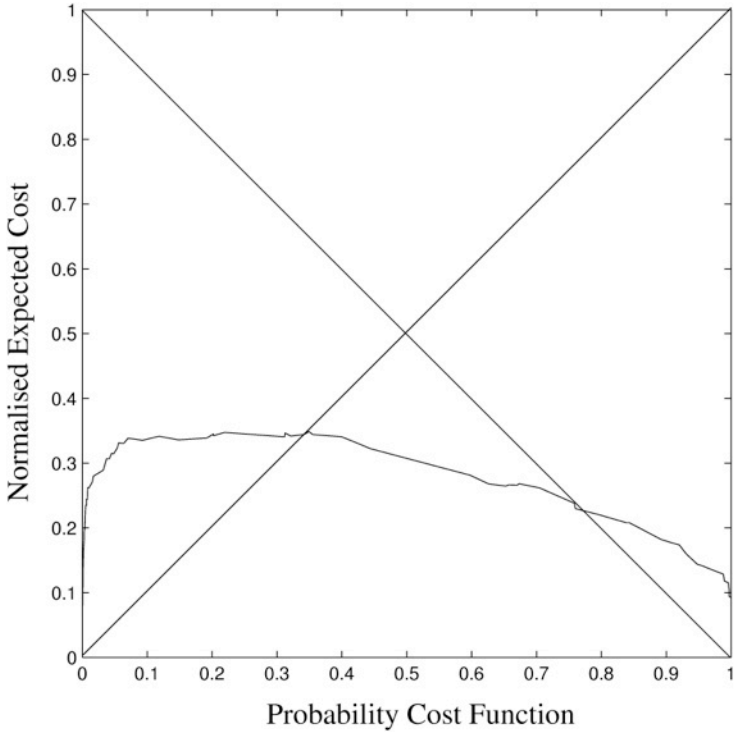
Classifier 3
tpr = 0.6
fpr = 0.2

Varying thresholds



ROC curve vs. cost curve



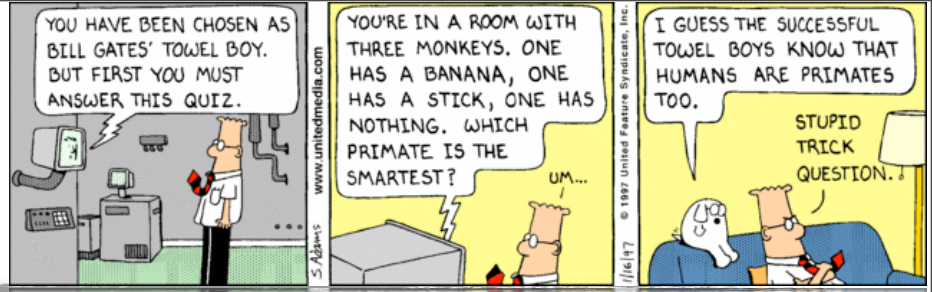


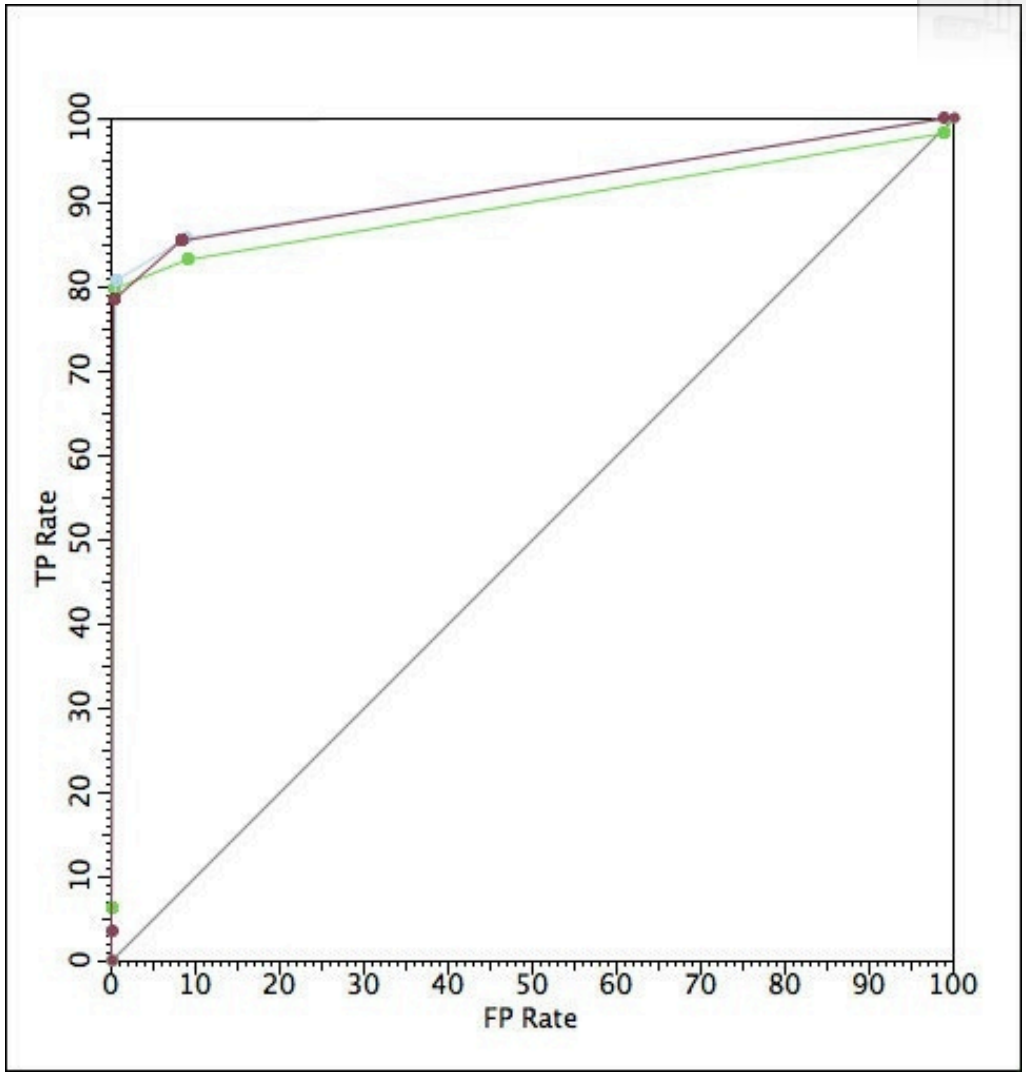
Part I: concluding remarks

- ROC analysis is useful for evaluating performance of classifiers and rankers
  - key idea: separate performance on classes

- ROC curves contain a wealth of information for understanding and improving performance of classifiers
  - requires visual inspection

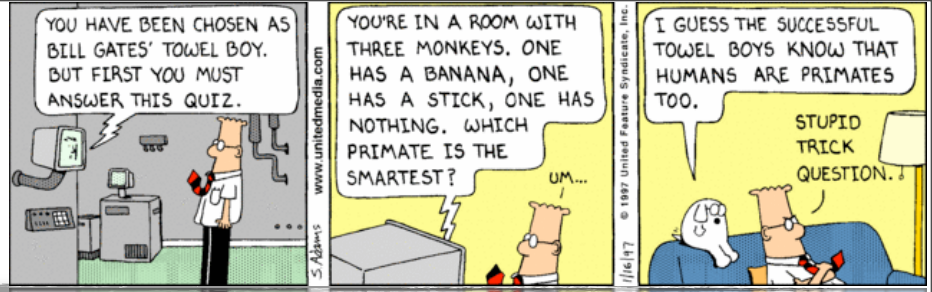
Quiz!

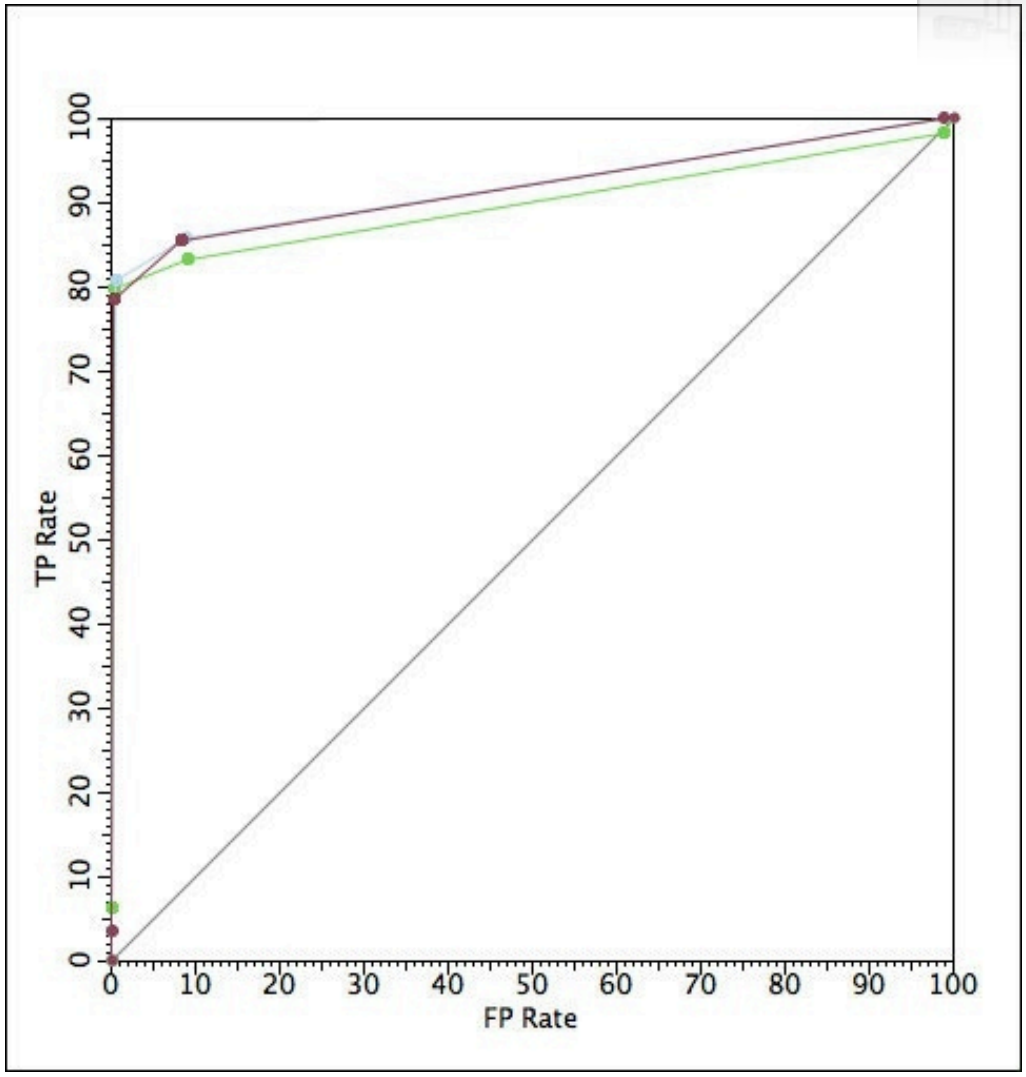




- Four models:
  - decision tree
  - k-nearest neighbour
  - linear classifier
  - naive Bayes
- trained on 2,000 examples and evaluated on
  - 18,000 test examples
  - 3,600 of those (20%)
  - 720 of those (4%)

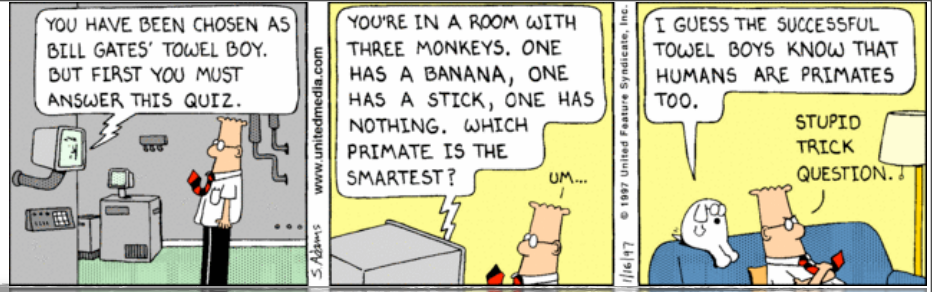
Quiz!

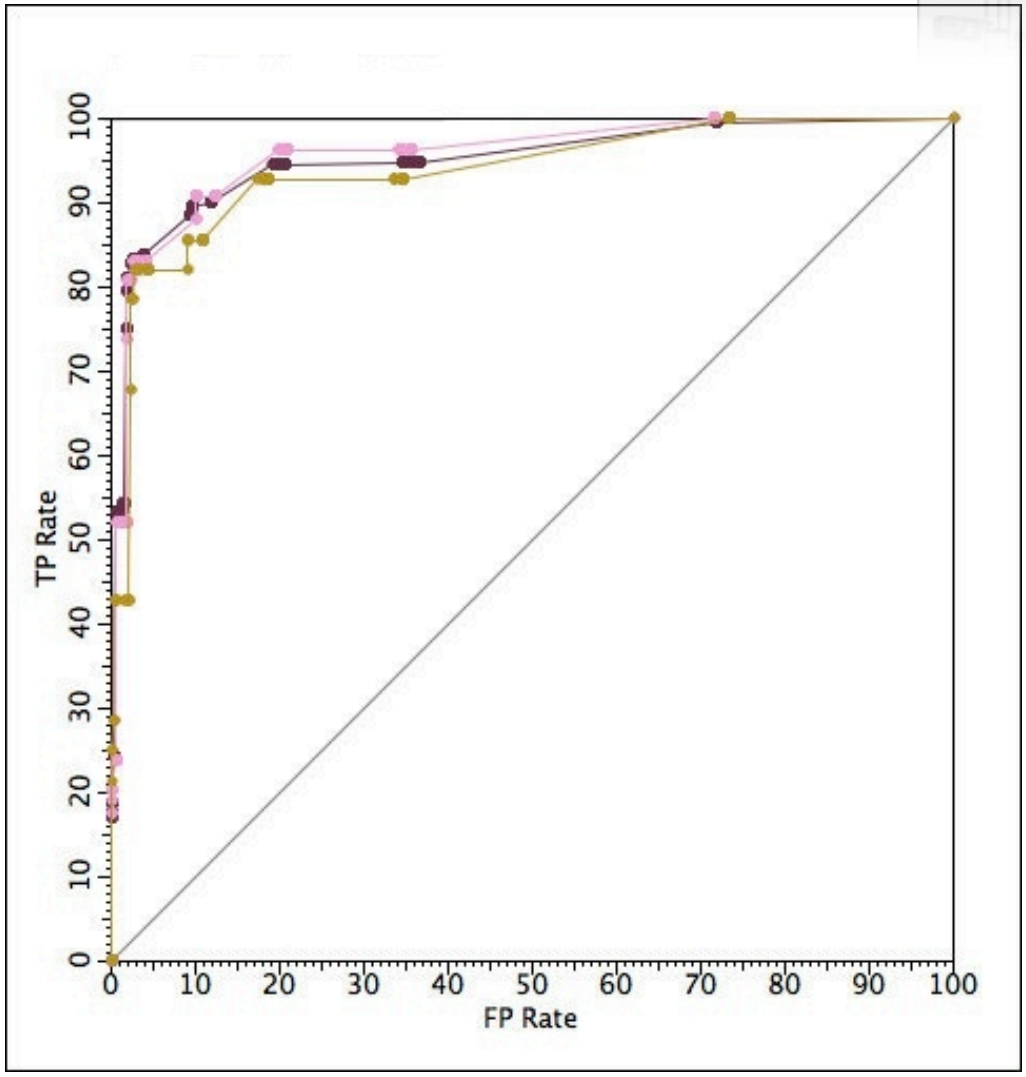




- Four models:
  - decision tree
  - k-nearest neighbour
  - linear classifier
  - naive Bayes
- trained on 2,000 examples and evaluated on
  - 18,000 test examples
  - 3,600 of those (20%)
  - 720 of those (4%)

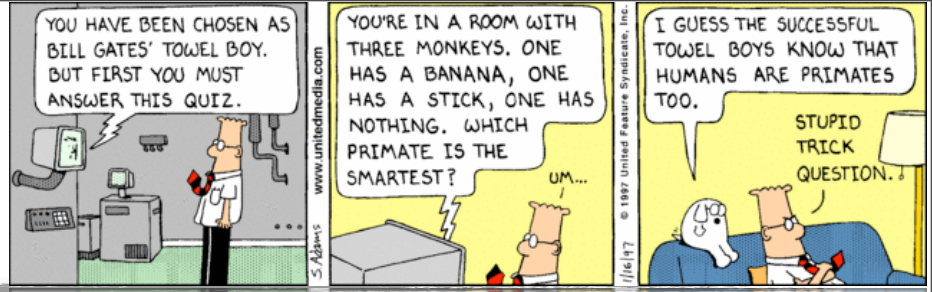
Quiz!

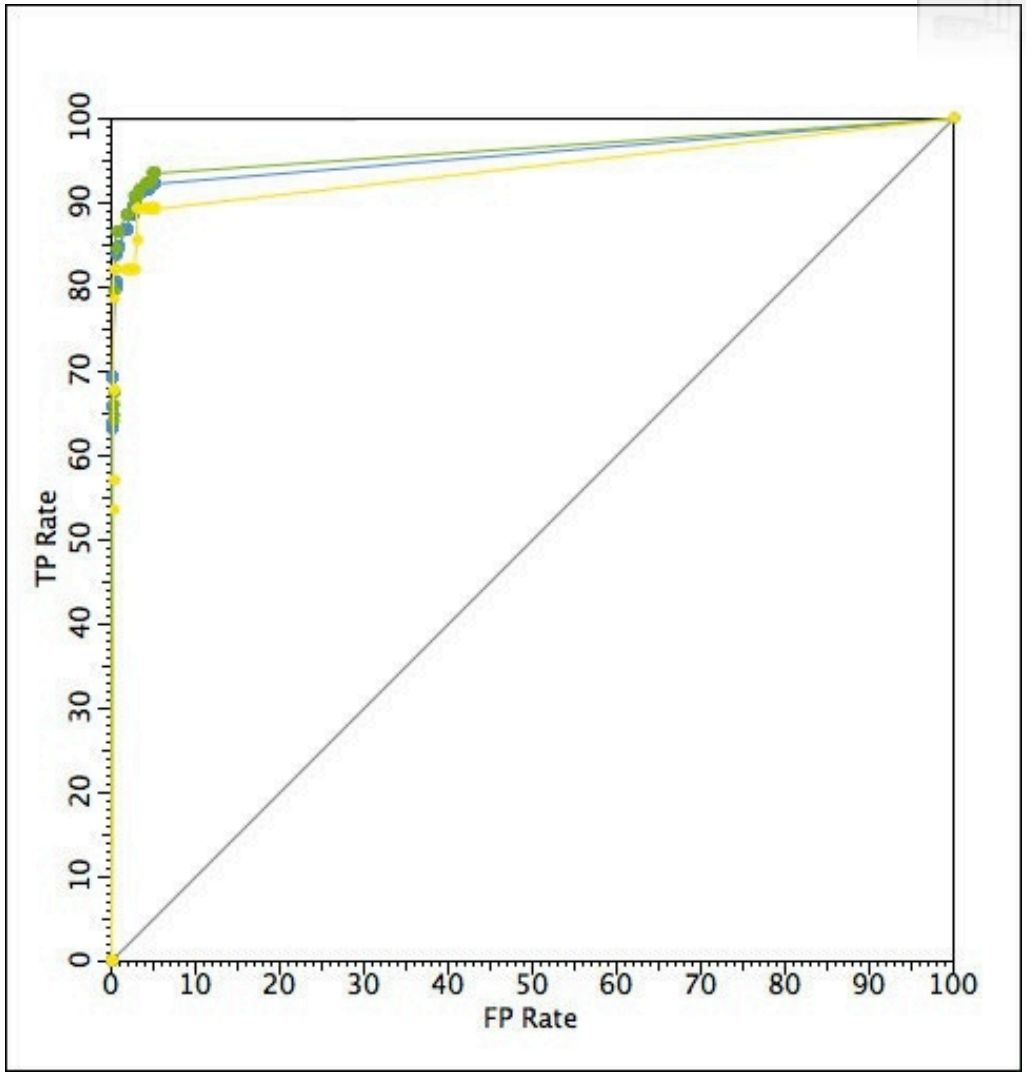




- Four models:
  - decision tree
  - k-nearest neighbour
  - linear classifier
  - naive Bayes
- trained on 2,000 examples and evaluated on
  - 18,000 test examples
  - 3,600 of those (20%)
  - 720 of those (4%)

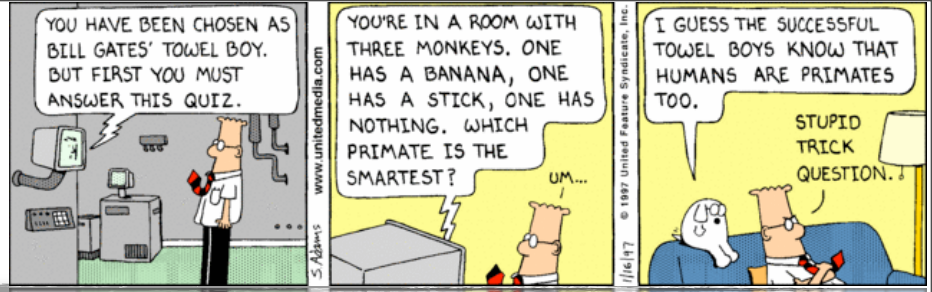
Quiz!

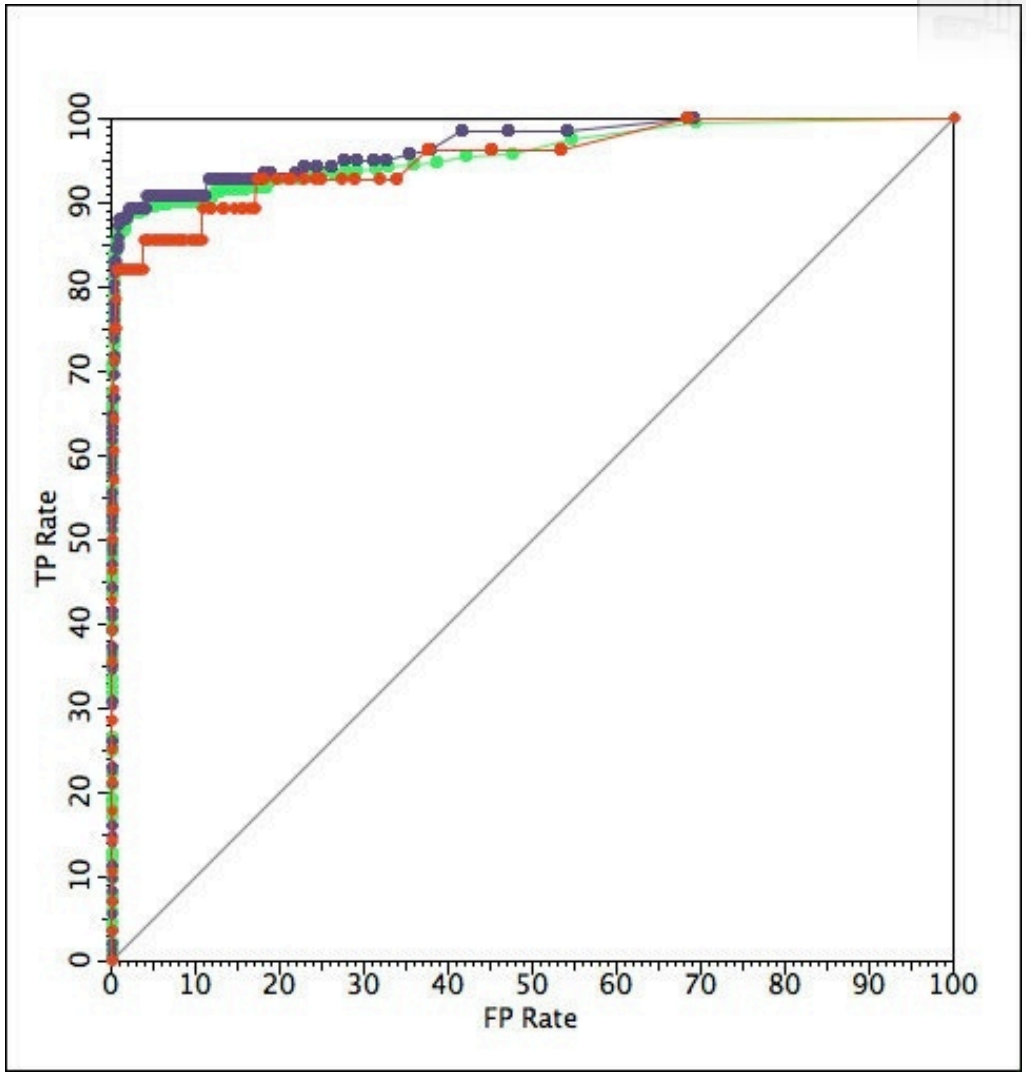




- Four models:
  - decision tree
  - k-nearest neighbour
  - linear classifier
  - naive Bayes
- trained on 2,000 examples and evaluated on
  - 18,000 test examples
  - 3,600 of those (20%)
  - 720 of those (4%)

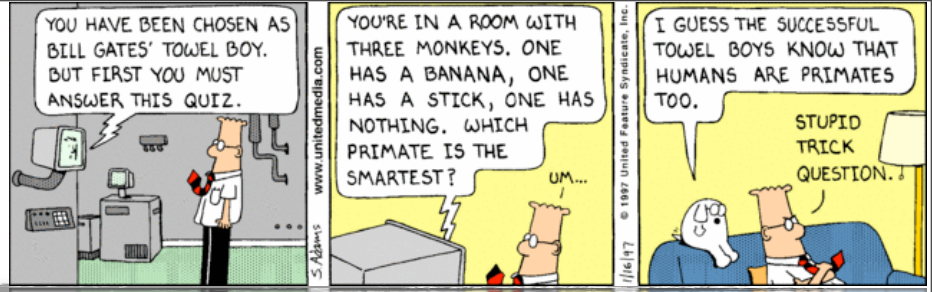
Quiz!

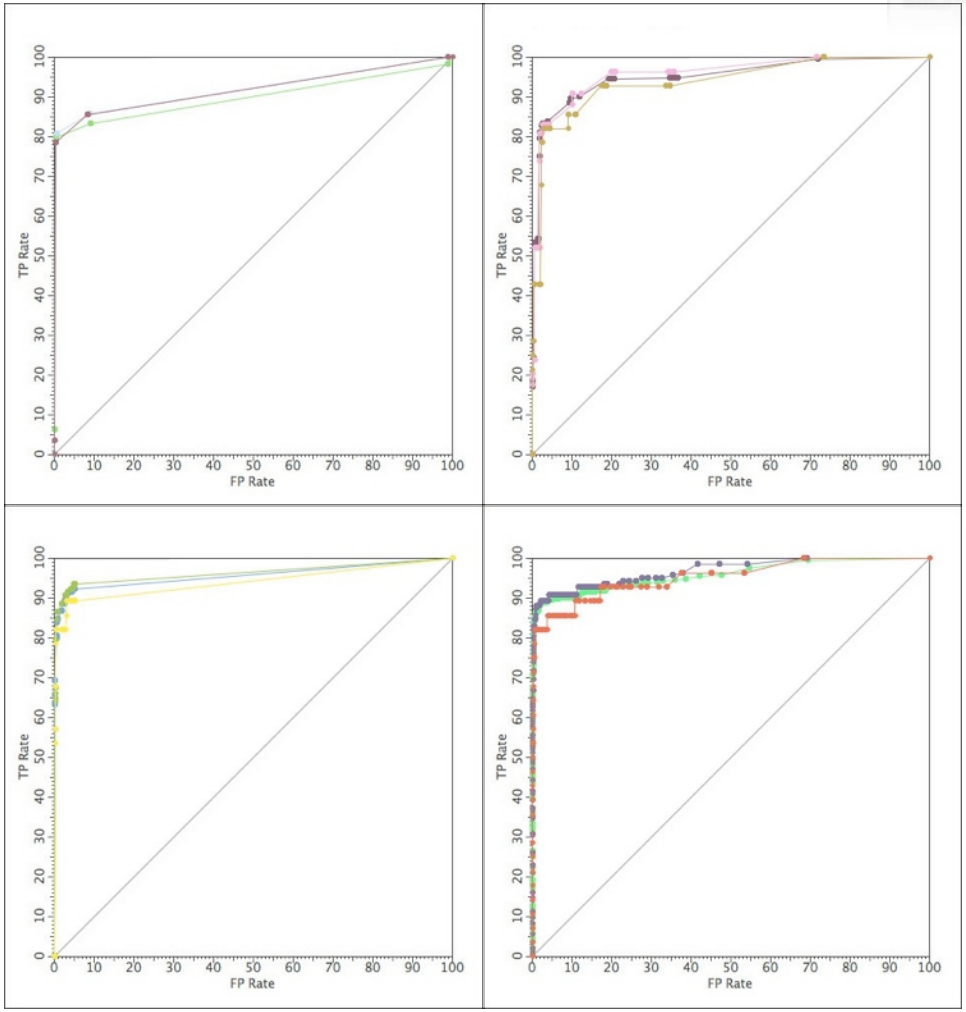




- Four models:
  - decision tree
  - k-nearest neighbour
  - linear classifier
  - naive Bayes
- trained on 2,000 examples and evaluated on
  - 18,000 test examples
  - 3,600 of those (20%)
  - 720 of those (4%)

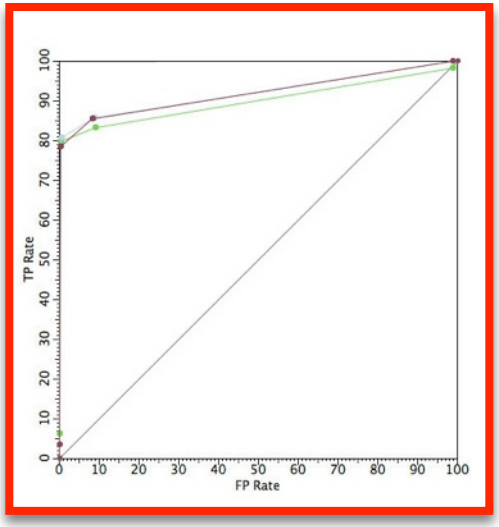
Quiz!

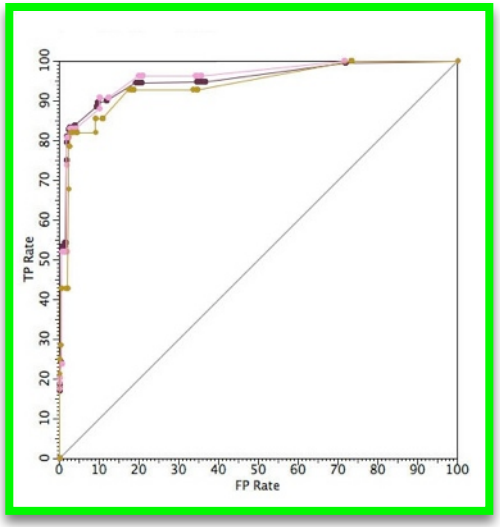


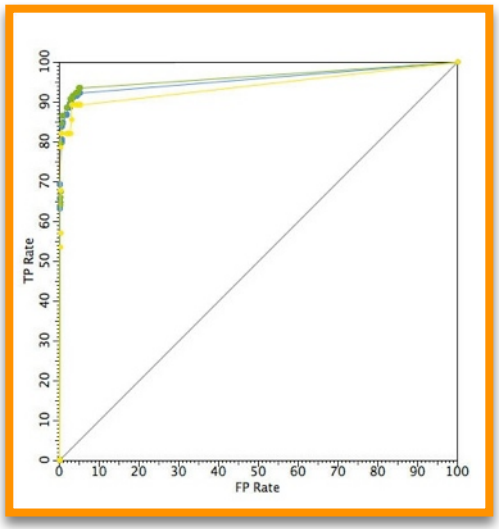


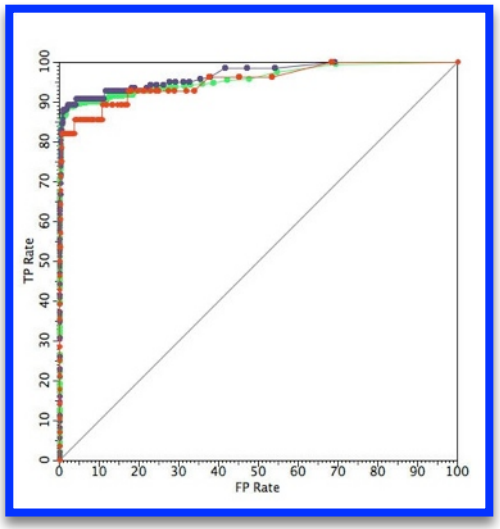
- Four models:
  - decision tree
  - k-nearest neighbour
  - linear classifier
  - naive Bayes
- trained on 2,000 examples and evaluated on
  - 18,000 test examples
  - 3,600 of those (20%)
  - 720 of those (4%)

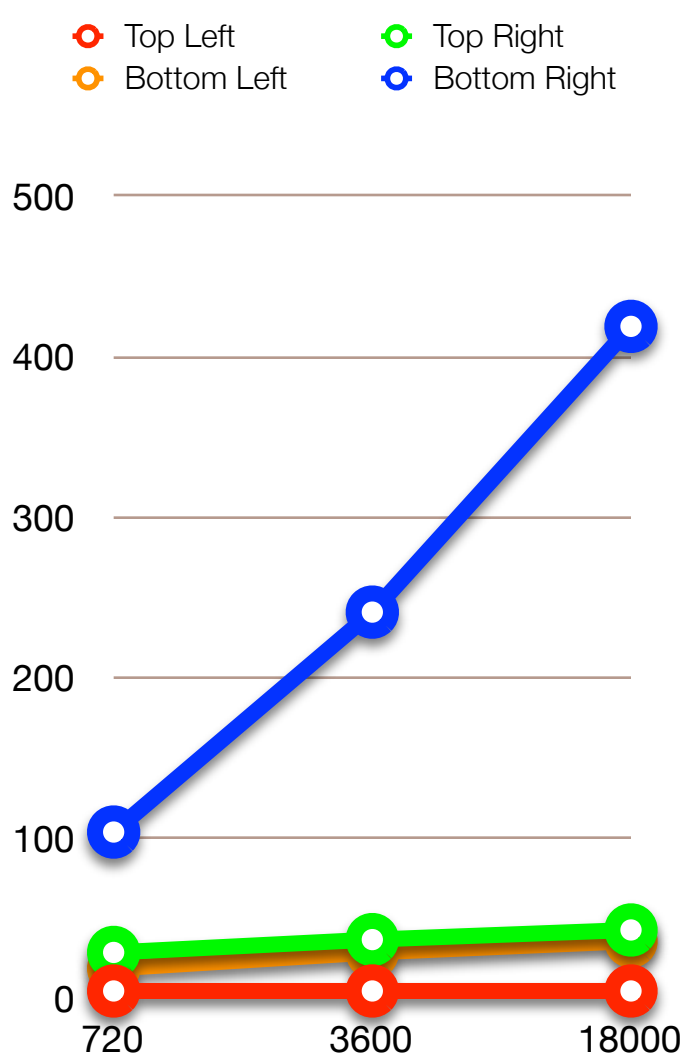
Which is which?







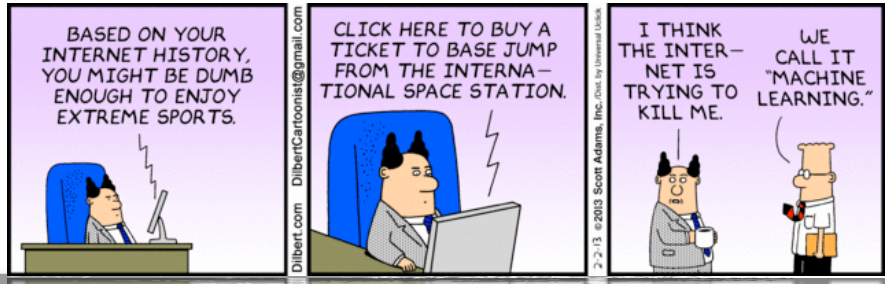




Part II: A broader view

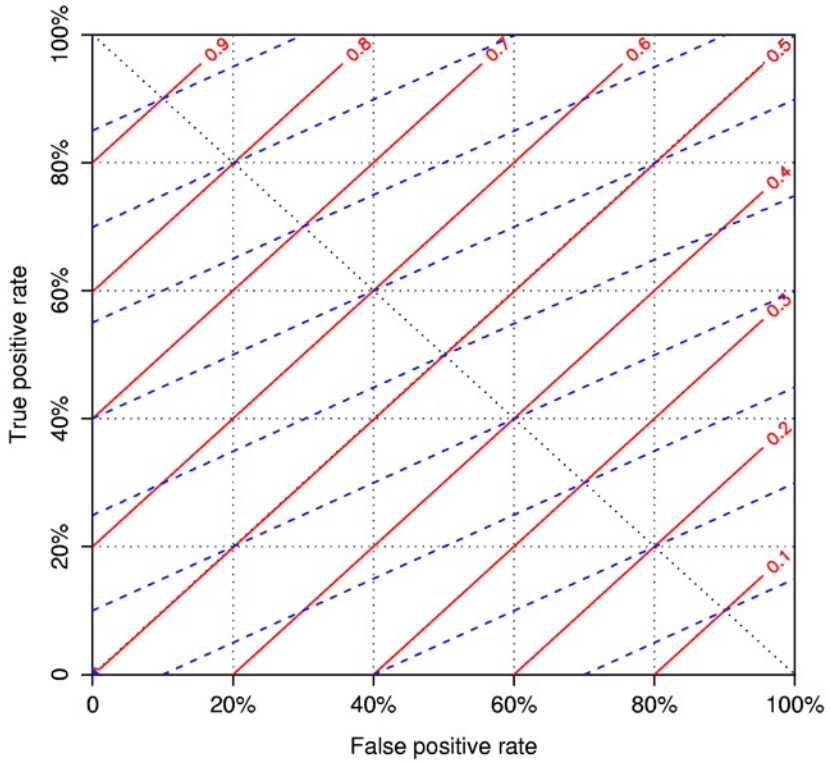
- Understanding ML metrics:
  - isometrics, basic types of linear isometric plots
  - linear metrics and equivalences between them
  - skew-sensitivity
  - non-linear metrics
- Model manipulation:
  - repairing concavities by locally adjusting rankings
- Multi-class ROC:
  - multi-objective optimisation, Pareto front, convex hull
  - multi-class AUC, multi-class calibration

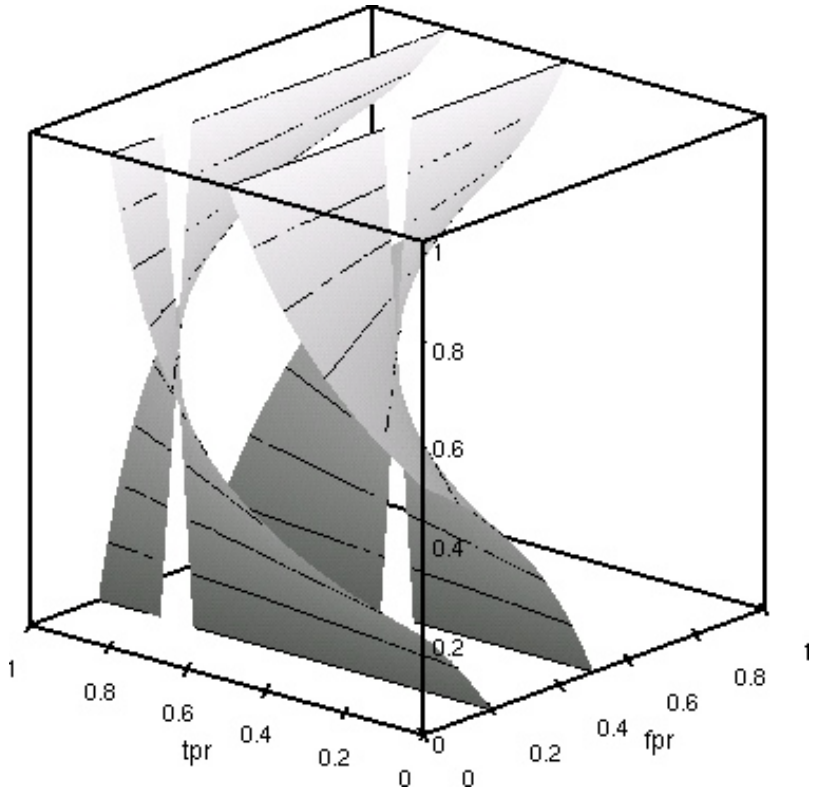
Understanding ML metrics



- We are referring here to metrics (or heuristics) that are used to rank (fpr,tpr) points
  - i.e., classifiers or parts of classifiers
    - NB. different sense of ranking than before!
- Metrics are equivalent if their rankings are the same
  - absolute value of metric not important
- This can be visualised very clearly by means of ROC isometrics
  - additional benefit of studying skew-sensitivity
  - see (Flach, 2003) and (Fürnkranz & Flach, 2003)

Iso-accuracy lines revisited





In 2D ROC space
c = 1, c = 1/2

In 3D ROC space
acc = 0.5, acc = 0.8

Isometrics and skew ratio

- Accuracy is weighted average of true positive/negative rates:

$$acc = pos \cdot tpr + neg \cdot (1 - fpr) = \frac{tpr + c \cdot (1 - fpr)}{c + 1}$$

- Skew ratio indicates relative importance of negatives over positives
  - without costs: c = neg/pos

- Isometric plots show contour lines in 2D ROC space for a given metric with skew ratio as parameter

Skew-sensitivity

- Strongly skew-insensitive metric is independent of skew ratio
  - isometric surfaces in 3D ROC space are vertical
  - can be obtained for any metric by fixing c
- Weakly skew-insensitive metric has the same isometric landscape for different values of c
  - any collection of ROC points is ranked the same way, regardless of c
- Line of skew-indifference: points where the metric is independent of c
  - for accuracy, this is the line tpr+fpr−1=0

Types of isometric plots

- Parallel linear isometrics
  - accuracy, weighted relative accuracy (WRAcc)
- Rotating linear isometrics
  - precision, lift, F-measure
- Non-linear isometrics
  - decision tree splitting criteria

Symmetries

- Inverting predictions of classifier
  - ROC space: point-mirroring through (0.5, 0.5)
  - contingency table: swapping columns
- Inverting test labels
  - ROC space: mirroring along ascending diagonal
  - contingency table: swapping rows
    - affects skew ratio (c becomes 1/c), so a test for skew-insensitivity
- Inverting both predictions and test labels
  - ROC space: mirroring along descending diagonal
  - contingency table: swapping rows and columns

Precision or confidence

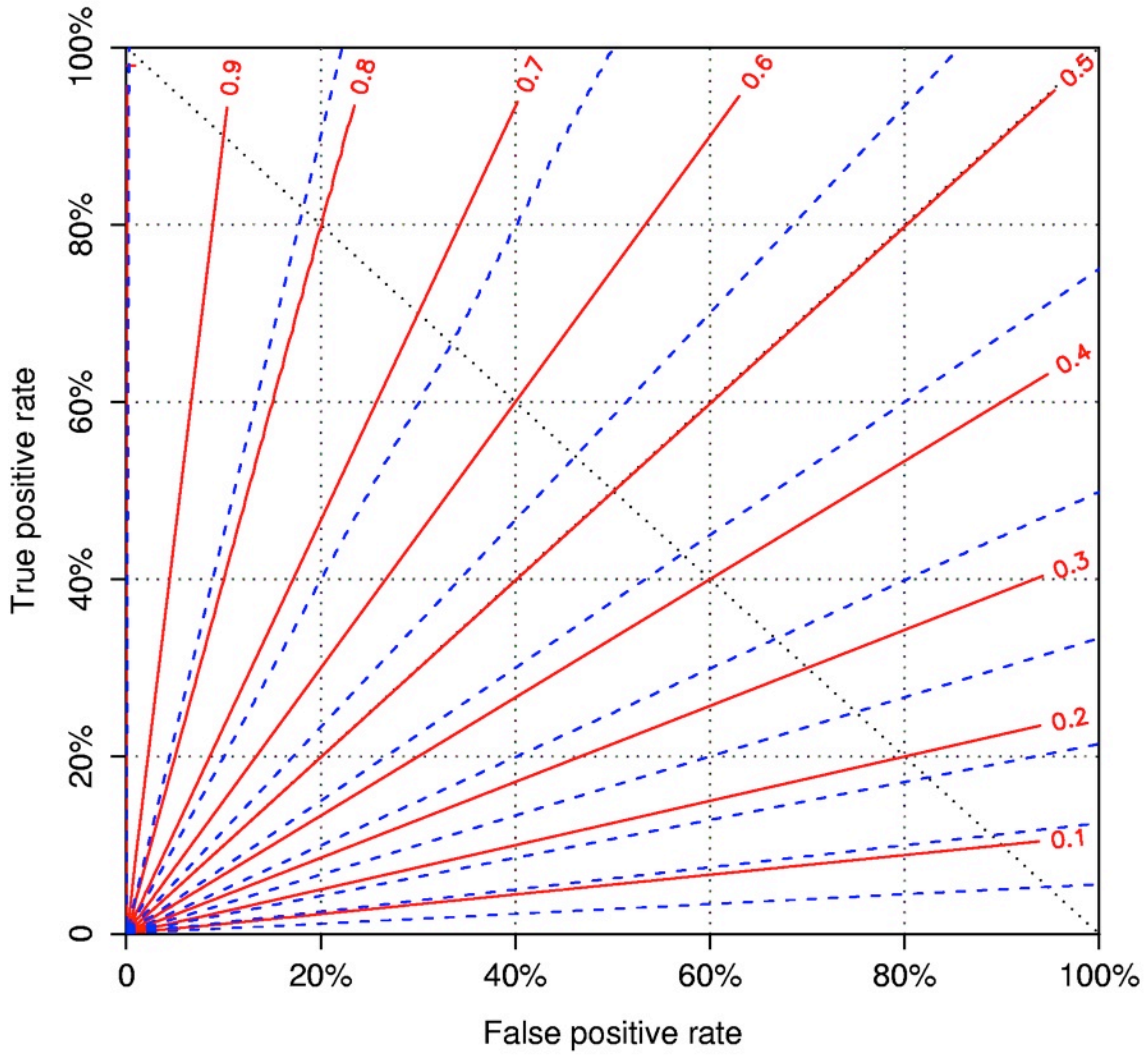
- Precision is defined as

$$prec = \frac{pos \cdot tpr}{pos \cdot tpr + neg \cdot fpr} = \frac{tpr}{tpr + c \cdot fpr}$$

- Weakly skew-insensitive, rotating isometrics
  - on tpr = fpr diagonal, prec = pos
  - singular point for tpr = fpr = 0

- Two variants with fixed value on diagonal
  - relative precision: prec–pos
  - lift: prec/pos

Precision isometrics



c = 1,
c = 1/2

F-measure

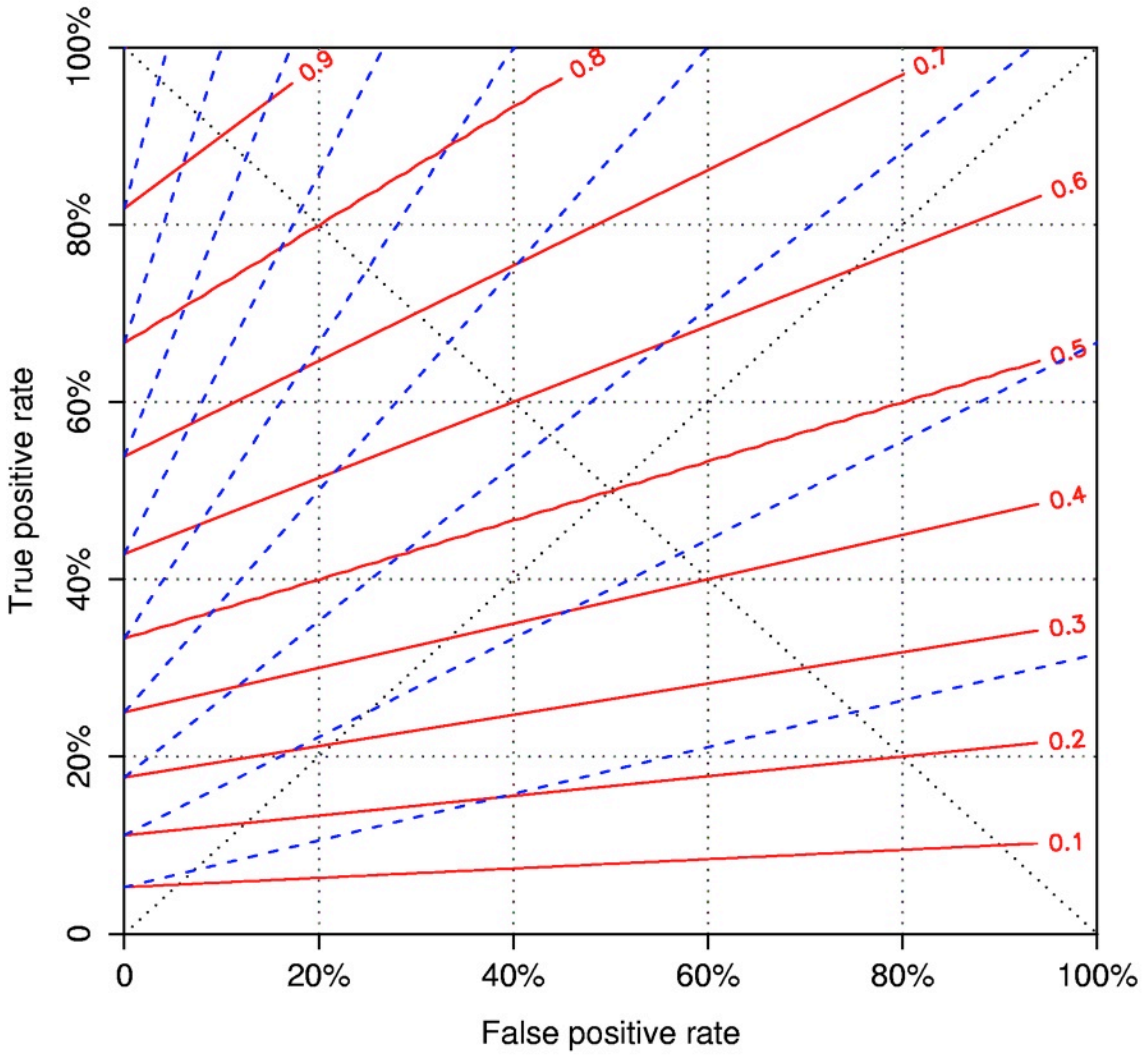
- F-measure is harmonic average of precision and recall (true positive rate)
  - alternatively, F-measure = precision (recall) with FP (FN) replaced with (FP+FN)/2

- In ROC notation: $$F = \frac{2\,tpr}{1 + tpr + c \cdot fpr}$$

- Rank-equivalent but simpler: $$G = \frac{tpr}{1 + c \cdot fpr}$$

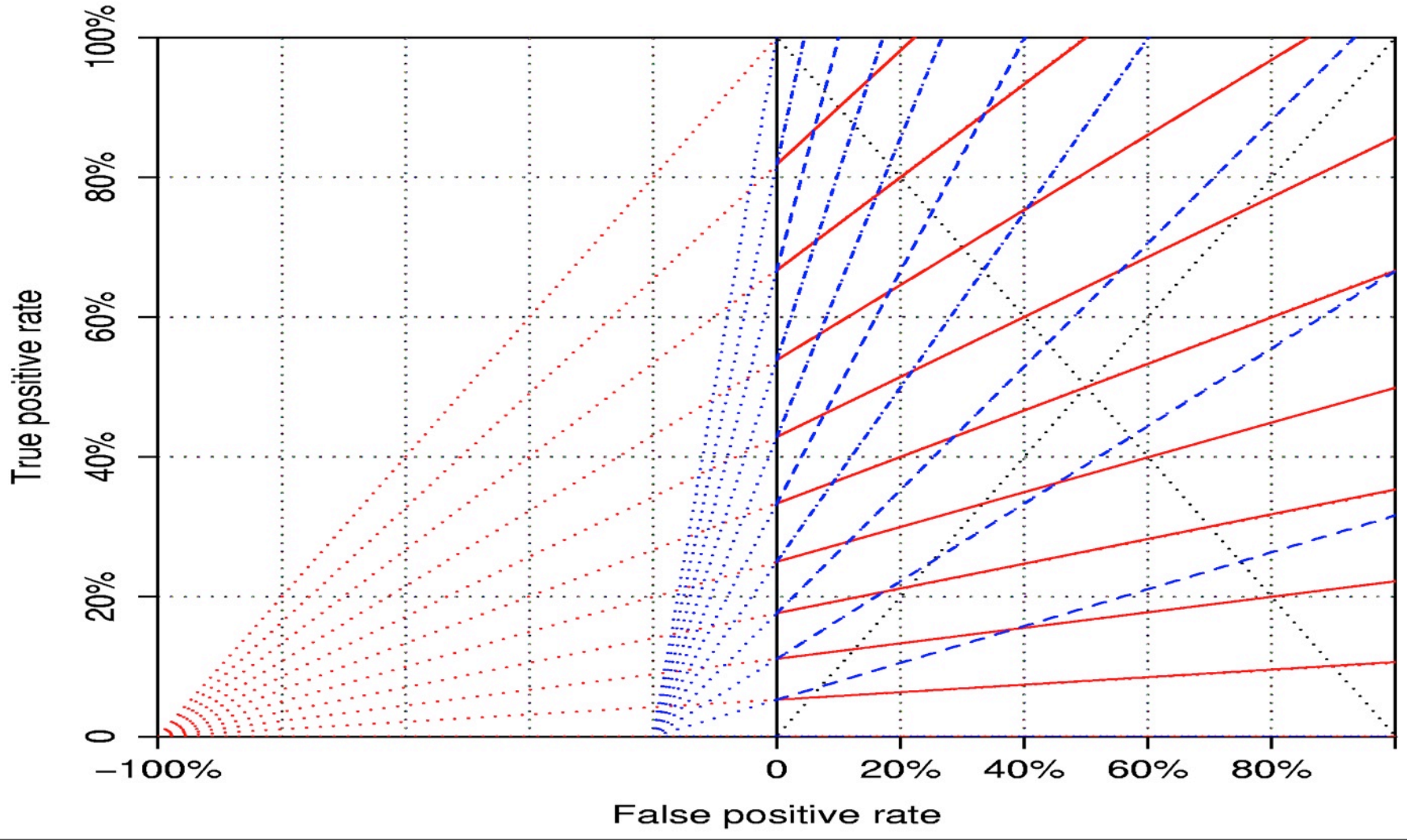
- fpr=0 is line of skew-indifference
- Singular point for tpr = 0, fpr = –1/c

F-measure isometrics



c = 1,
c = 5

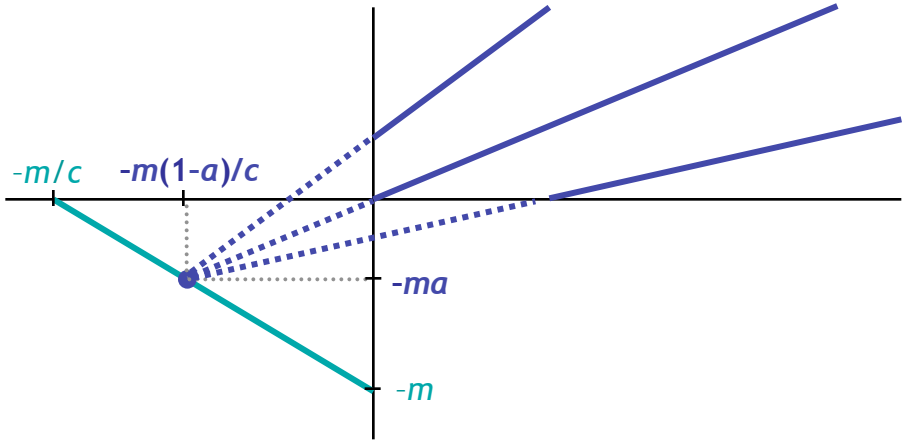
F-measure isometrics



Generalised linear isometrics

- Laplace correction and m-estimate are other examples which translate the rotation point

- General form: $\frac{tpr + ma}{tpr + c \cdot fpr + m}$
  - m=0: precision
  - m→∞: parallel isometrics with slope $\frac{ac}{1-a}$
    - e.g. accuracy: a=1/2



Linear metrics: summary

Metric	Formula	Skew-insensitive version	Isometric slope
Accuracy	$\frac{tpr + c(1 - fpr)}{c + 1}$	$\frac{(tpr + 1 - fpr)}{2}$	c
WRAcc*	$\frac{4c}{(c+1)^2}(tpr - fpr)$	$tpr - fpr$	1
Precision*	$\frac{tpr}{tpr + c \cdot fpr}$	$\frac{tpr}{tpr + fpr}$	$\frac{tpr}{fpr}$
Lift*	$\frac{c+1}{2}\frac{tpr}{tpr + c \cdot fpr}$	$\frac{tpr}{tpr + fpr}$	
Relative precision*	$\frac{2c}{c+1}\frac{(tpr - fpr)}{tpr + c \cdot fpr}$	$\frac{tpr - fpr}{tpr + fpr}$	
F-measure	$\frac{2tpr}{tpr + c \cdot fpr + 1}$	$\frac{2tpr}{tpr + fpr + 1}$	$\frac{tpr}{fpr + 1/c}$
G-measure	$\frac{tpr}{c \cdot fpr + 1}$	$\frac{tpr}{fpr + 1}$	

All metrics are re-scaled such that the strongly skew-insensitive version is in [0,1] or [–1,1]. An asterisk (*) denotes weak skew-insensitivity.

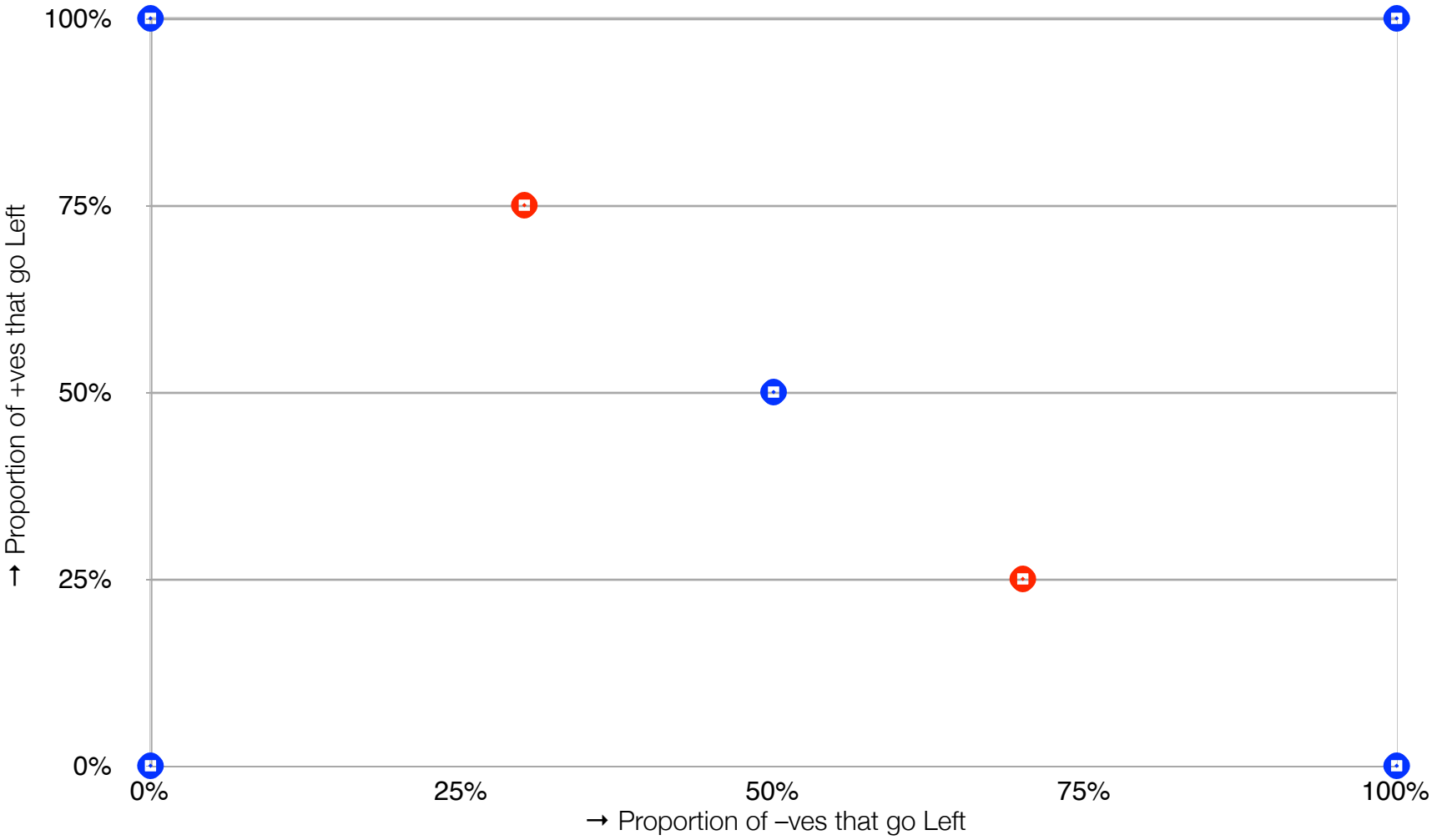
Splitting criteria

- Splitting criteria are invariant under swapping columns, i.e. point-mirroring through (0.5,0.5)
  - if skew-insensitive then isometrics are symmetric across both diagonals
- They compare impurity of the parent with weighted average impurity of the children:

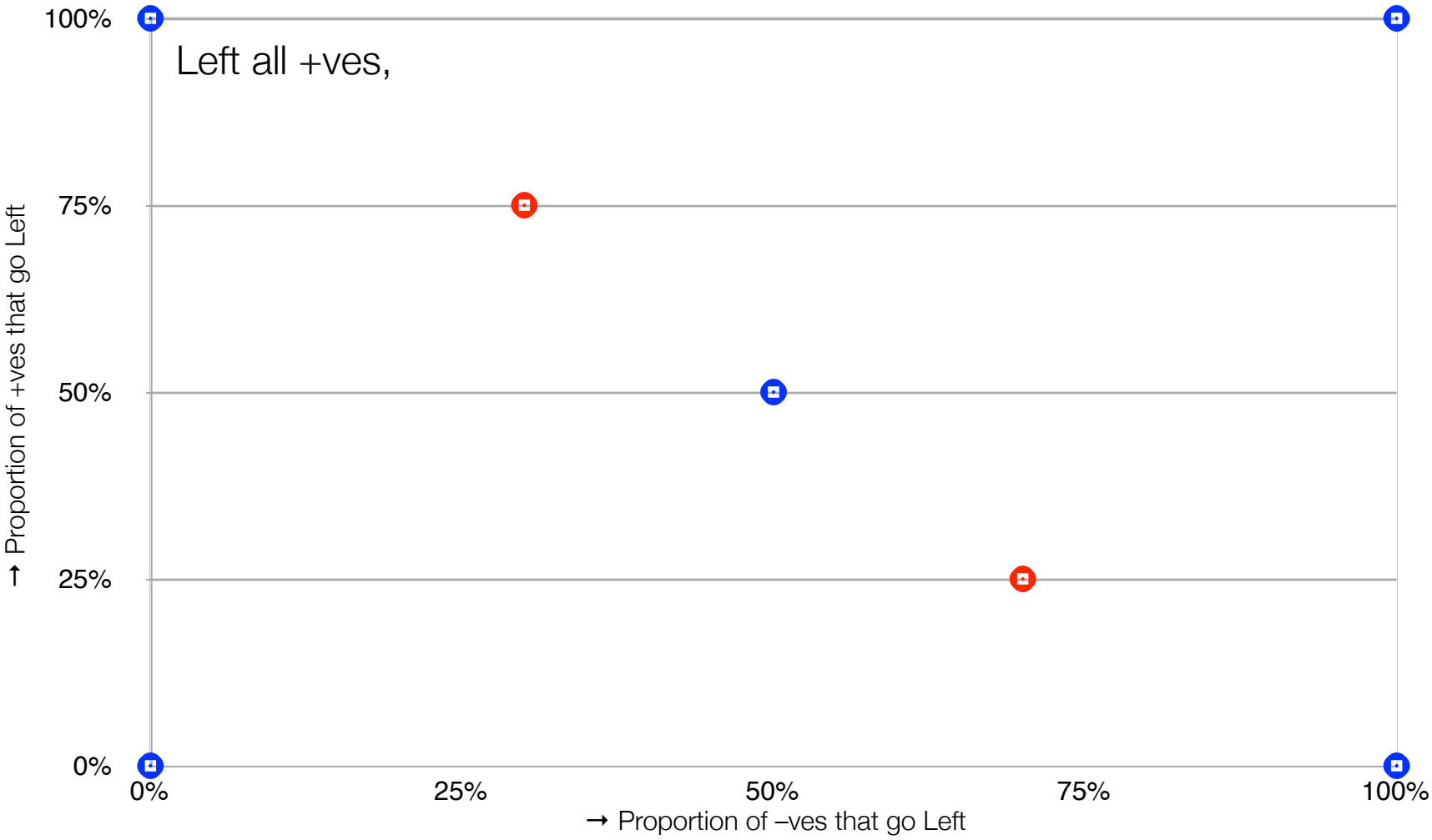
$$\text{Imp}\left(\frac{Pos}{N}, \frac{Neg}{N}\right) - \frac{Left}{N}\text{Imp}\left(\frac{TP}{Left}, \frac{FP}{Left}\right) - \frac{Right}{N}\text{Imp}\left(\frac{FN}{Right}, \frac{TN}{Right}\right)$$

	Left child	*Right child*	
Actual ⊕	**TP**	**FN**	Pos
Actual ⊖	**FP**	**TN**	Neg
	Left	Right	N

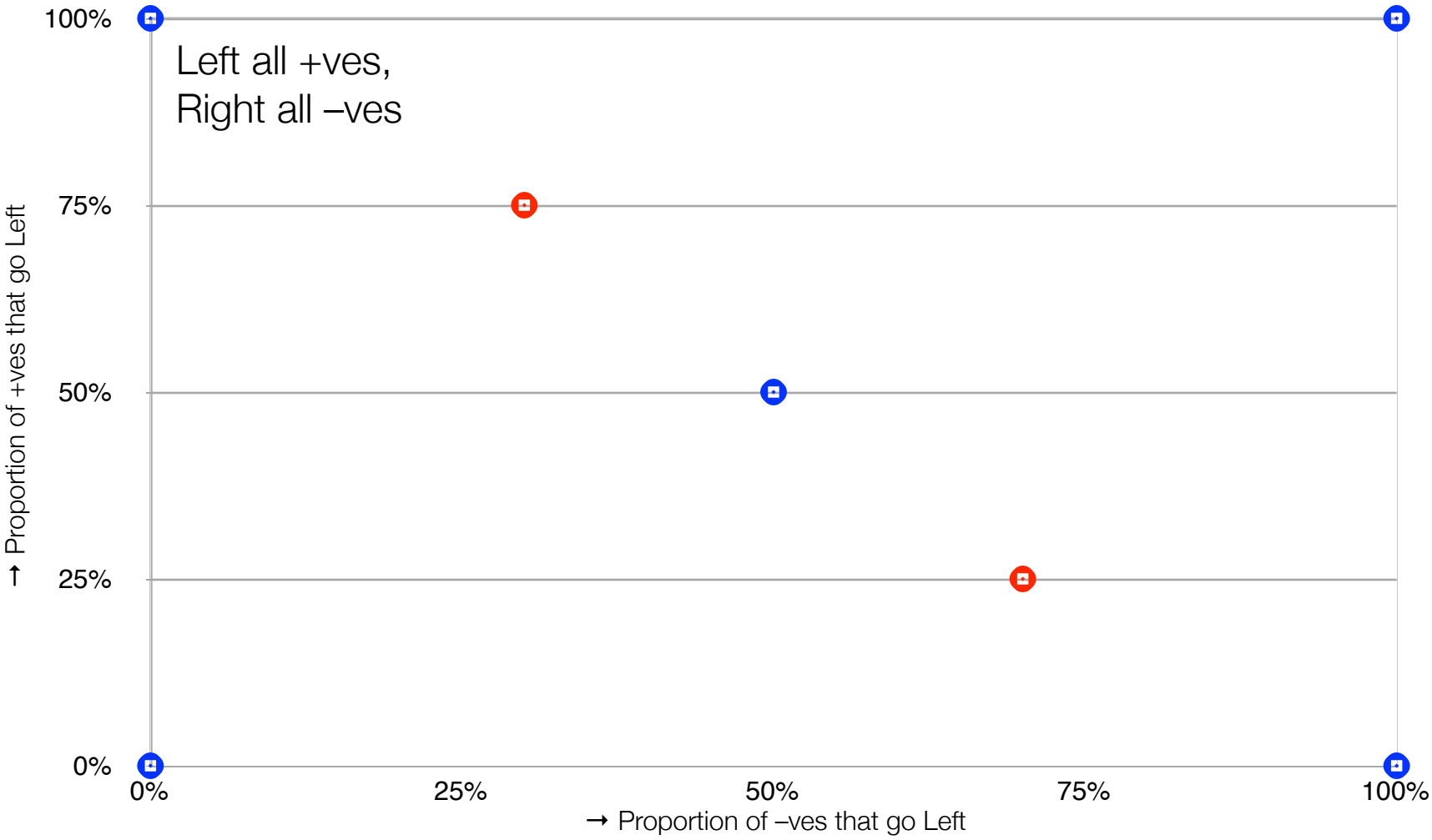
ROC space for splitting criteria



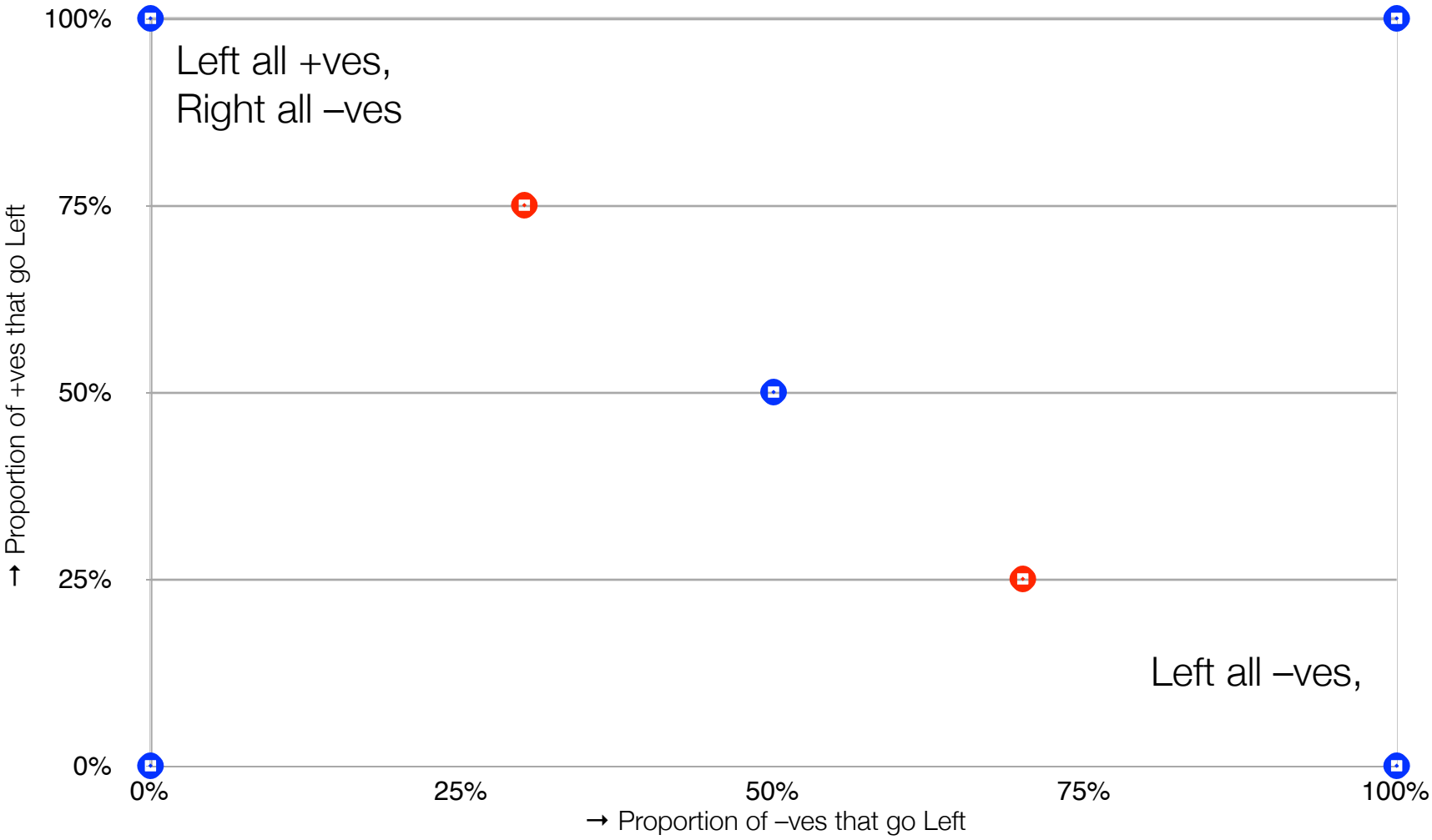
ROC space for splitting criteria



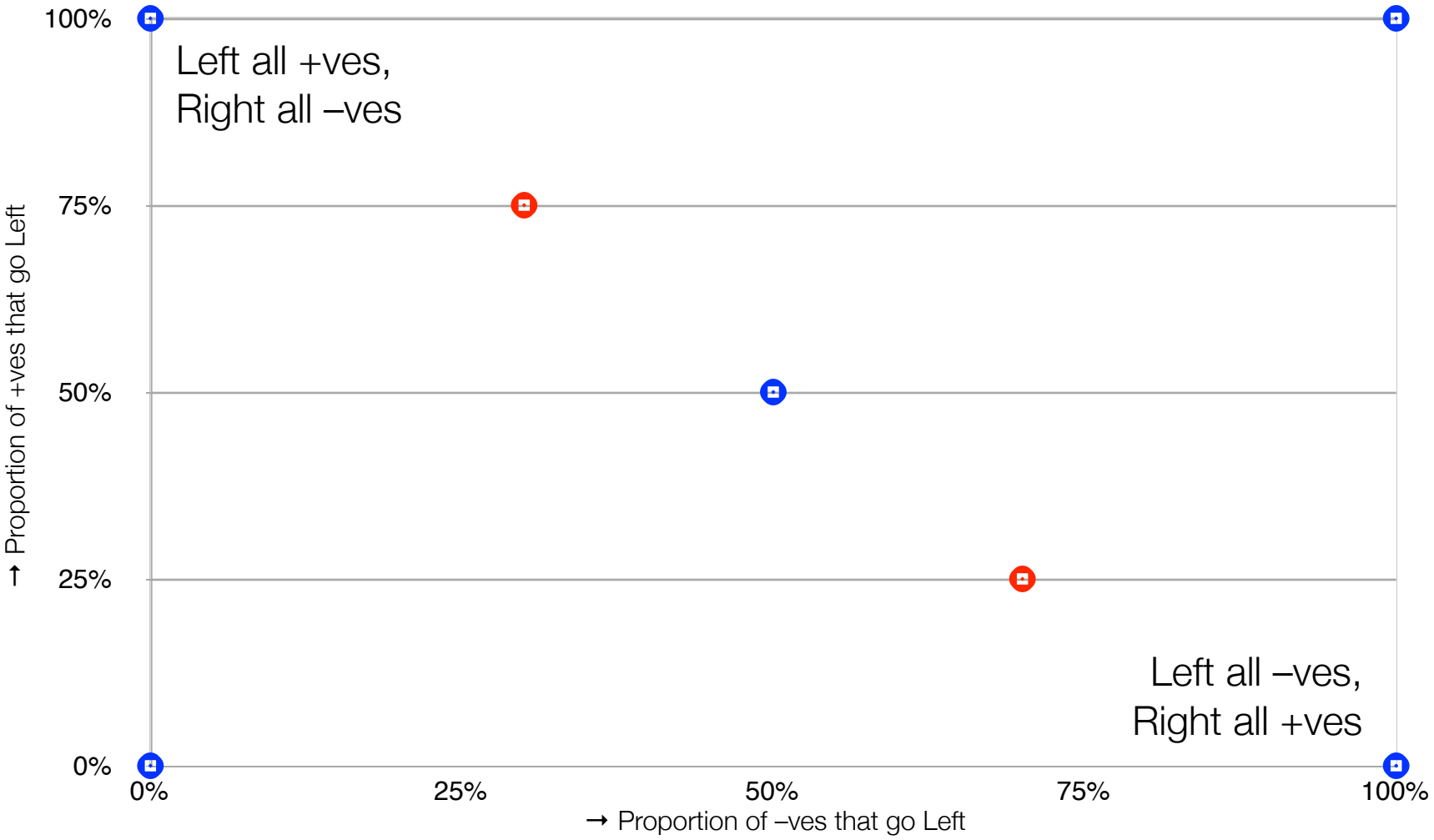
ROC space for splitting criteria



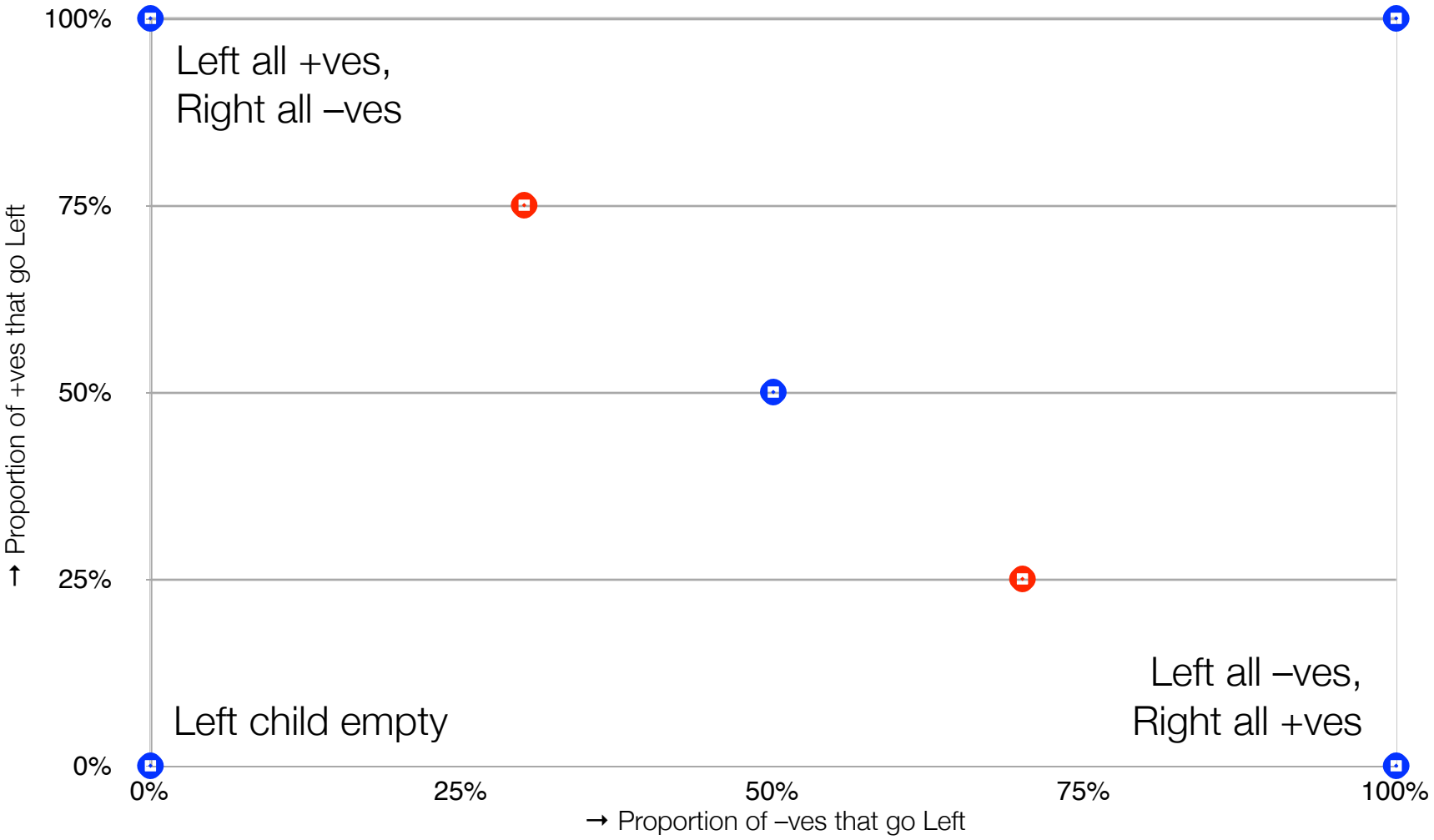
ROC space for splitting criteria



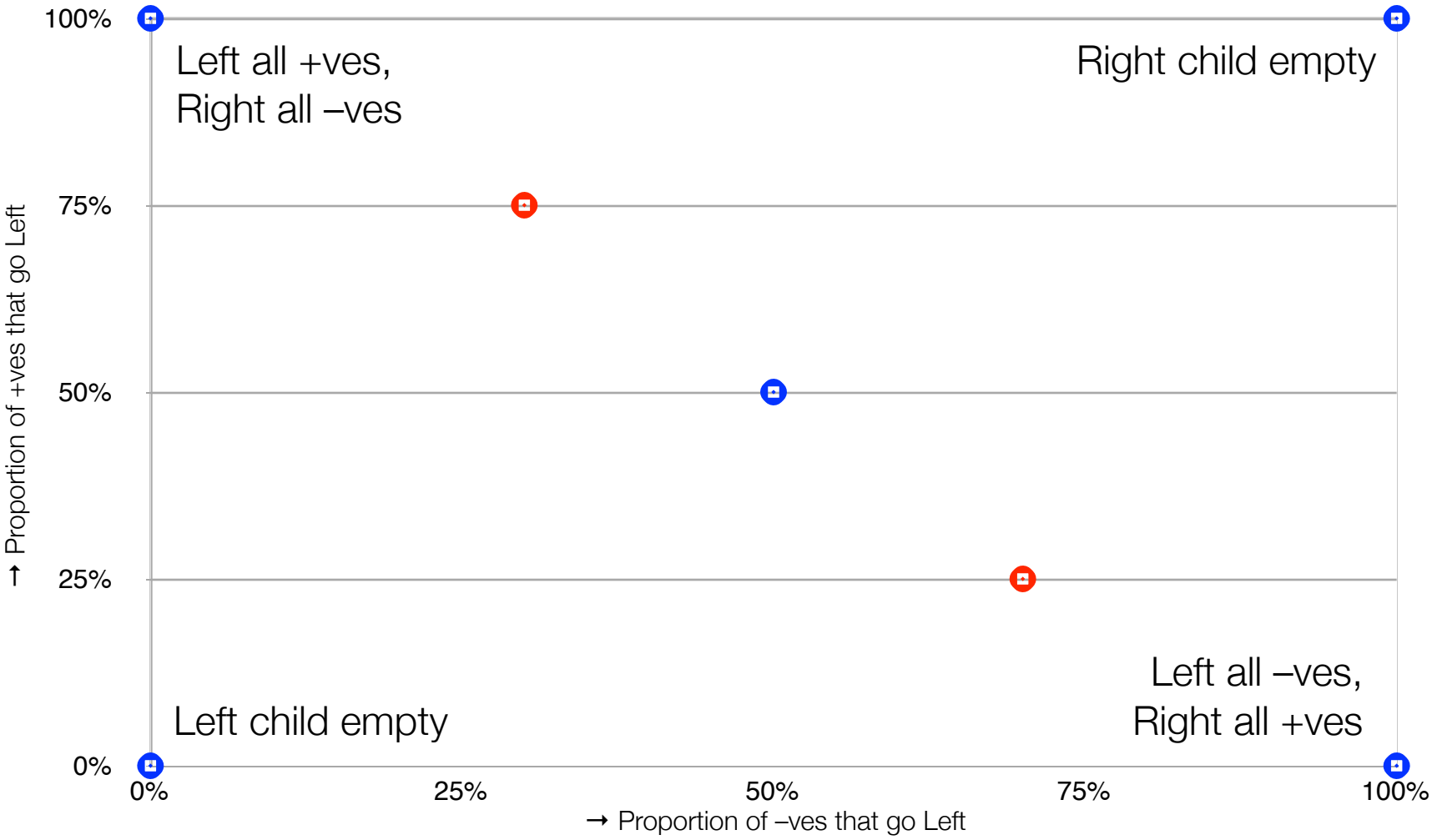
ROC space for splitting criteria



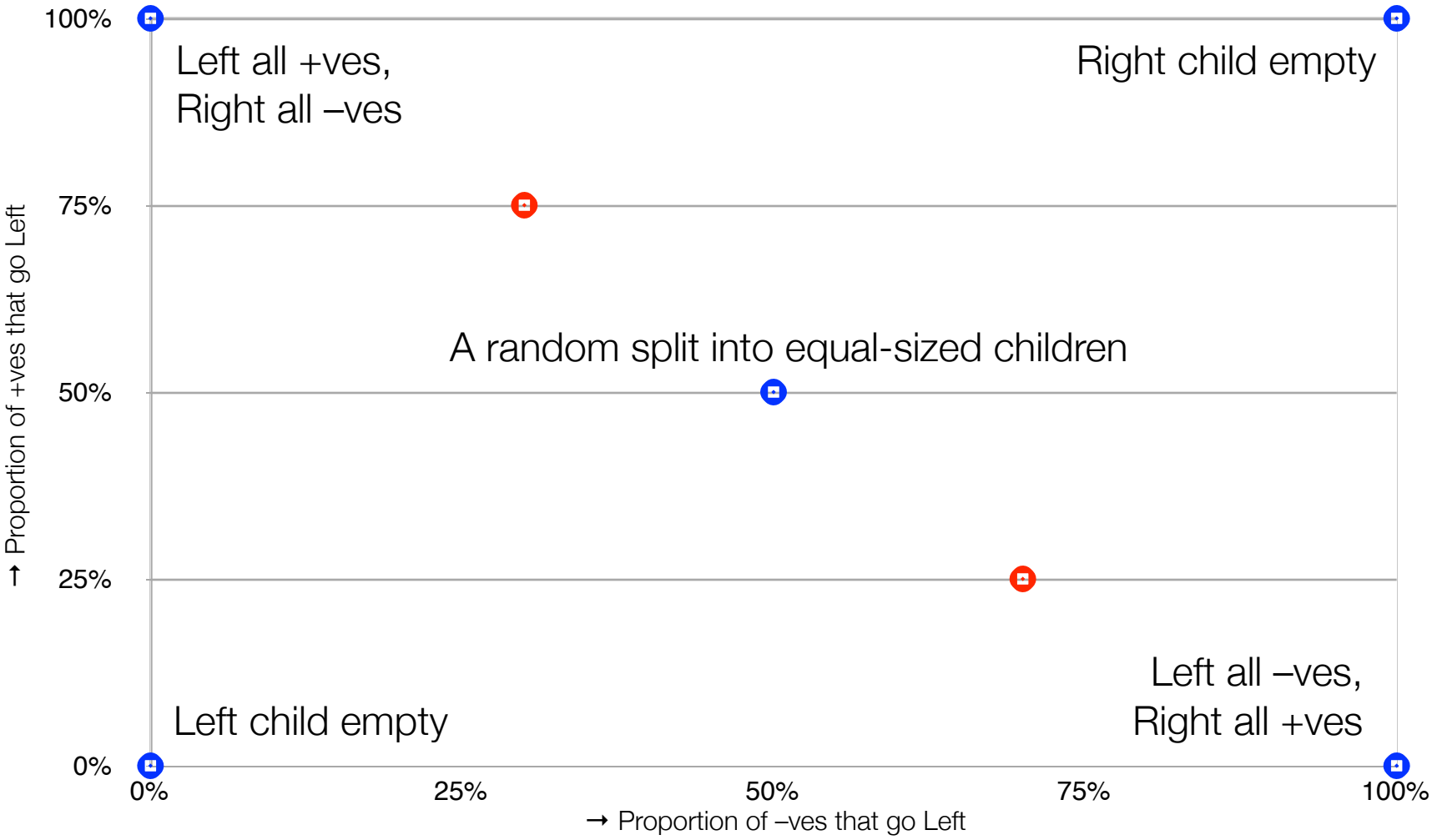
ROC space for splitting criteria



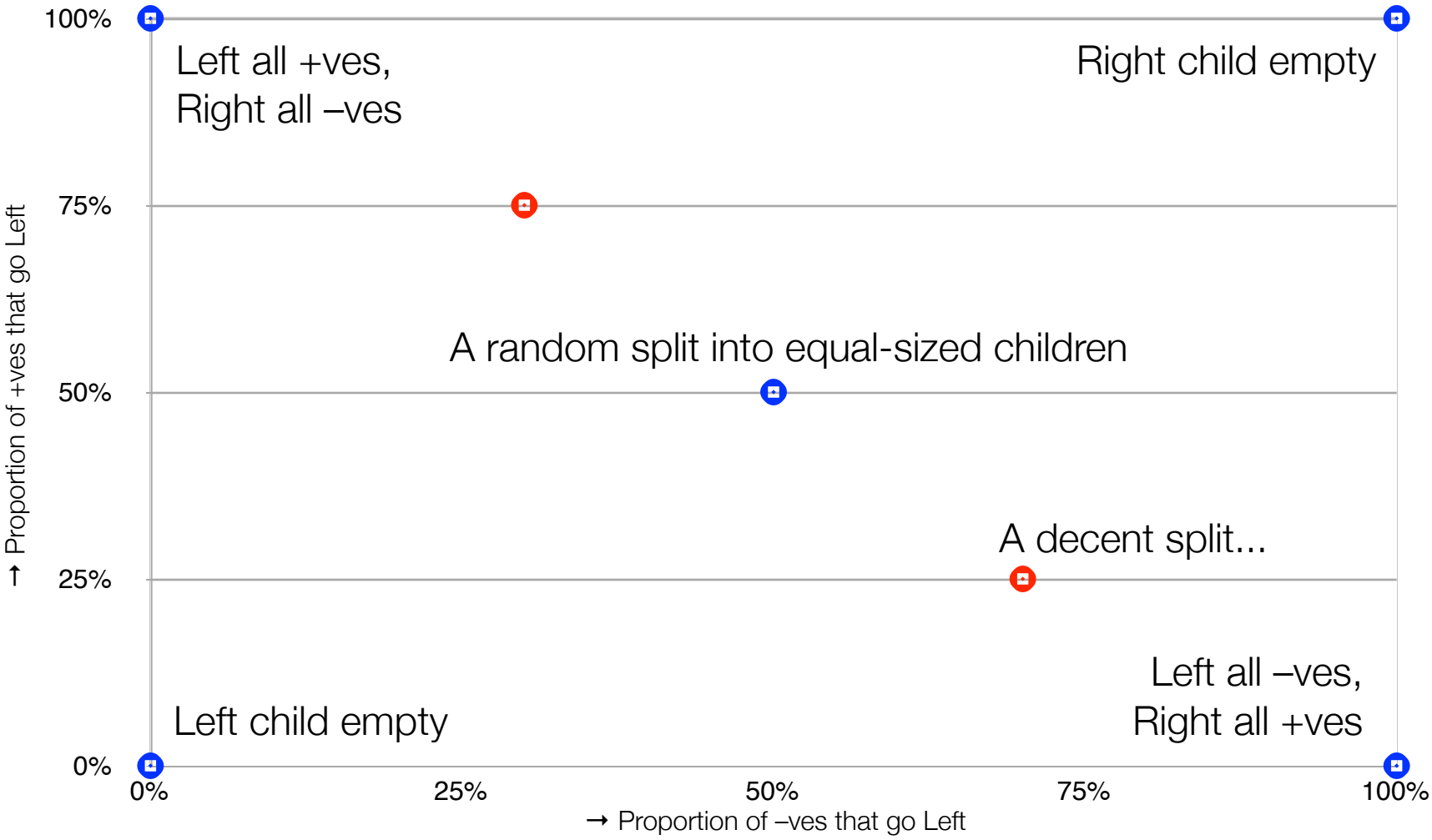
ROC space for splitting criteria



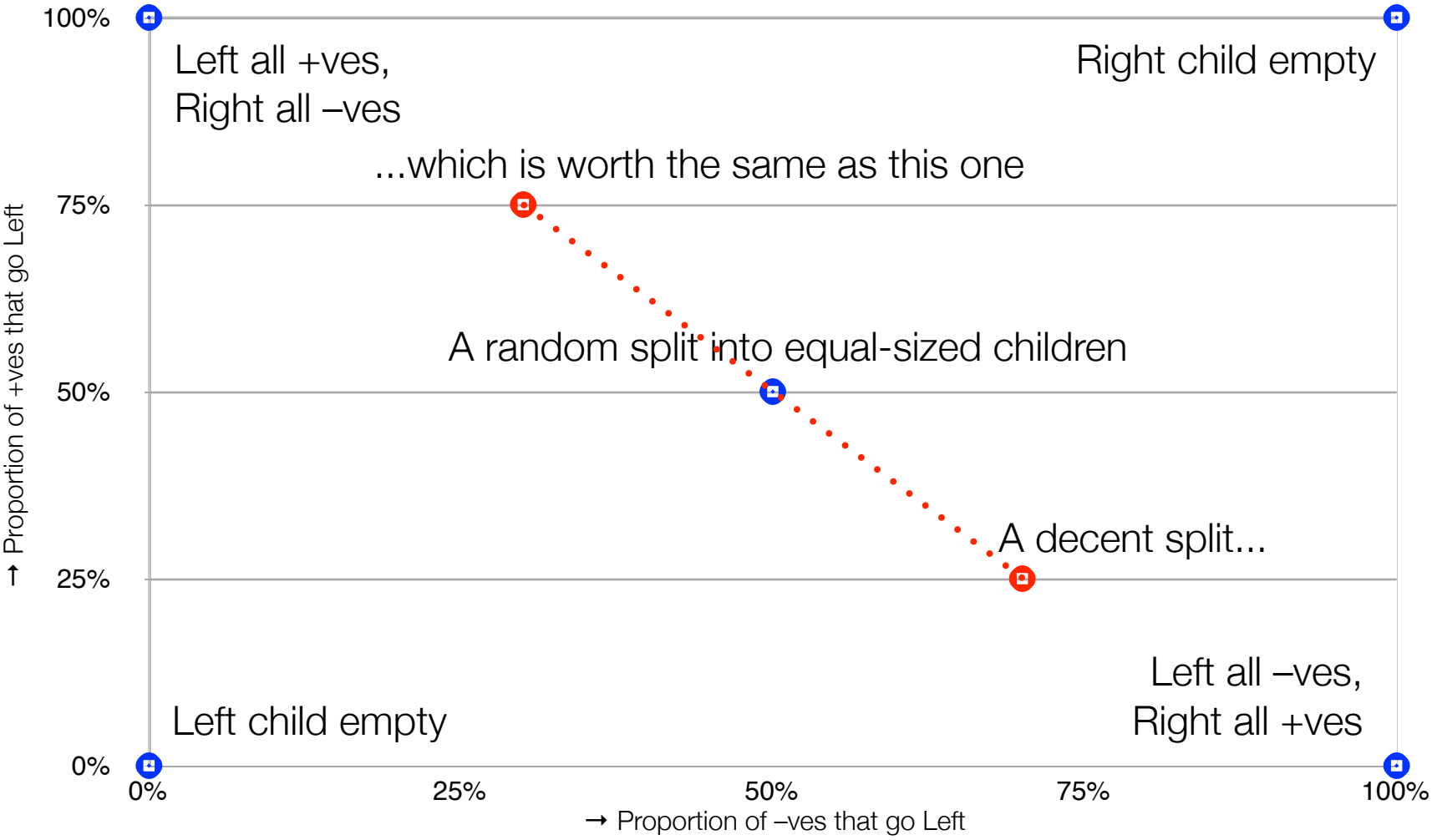
ROC space for splitting criteria



ROC space for splitting criteria



ROC space for splitting criteria



Impurity functions

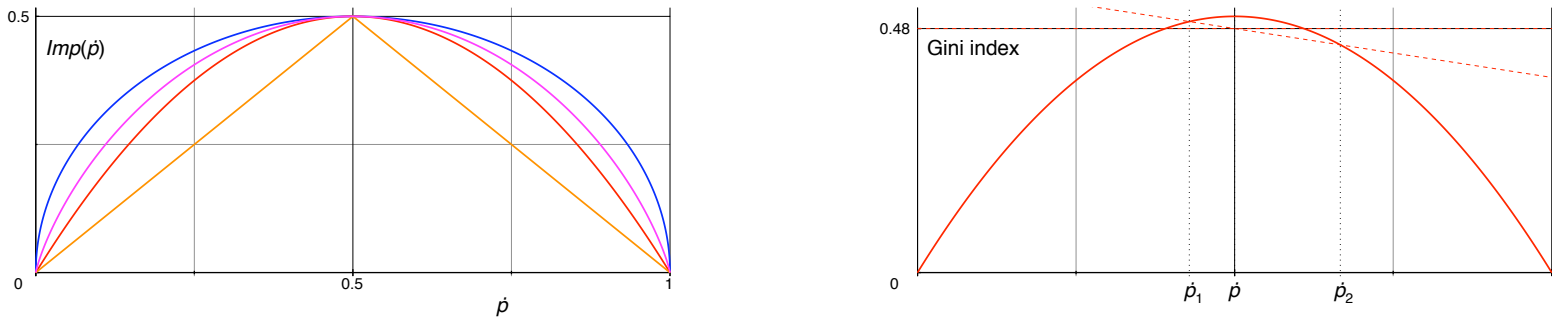


Figure 5.2. **(left)** Impurity functions plotted against the empirical probability of the positive class. From the bottom: the relative size of the minority class, $\min(\dot{p}, 1-\dot{p})$; the Gini index, $2\dot{p}(1-\dot{p})$; entropy, $-\dot{p}\log_2\dot{p} - (1-\dot{p})\log_2(1-\dot{p})$ (divided by 2 so that it reaches its maximum in the same point as the others); and the (rescaled) square root of the Gini index, $\sqrt{\dot{p}(1-\dot{p})}$ – notice that this last function describes a semi-circle. **(right)** Geometric construction to determine the impurity of a split (Teeth = [many, few] from Example 5.1): $\dot{p}$ is the empirical probability of the parent, and $\dot{p}_1$ and $\dot{p}_2$ are the empirical probabilities of the children.

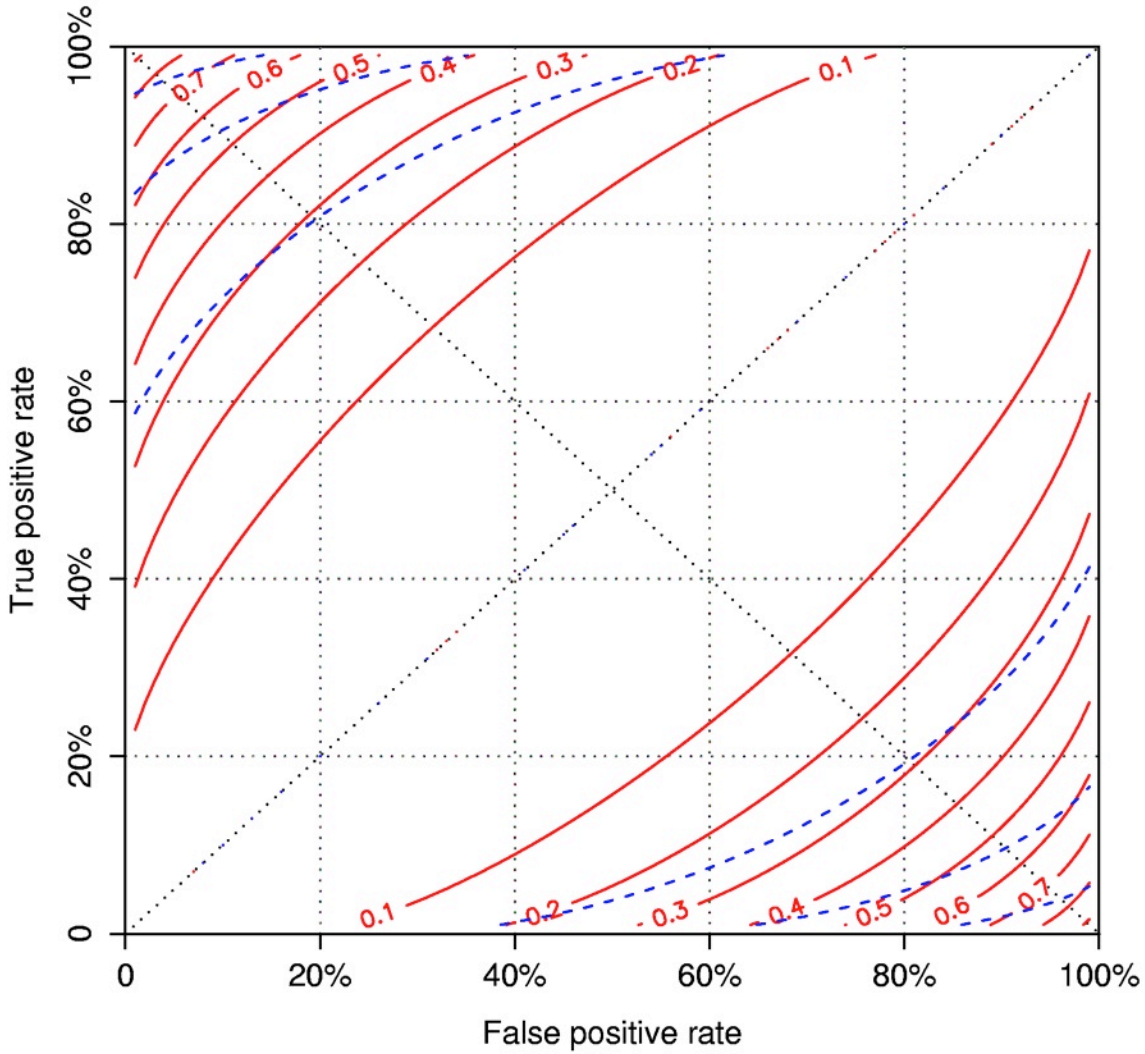
Impurity functions (2)

- relative impurity is defined as weighted impurity of (left) child in proportion to impurity of parent

Impurity	$Imp(p,n)$	Relative impurity
Entropy	$-p\log p - n\log n$	
Gini index	$4pn$	$\frac{(1+c)\cdot tpr\cdot fpr}{tpr + c\cdot fpr}$
DKM	$2\sqrt{pn}$	$\sqrt{tpr\cdot fpr}$

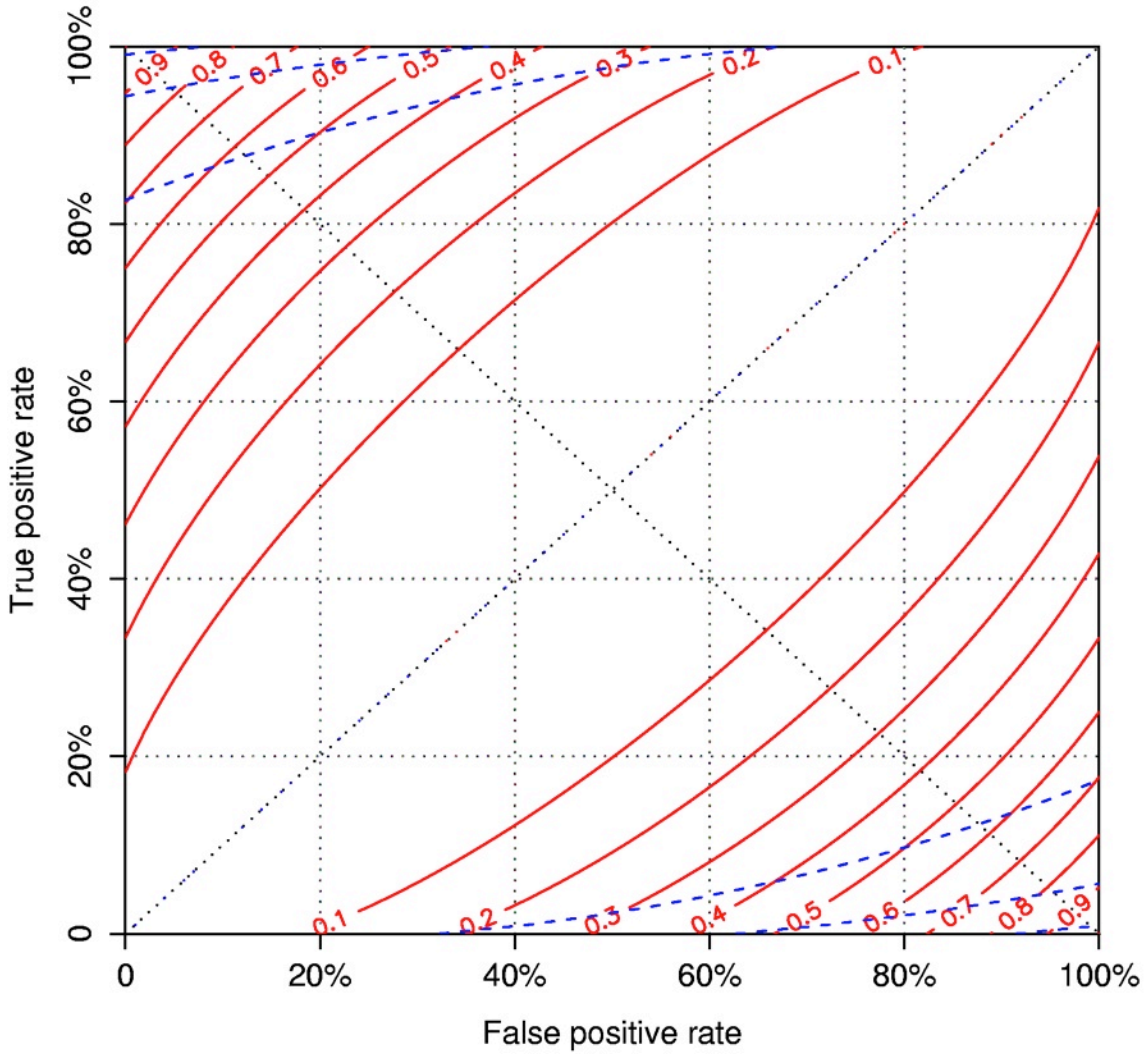
All impurity functions are re-scaled to [0,1]. DKM refers to (Dietterich, Kearns & Mansour, 1996). The skew-insensitivity of DKM-split for binary splits was shown by (Drummond & Holte, 2000).

Information gain isometrics



c = 1,
c = 1/10

Gini-split isometrics



c = 1,
c = 1/10

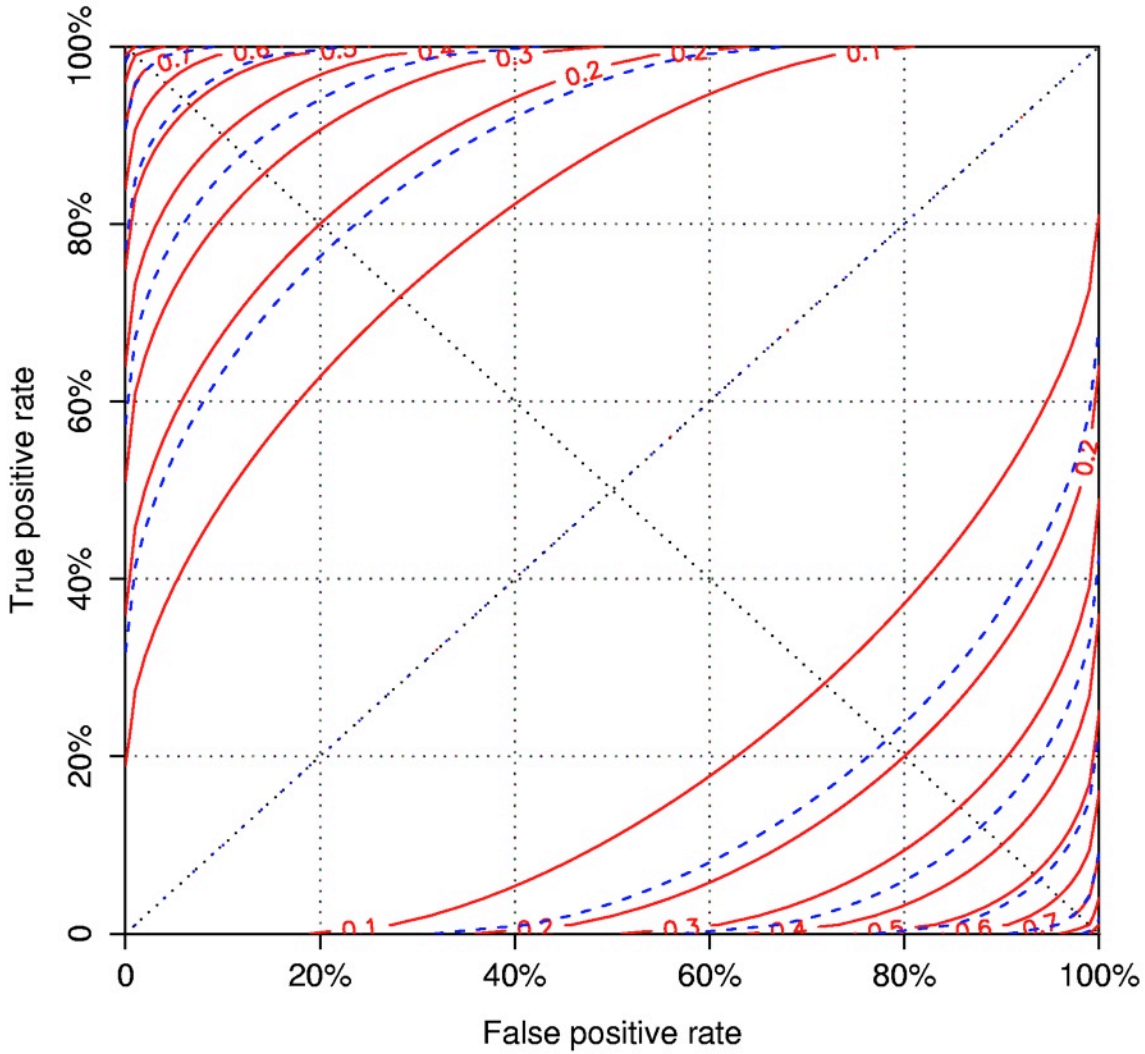
Comments on Gini-split

- More skew-sensitive than information gain
- Equivalent to two-by-two χ^2 normalised by sample size (i.e., Φ^2)
- Strongly skew-insensitive version obtained by setting c=1:

$$GiniROC = 1 - \frac{2\,tpr \cdot fpr}{tpr + fpr}$$

  - complement of the harmonic mean of true and false positive rates

DKM-split isometrics



c = 1,
c = 1/10

Skew-insensitive splitting

- The best splits do well on both classes, even with highly unbalanced data sets
  - so the trees optimise macro-averaged accuracy $(tpr + 1 - fpr)/2$
  - rather than micro-averaged accuracy $pos \cdot tpr + neg \cdot (1 - fpr)$
- Inflating a class does not change split quality
  - bar rounding errors and tie-breaking
- Skew-sensitivity comes into play when pruning a decision tree

Tree learning: Peter's favourite recipe

1. First and foremost, I would concentrate on getting good ranking behaviour, because from a good ranker I can get good classification and probability estimation, but not necessarily the other way round.
2. I would therefore try to use an impurity measure that is distribution-insensitive, such as $\sqrt{\text{Gini}}$; if that isn't available and I can't hack the code, I would resort to oversampling the minority class to achieve a balanced class distribution.
3. I would disable pruning and smooth the probability estimates by means of the Laplace correction (or the m-estimate).
4. Once I know the deployment operation conditions, I would use these to select the best operating point on the ROC curve (i.e., a threshold on the predicted probabilities, or a labelling of the tree).
5. (optional) Finally, I would prune away any subtree whose leaves all have the same label.

ROC-based model manipulation

- ROC analysis allows creation of model variants without re-training
  - (Part I) manipulating ranker thresholds
  - (Part I) Re-labelling decision trees (Ferri et al., 2002)

- Example: Repairing concavities in ROC curves (Flach & Wu, 2003)

ROC-based model manipulation

- ROC analysis allows creation of model variants without re-training
  - (Part I) manipulating ranker thresholds
  - (Part I) Re-labelling decision trees (Ferri et al., 2002)
- Example: Repairing concavities in ROC curves (Flach & Wu, 2003)

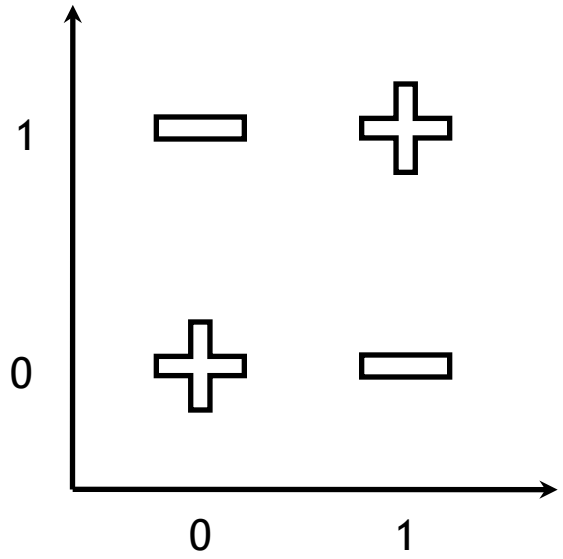
"The article of Repairing Concavities in ROC Curves that will help you to perform all the main maintenance and repair work correctly and efficiently."

http://veganpr0n.com/repairing-concavities-in-roc-curves/

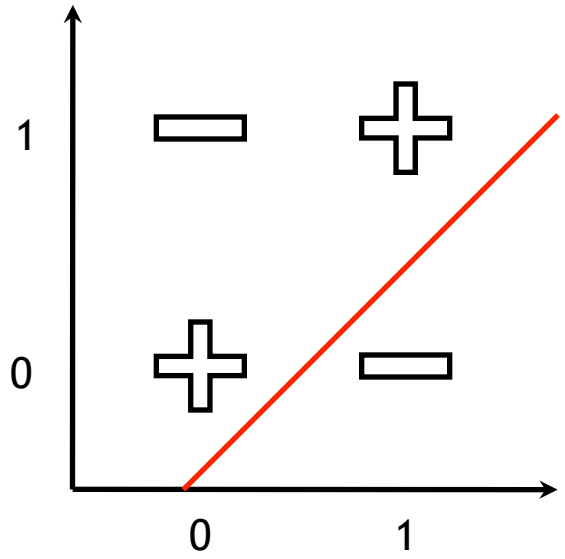
Locally adjusted rankings

- 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 → validation set

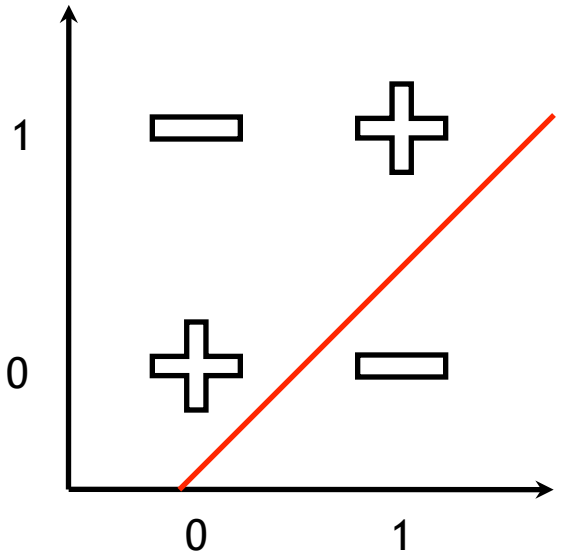
Example: XOR

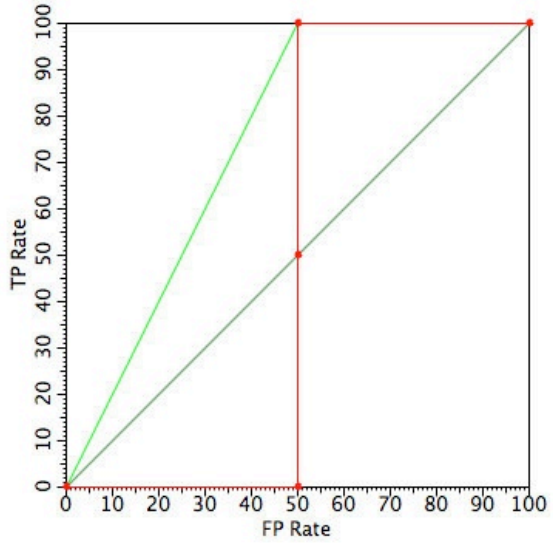


Example: XOR

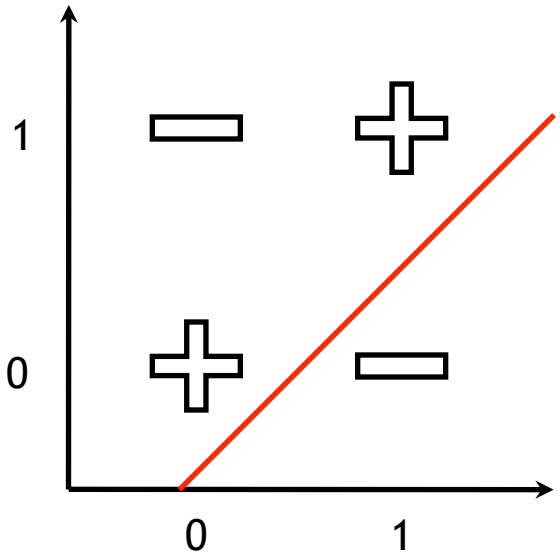


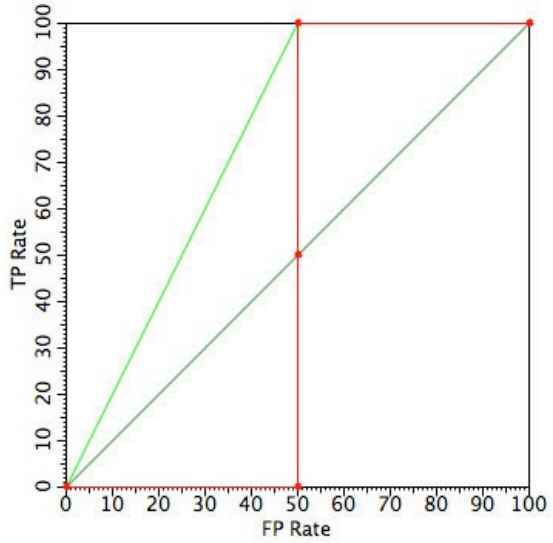
Example: XOR





Example: XOR





above decision boundary?

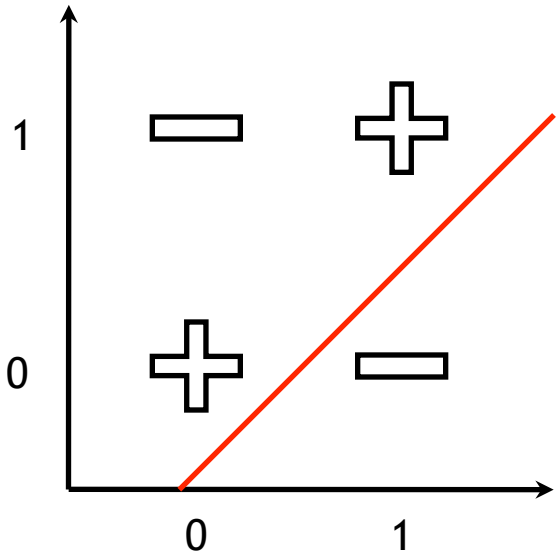
yes

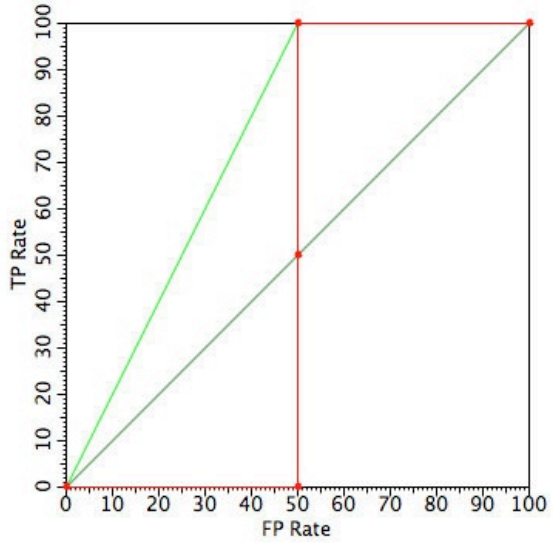
no

use ranking
in 1st segment

use ranking
in 2nd segment

Example: XOR





above decision boundary?

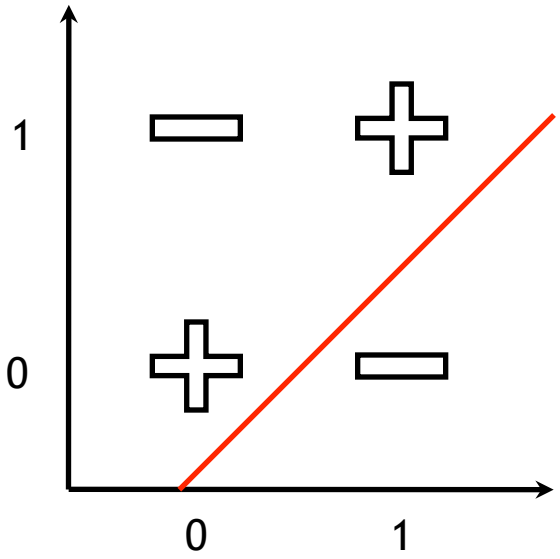
yes

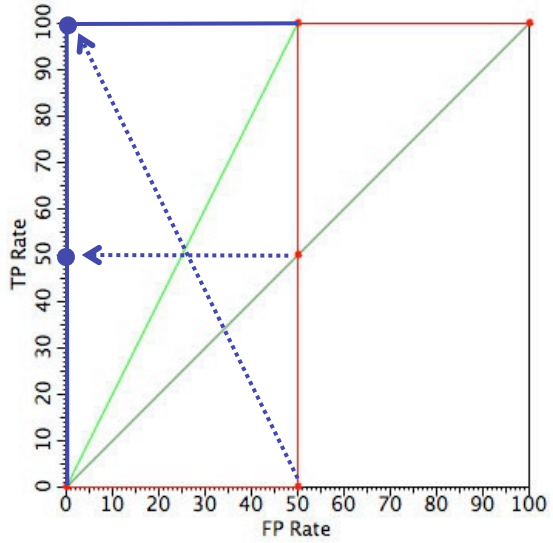
no

tied ~~use~~ ranking in 1st segment

use ranking in 2nd segment

Example: XOR





above decision boundary?

yes

no

invert XXXX use ranking in 1st segment

use ranking in 2nd segment

Algorithm RepairSection

- Given a scoring model M and two thresholds T1>T2, construct a scoring model M' predicting scores as follows:

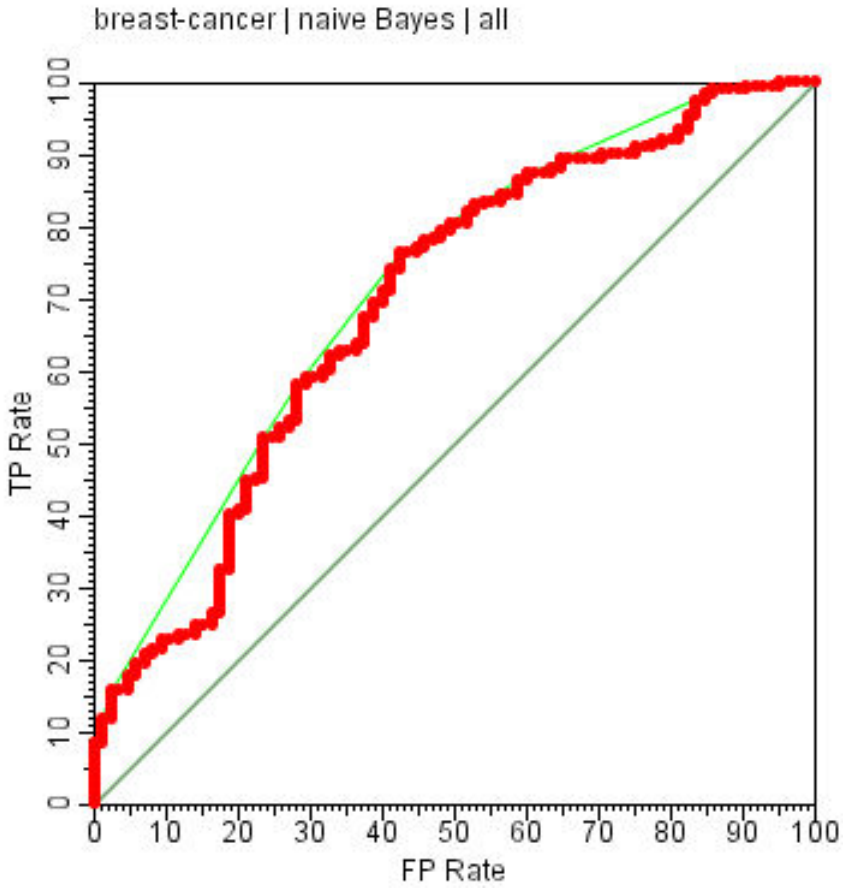
- Let S(x) be the score predicted by M for instance x:
  - If X>T1, then predict S(x);
  - If X<T2, then predict S(x);
  - Otherwise, predict T1+T2–S(x).

Experimental design

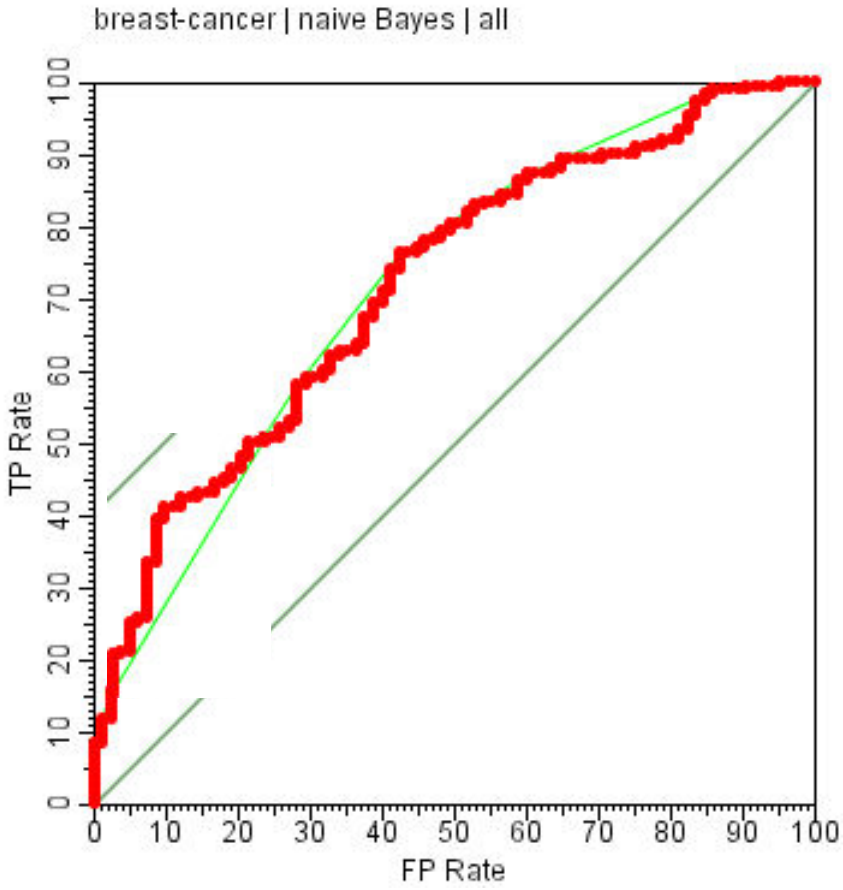
1. Train a naive Bayes or decision tree model M on the training data; construct a ROC curve C and its convex hull H on the training data.
2. Find adjacent points on H such that in this interval the area between C and H is largest. Let T_1 and T_2 be the corresponding score thresholds.
3. Produce a new probabilistic model M' by calling Repair-Section(T_1, T_2).
4. Evaluate M and M' on the validation set, construct their ROC curves and calculate their AUCs. If AUC(M') $\leq$ AUC(M) then go to 6.
5. Evaluate M and M' on the test set, construct their ROC curves and calculate their AUCs.
6. Go to 1. until each fold has been used as a test set.

10-fold cross-validation: use 8 folds for training, 1 fold for validation and 1 fold for testing

Example


breast-cancer | naive Bayes | all
TP Rate
FP Rate
0
10
20
30
40
50
60
70
80
90
100

Example


breast-cancer | naive Bayes | all
TP Rate
FP Rate
0
10
20
30
40
50
60
70
80
90
100

Summary of experimental results

- We get small but significant improvements in AUC using decision trees and naive Bayes as base learners (in about half of the data sets)

- What didn't work well:
  - Not using a validation set
  - Repairing all concavities, not just the largest one
  - Using two validation folds with decision trees

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(¬c|c) for each class c∈C
    → |C| dimensions
    - ROC space spanned by either tpr for each class or fpr for each class
  - 1-vs-1: different misclassification costs C(ci|cj) for each pair of classes ci≠cj —> |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

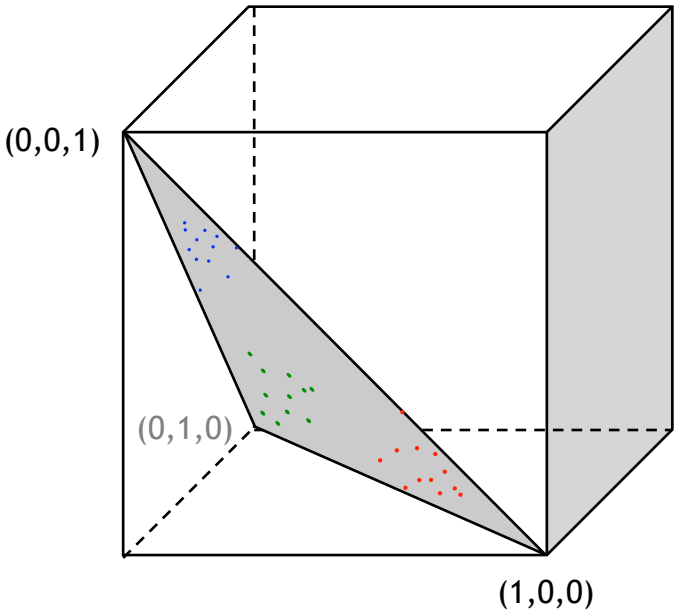
- How to manipulate scores f(x,c) in order to obtain different ROC points?
  - depends on the cost model

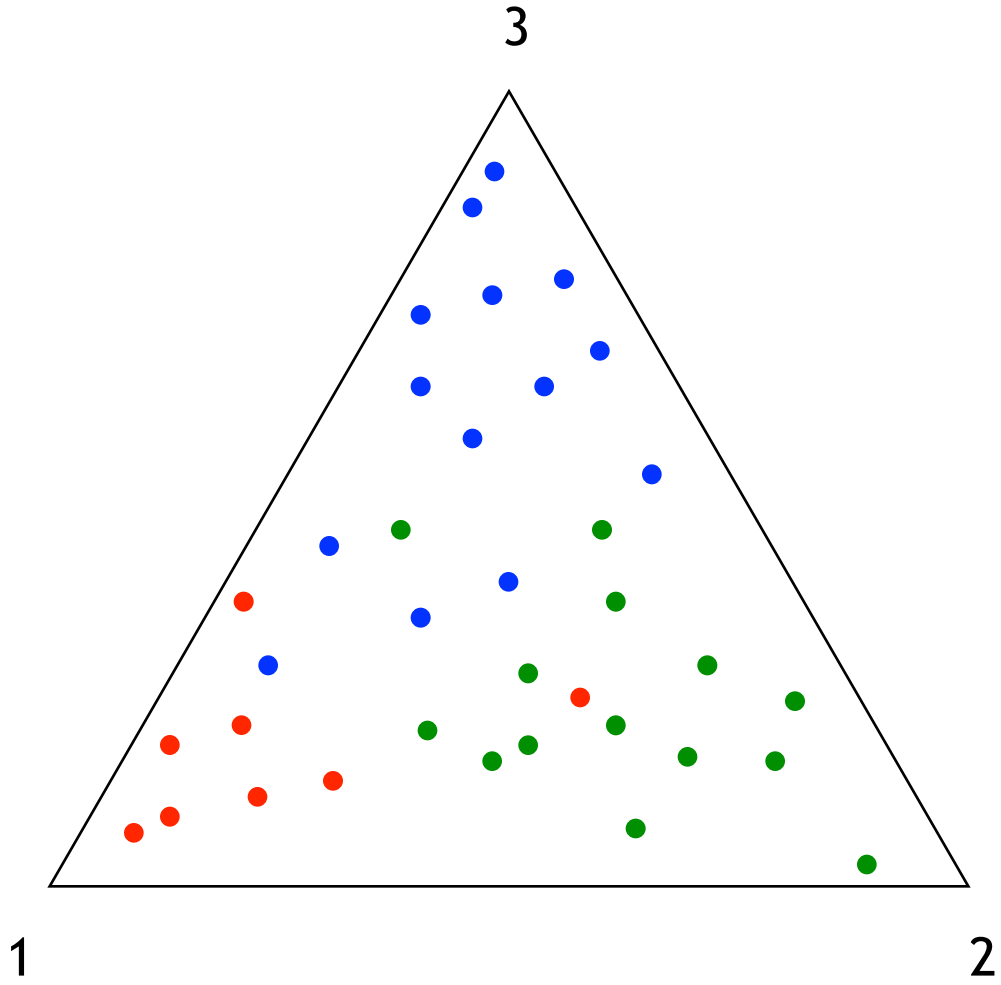
- How to search these ROC points to find optimum?
  - exhaustive search probably infeasible, so needs to be approximated

A simple 1-vs-rest approach

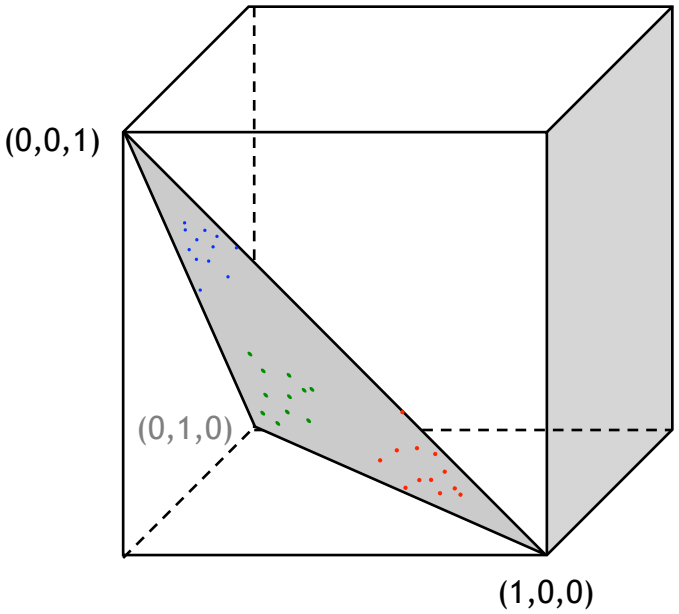
- From thresholds to weights:
  - predict argmaxc w_c f(x,c)
  - NB. two-class thresholds are a special case:
    - w_+ f(x,+) > w_- f(x,–) ⇔ f(x,+)/f(x,–) > w_-/w_+
- Setting the weights (Lachiche & Flach, 2003)
  - Assume an ordering on classes and set the weights in a greedy fashion
    - Set $w_1 = 1$
    - 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

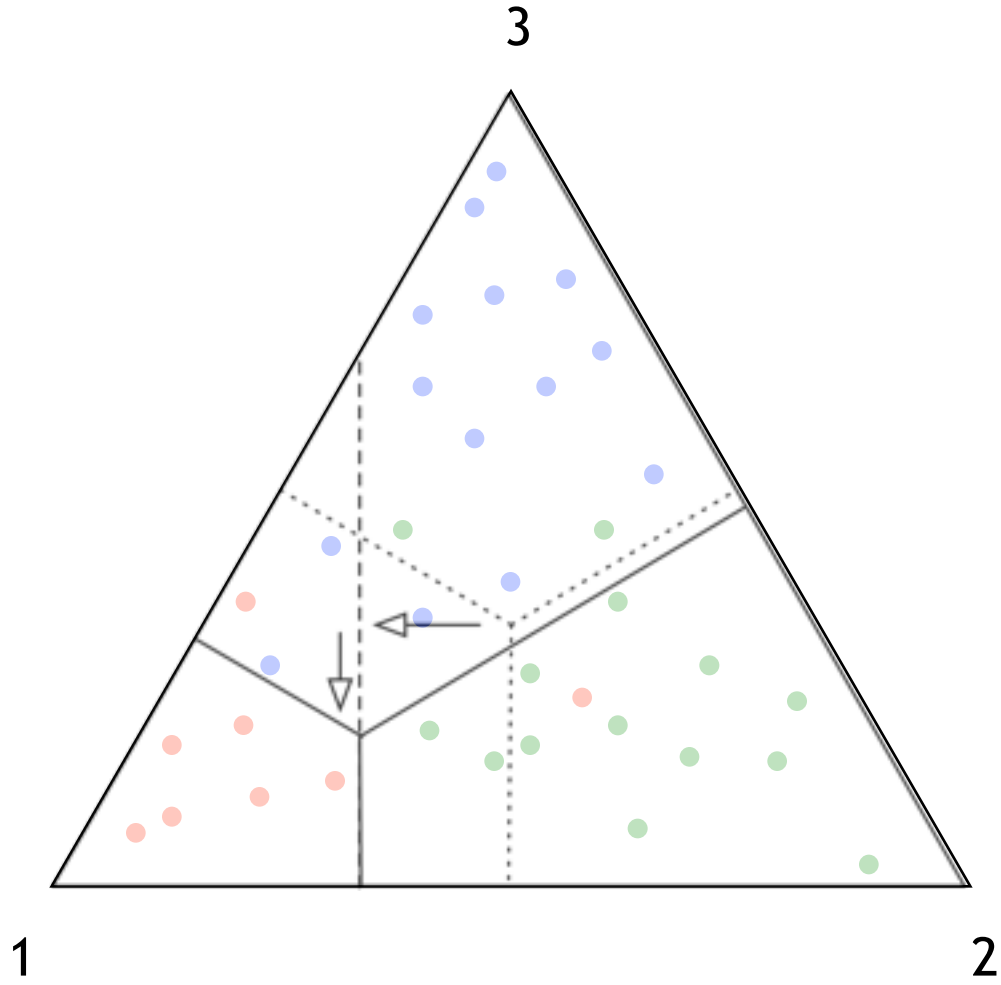
Example: 3 classes





Example: 3 classes





Discussion

- Strong experimental results
  - 13 significant wins (95%), 22 draws, 2 losses on UCI data
- Sensitive to the ordering of classes
  - largest classes first is best
- No guarantee to find a global (or even a local) optimum
  - lots of scope for improvement, e.g. stochastic search

Part II: concluding remarks

- Isometric plots visualise the behaviour of machine learning metrics
  - equivalences, skew-sensitivity, skew-insensitive versions
- One model can be many models
  - ROC analysis can be used to obtain alternative labellings of trees, adjust rankings, etc.
- Multi-class ROC

Part III: Comparing machine learning metrics

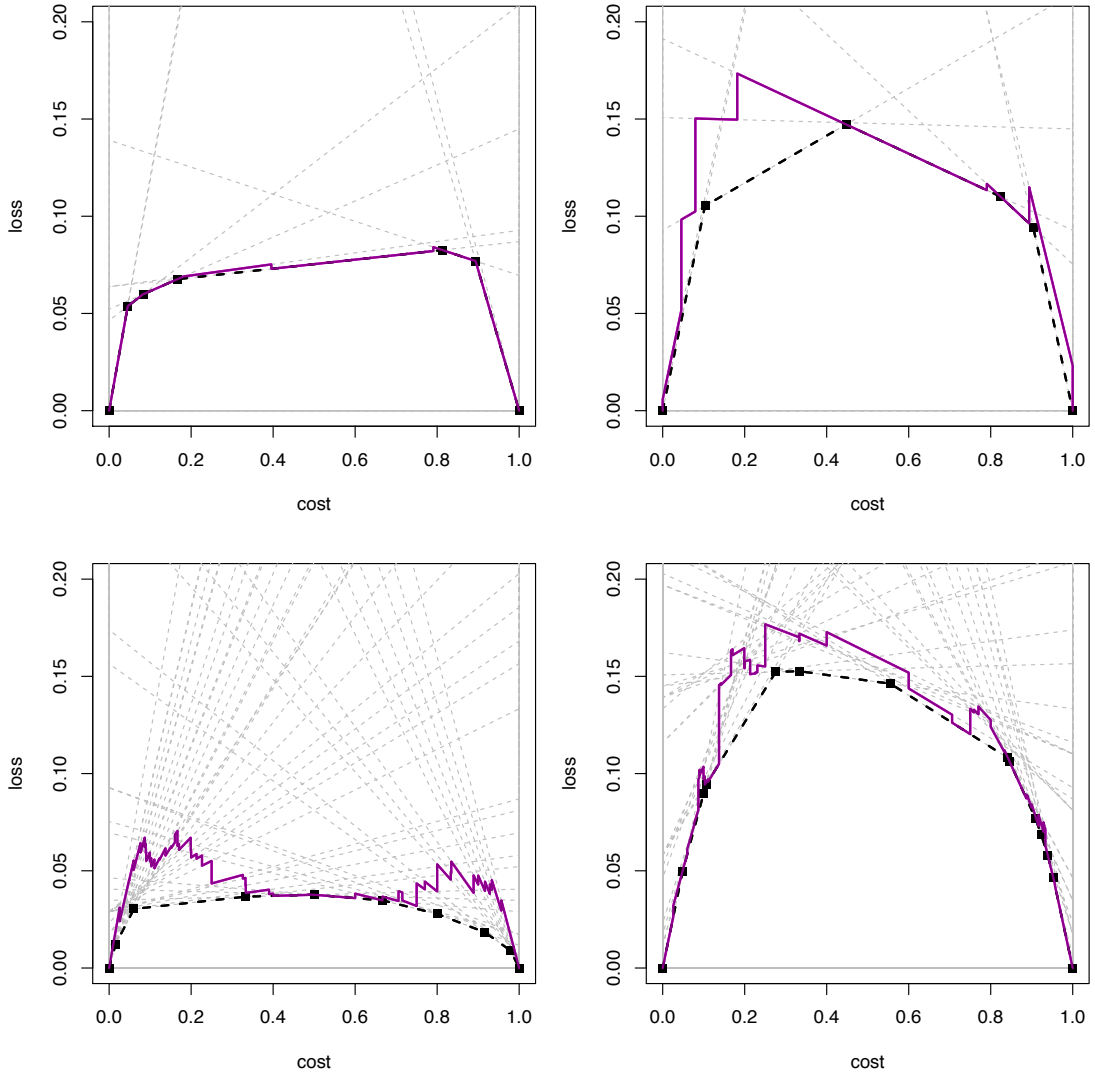
- This is based on recent work with Jose Hernandez-Orallo and Cesar Ferri.

- The main question is: what do metrics such as AUC — which do not directly measure classification performance — tell us about classification?

Quiz: Decision tresholds

- Suppose you train a two-class naive Bayes model on a training set with balanced classes
  - the model uses the default decision threshold (0.5 on estimated posterior probabilities) and achieves a certain performance, measured as accuracy, MAE, Brier score and AUC.
- You are now given a new data set; it is unlabelled, but you are told the proportion of positives $\pi \neq 0.5$. You are asked to classify this data set with your naive Bayes model. Which threshold do you use?
  1. the threshold is kept at 0.5.
  2. the threshold is set uniformly randomly.
  3. the threshold is set to $1-\pi$.
- What would the expected 0/1 loss be in each case, assuming a uniform distribution over π?

Score-driven threshold selection in cost space



- Training set (left), test set (right); pruned tree (top), unpruned tree (bottom)
- Depending on the operating condition (x-axis) we choose a different operating point and hence a different cost line.
- These curves are called Brier curves as their area is the Brier score.

Brier score decomposition

- The Brier score is the mean squared deviation from the ideal (rather than true) probabilities:

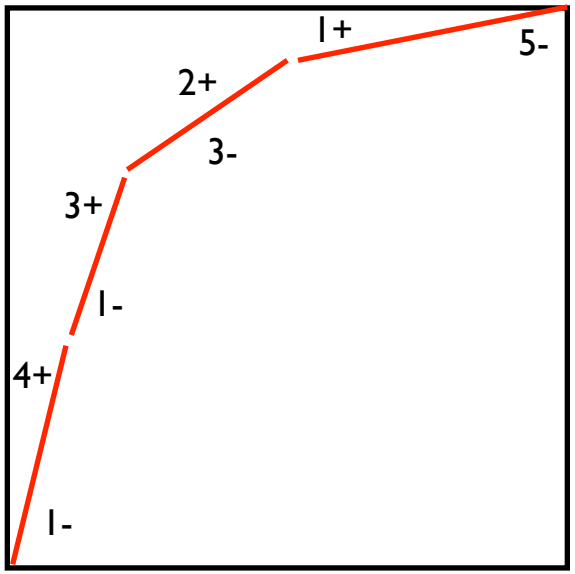
$$\mathrm{BS} = \frac{1}{|D|}\left(\sum_{i \in \oplus} \hat{p}_i^2 + \sum_{j \in \ominus} (1-\hat{p}_j)^2\right)$$

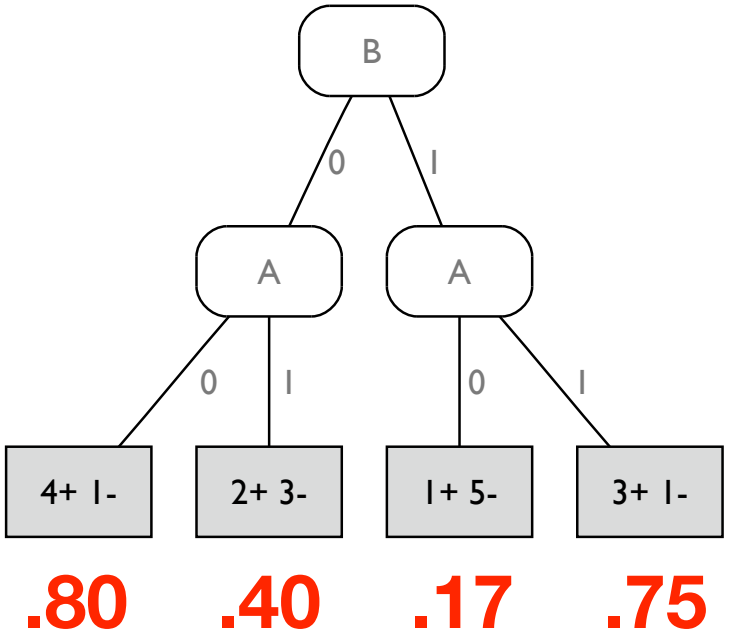
- Over the segments in the ROC curve, this can be decomposed into calibration loss and refinement loss:

$$\mathrm{BS} = \frac{1}{|D|}\sum_k n_k(\hat{p}_k - \dot{p}_k)^2 + \frac{1}{|D|}\sum_k n_k \dot{p}_k(1-\dot{p}_k)$$

$$n_k = n_k^{\oplus} + n_k^{\ominus}, \dot{p}_k = n_k^{\oplus}/n_k$$

Brier score example

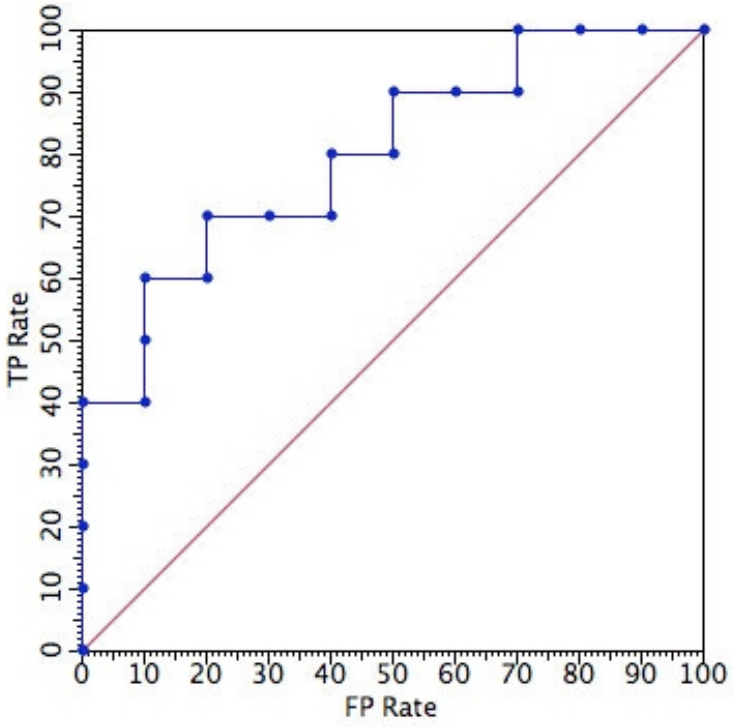




Brier score = $(4*.2^2 + 2*.6^2 + .83^2 + 3*.25^2 + .8^2 + 3*.4^2 + 5*.17^2 + .75^2)/10 = 0.358$.
Zero calibration loss as all predicted probabilities equal empirical probabilities.
Refinement loss = $(5*.8*.2 + 4*.4*.6 + 5*.17*.83 + 6*.75*.25)/10 = 0.358$.

Refinement loss quantifies tied ranking

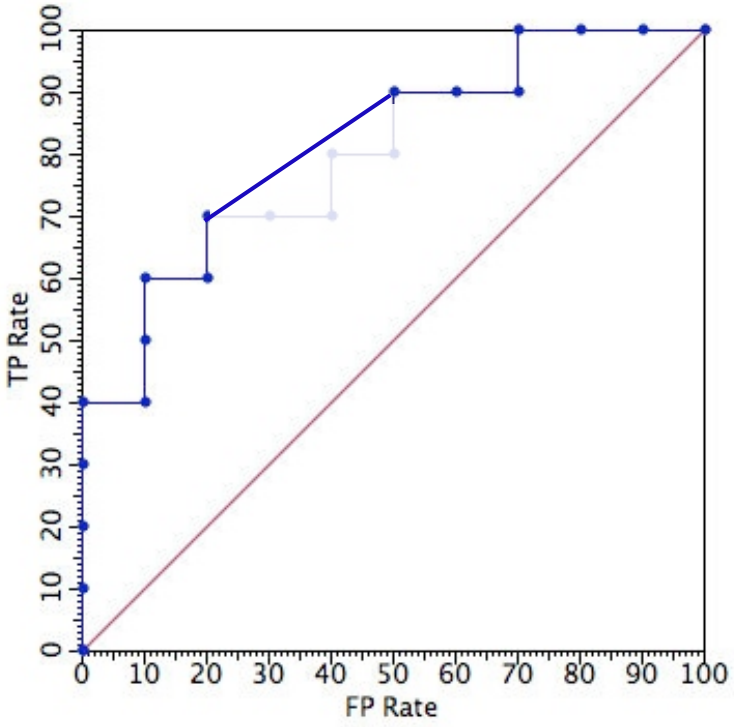
+ + + + - + + - + - - + - + - - + - - -



Zero refinement loss

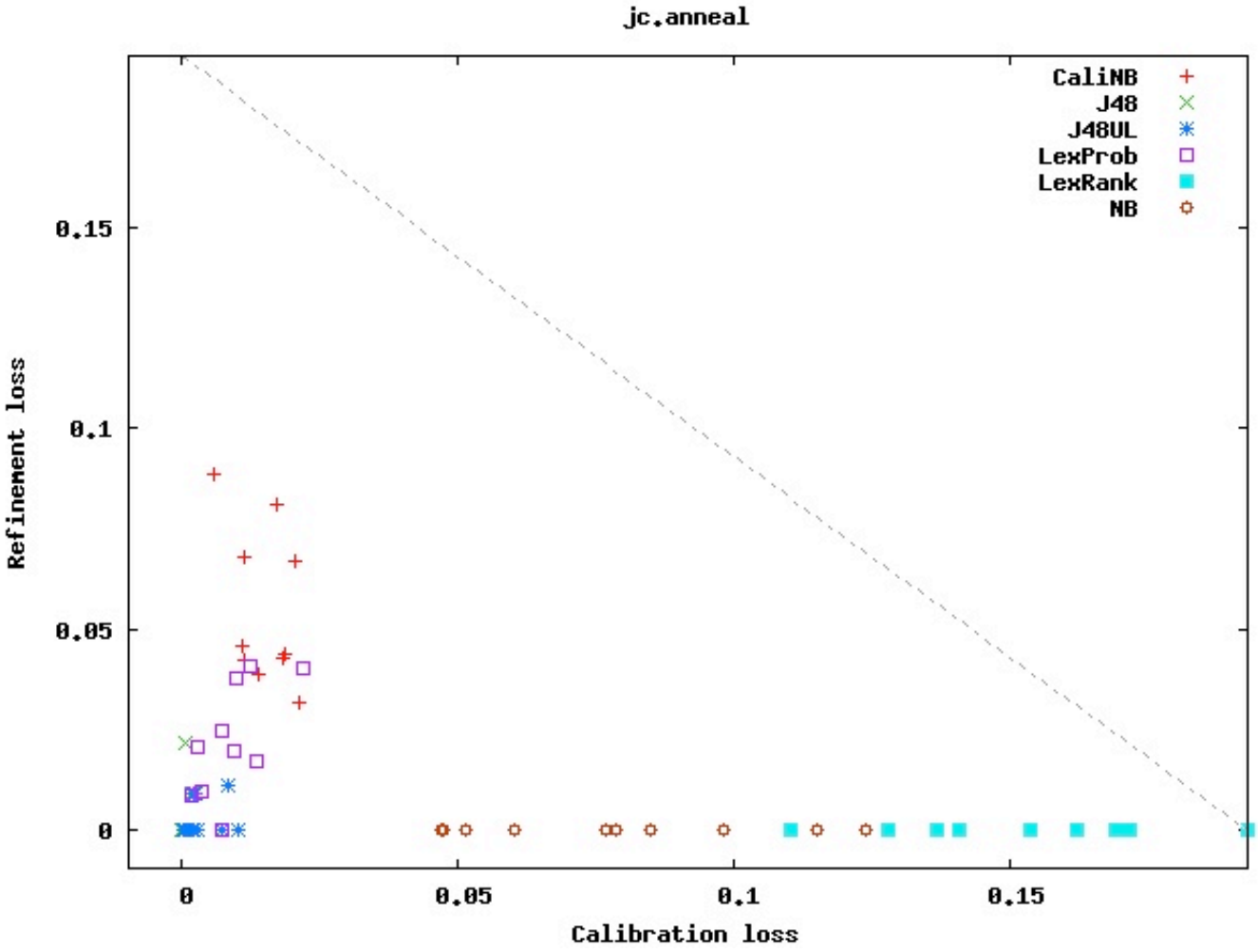
+ + + + - + + - +

- - + - -

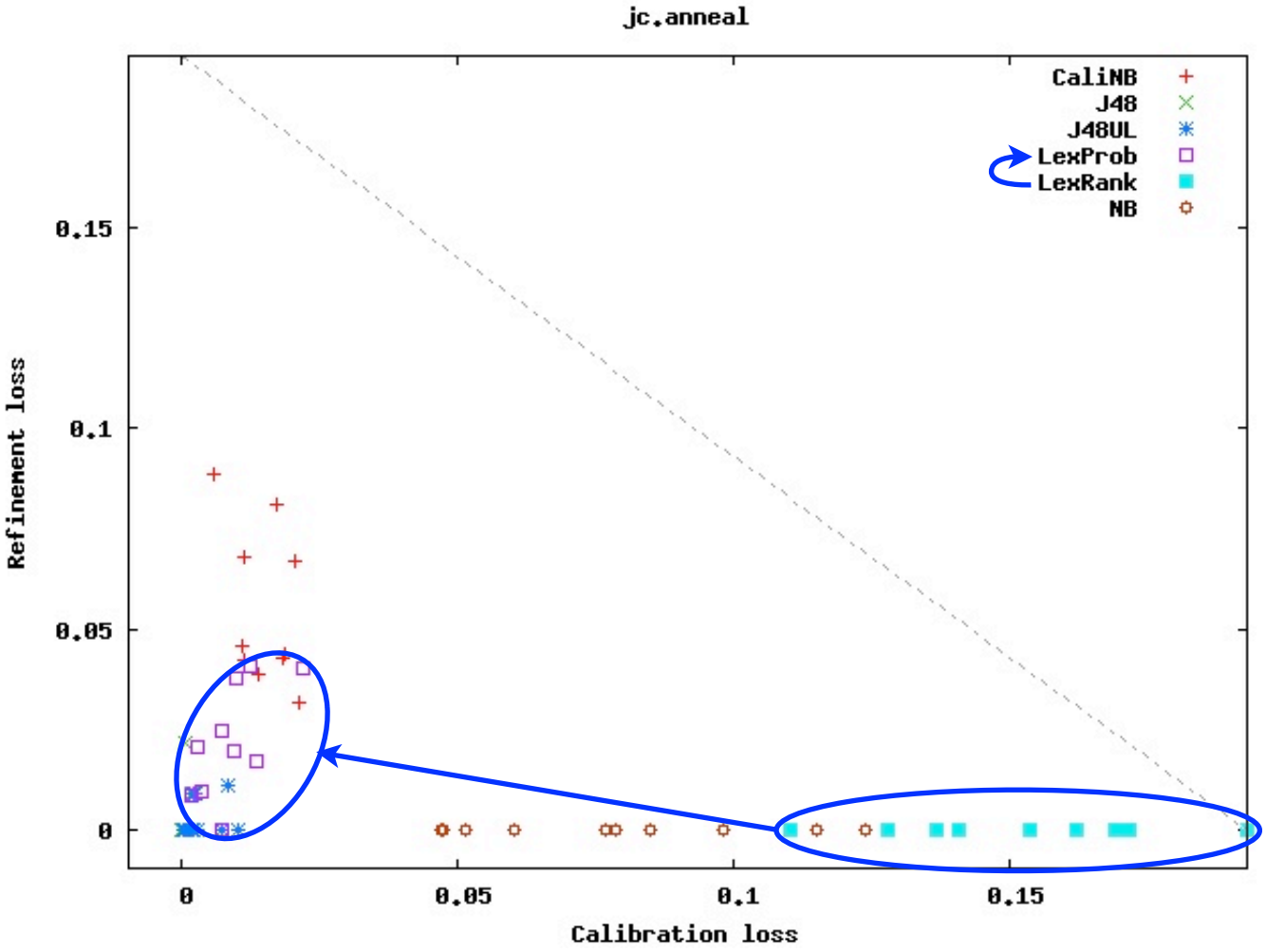


5*.4*.6/10 = 0.12 refinement loss

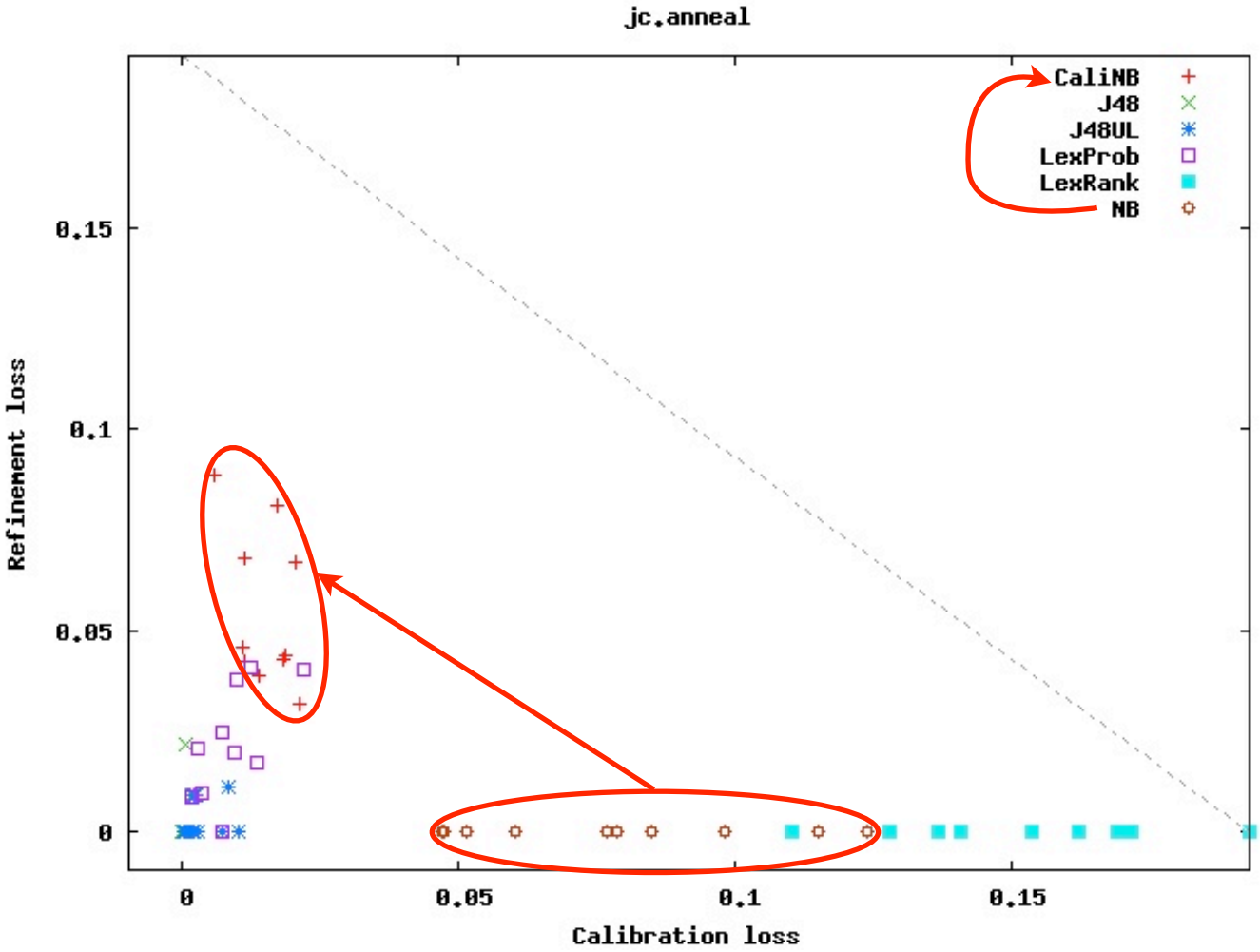
Refinement vs. calibration plot



Refinement vs. calibration plot



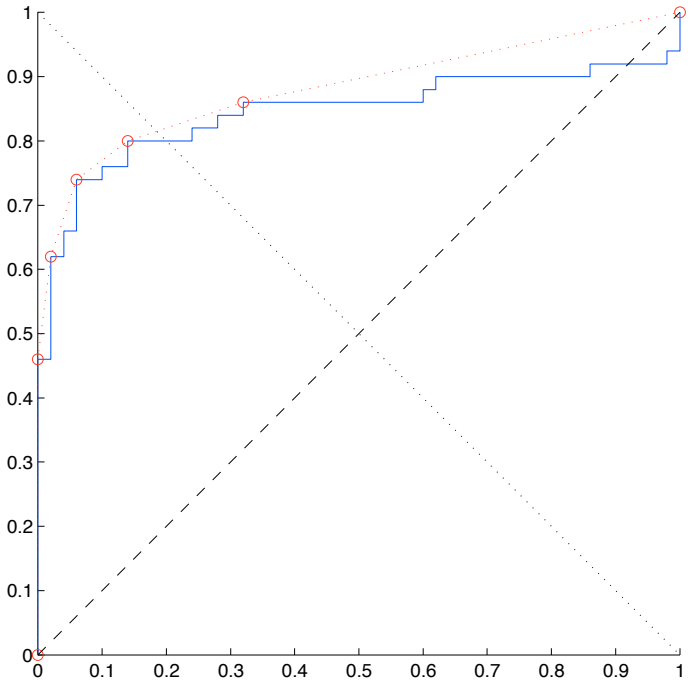
Refinement vs. calibration plot



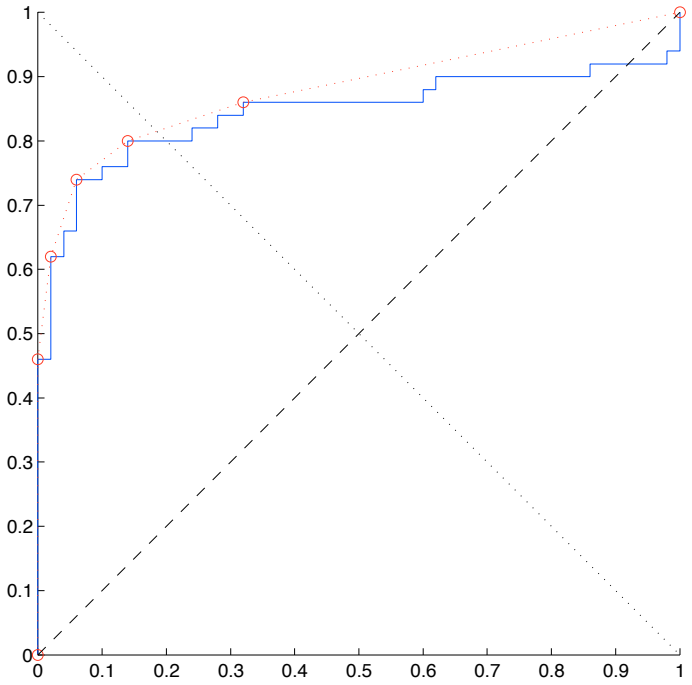
Connecting AUC to expected loss

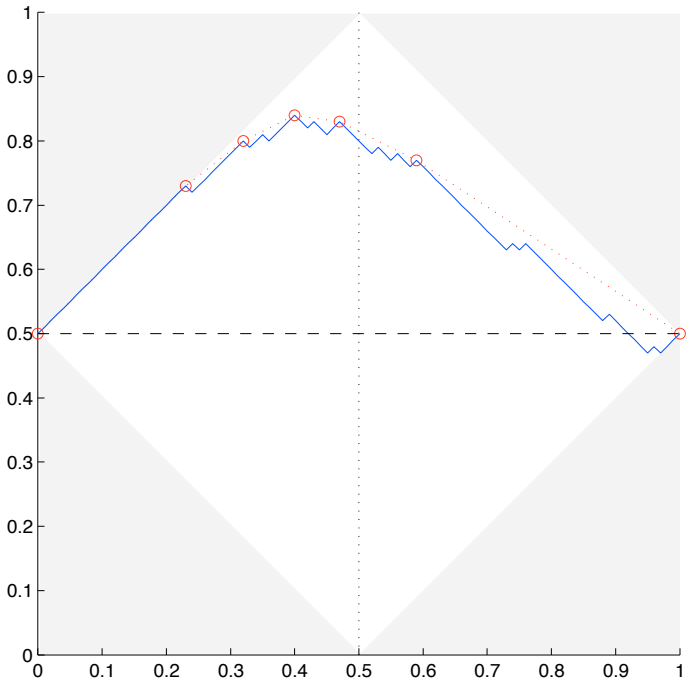
- We saw that setting the decision threshold to 1–π for proportion of positives π allows us to connect the expected 0/1 loss over uniform π to the Brier score. Can we do something similar for AUC?
- David Hand (MLj 2009) established a connection that however depended on the score distribution of the model. He concluded that AUC cannot measure classification performance in a coherent way, and proposed the H-measure as an alternative.
- In response,
  - Flach, Hernandez-Orallo and Ferri (ICML 2011) showed that AUC could be connected to expected loss if non-optimal thresholds were taken into account. They also showed that the H-measure is a variant on the area under the optimal (lower-envelope) cost curve.
  - Hernandez-Orallo, Flach and Ferri (JMLR 2012) gave an alternative connection between AUC and 0/1-loss. They also showed that for optimal thresholds the expected loss is not related to the **area** under the ROC curve but rather to its **shape** through the refinement loss.

From ROC curve to ROL curve

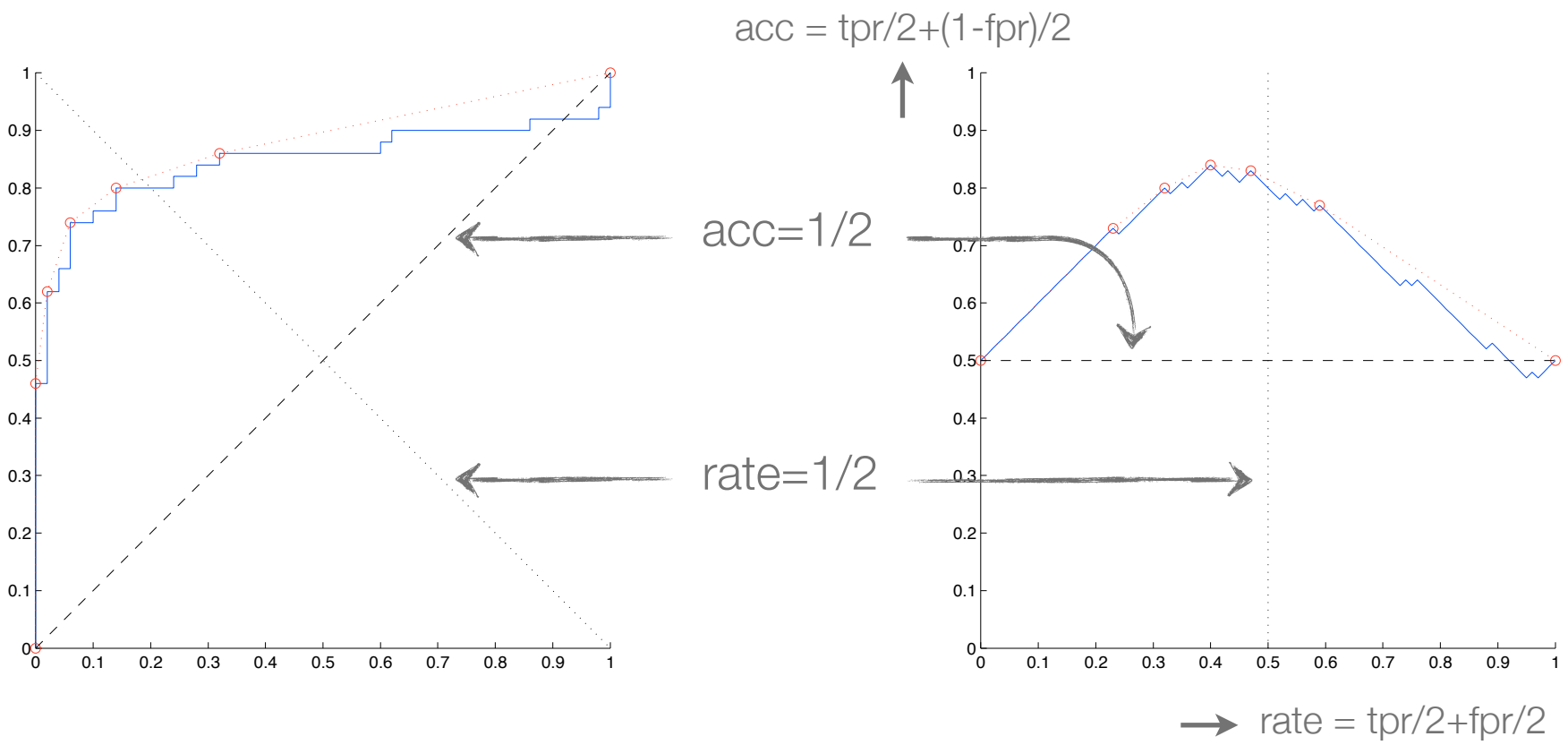


From ROC curve to ROL curve

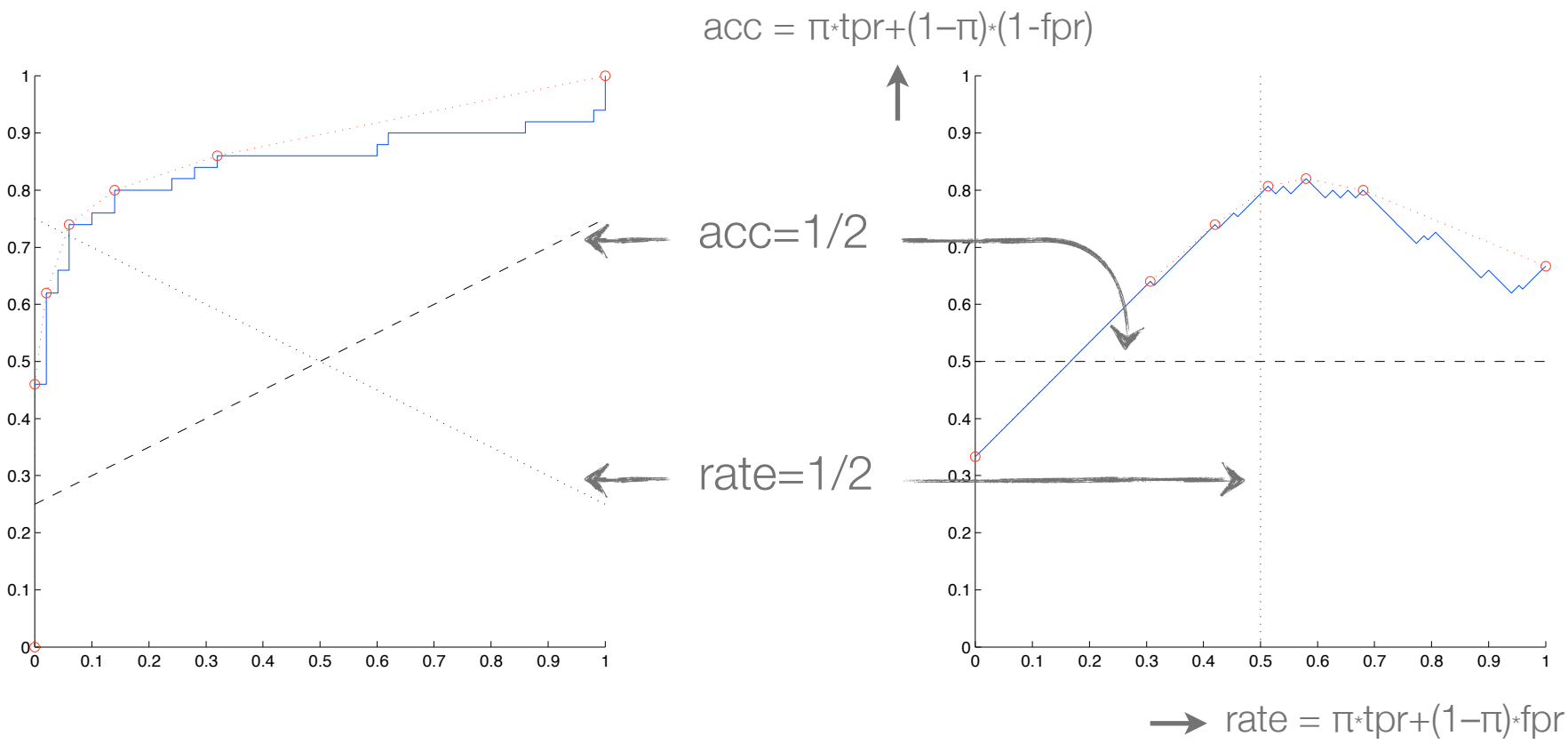




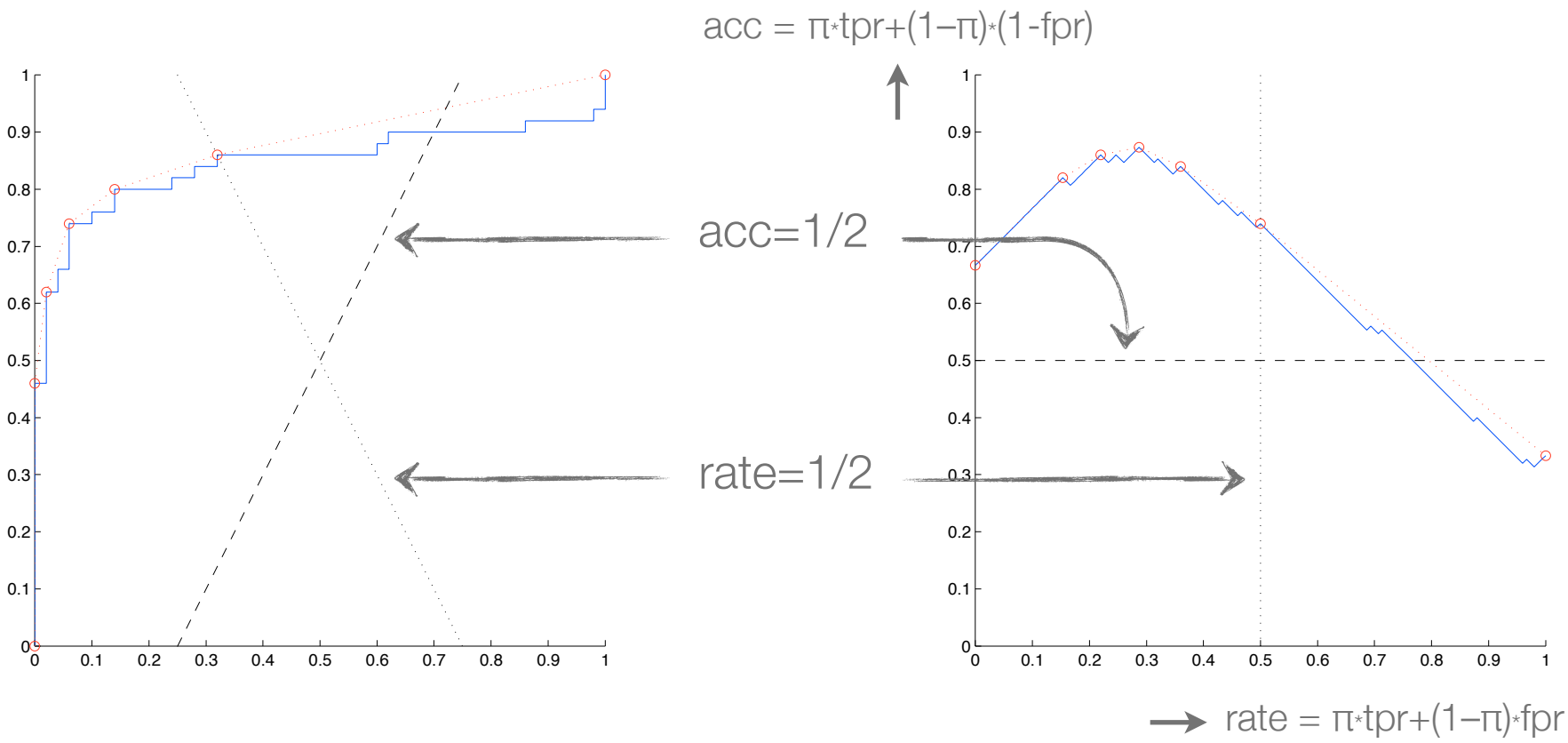
From ROC curve to ROL curve (π = 1/2)



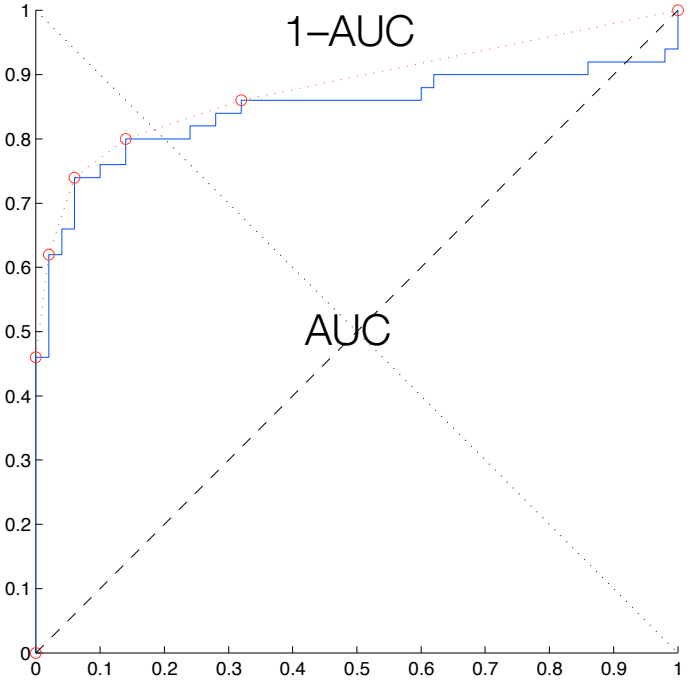
From ROC curve to ROL curve (π > 1/2)

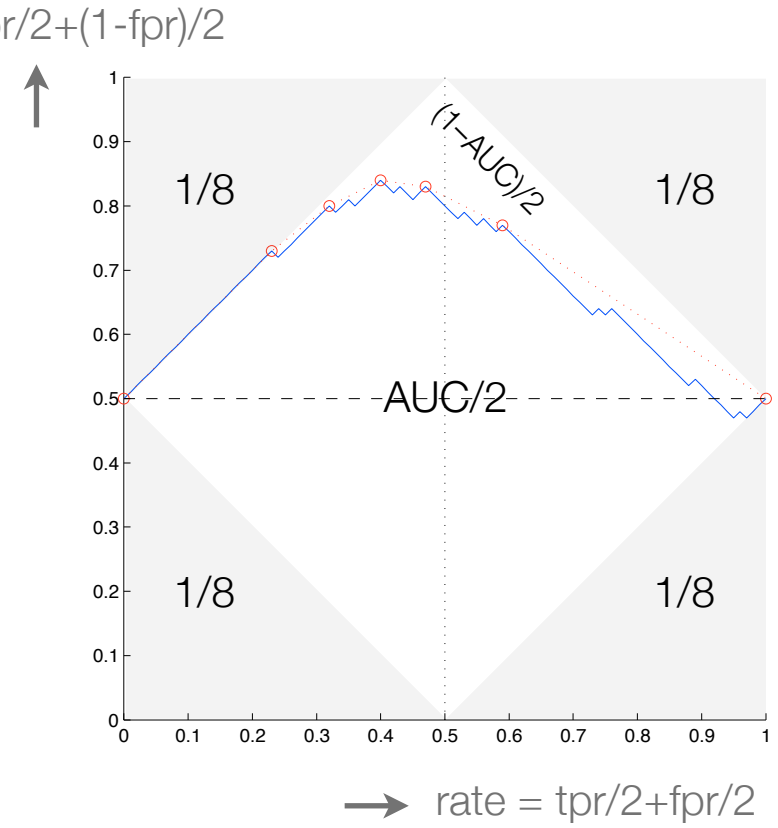


From ROC curve to ROL curve (π < 1/2)



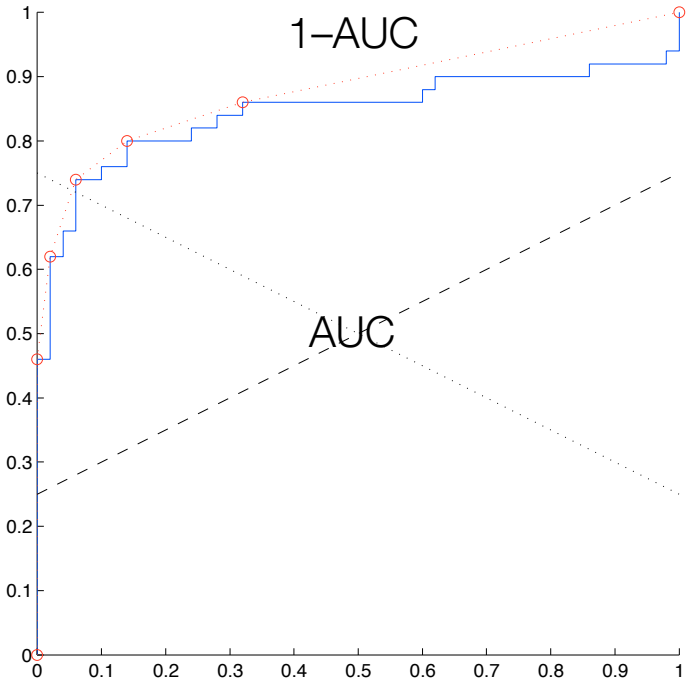
AUC and expected loss (π = 1/2)



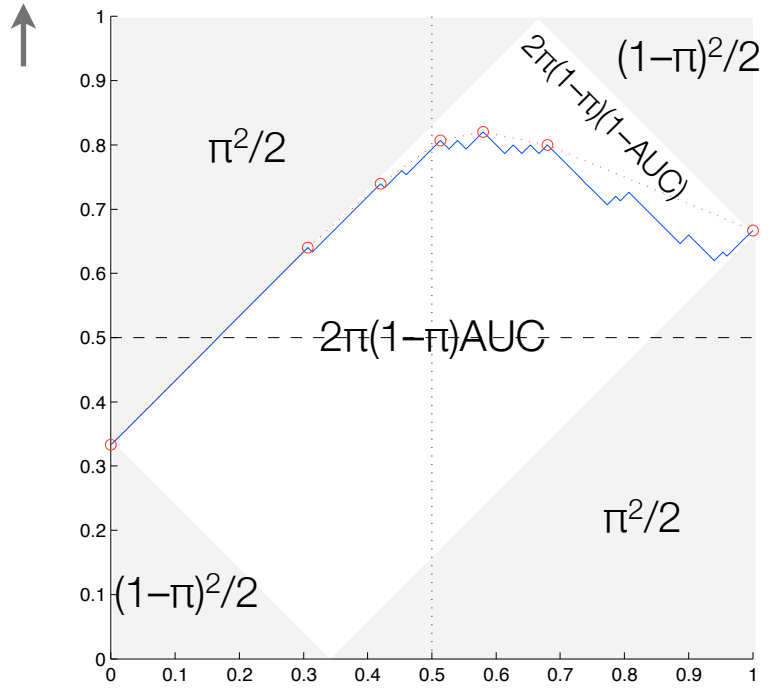


The expected loss for uniform rate is (1-AUC)/2+1/4 = (1-2AUC)/4+1/2.

AUC and expected loss (general case)



acc = π·tpr+(1–π)·(1-fpr)



rate = π·tpr+(1–π)·fpr

Expected loss for uniform rate is
$2\pi(1-\pi)(1\text{-AUC})+\pi^2/2+(1-\pi)^2/2 = \pi(1-\pi)(1\text{-}2\text{AUC})+1/2$.

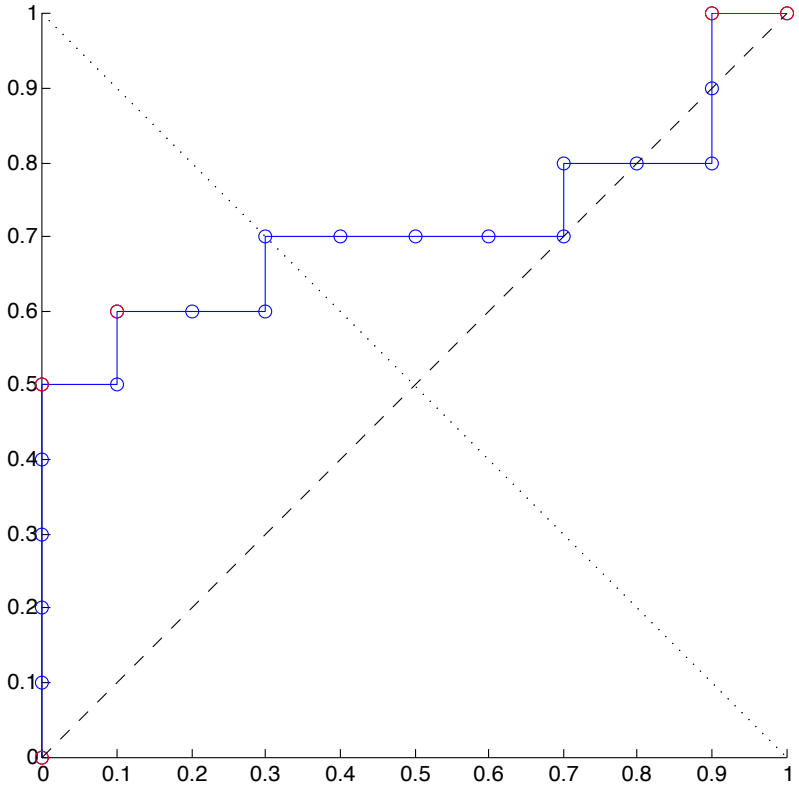
AUC as a classification performance metric

- AUC is a measure of ***ranking*** performance: it estimates the probability that a uniformly randomly selected positive and a uniformly randomly selected negative are ranked correctly.

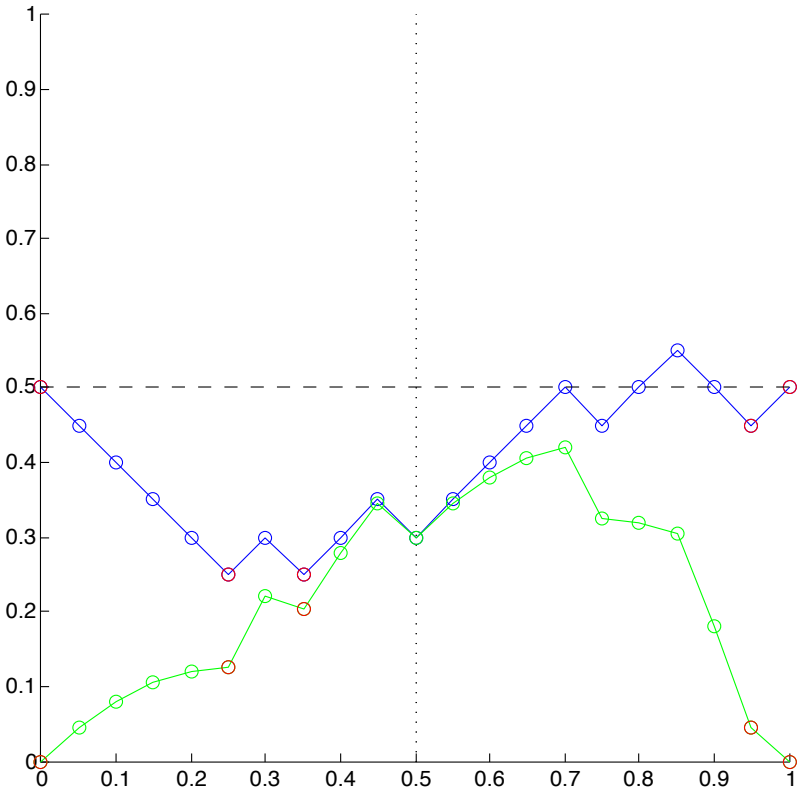
- The ROL curve demonstrates that it is also a measure of ***classification*** performance: the expected loss for a uniformly randomly chosen predicted positive rate is $\pi(1-\pi)(1-2AUC)+1/2$.

- Setting the rate equal to π decreases the expected loss with 1/6 to $\pi(1-\pi)(1-2AUC)+1/3$.
  - also known as the precision/recall break-even point.

Rate-driven loss example



ROC curve



Rate-uniform cost curve (blue);
rate-driven cost curve (green)

Discussion

- If we know the operating condition (here: proportion of positives π) it is always better to take it into account in setting the decision threshold:
  - for score-based thresholds this reduces the expected loss from absolute error to squared error (Brier score).
  - for rate-based thresholds this reduces the expected loss with 1/6.
- One intuition is that knowing the majority class gives us an advantage.
- However, if we misjudge π the resulting performance may be worse than if we ignore it altogether and make a random choice instead (e.g., predict positive with probability s and negative with probability 1–s).

Expected loss for optimal thresholds

- If we choose thresholds optimally we are ignoring all operating points that are not on the ROC convex hull.
- In this case it can be shown that the expected 0/1 loss is equal to the refinement loss of the convex hull
  - shape rather than area
- One way to achieve this is through a perfectly calibrated classifier
  - implies convex ROC curve
  - zero calibration loss so Brier score = refinement loss
  - we can also show that in that case the Brier score is MAE/2

The many faces of ROC analysis

- 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 visualisation for understanding ML metrics
  - towards a theory of ML metrics
    - types of metrics, equivalences, skew-sensitivity
- ROC metrics 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

References

- C. Cortes and M. Mohri (2003). AUC optimization vs. error rate minimization. In Advances in Neural Information Processing Systems (NIPS'03). MIT Press.
- T.G. Dietterich, M. Kearns, and Y. Mansour (1996). Applying the weak learning framework to understand and improve C4.5. In L. Saitta, editor, Proc. 13th International Conference on Machine Learning (ICML'96), pp. 96-103. Morgan Kaufmann.
- C. Drummond and R.C. Holte (2000). Exploiting the cost (in)sensitivity of decision tree splitting criteria. In P. Langley, editor, Proc. 17th International Conference on Machine Learning (ICML'00), pp. 239-246.
- 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.
- 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 J.M. Rossiter and T.P. Martin, editors, Proc. 2003 UK workshop on Computational Intelligence (UKCI'03), pp. 38–44. University of Bristol.
- 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 (forthcoming). ROC 'n' rule learning — towards a better understanding of covering algorithms. Machine Learning, accepted for publication.
- D. Gamberger and N. Lavrac (2002). Expert-guided subgroup discovery: methodology and application. Journal of Artificial Intelligence Research, 17, 501–527.
- 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.
- A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki (1997). The DET curve in assessment of detection task performance. In Proc. 5th European Conference on Speech Communication and Technology, vol. 4, pp. 1895-1898.
- 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 (2002). Transforming classifier scores into accurate multiclass probability estimates. In Proc. 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02), pp. 694-699.

Acknowledgements

- Many thanks to:
  - Johannes Fürnkranz, Cèsar Ferri, José Hernández-Orallo, Nicolas Lachiche, Edson Matsubara, Ronaldo Prati & Shaomin Wu for joint work on ROC analysis and for some of the material
  - Jim Farrand & Ronaldo Prati for ROC visualisation software
  - Chris Drummond & Rob Holte for material and discussion on cost curves
  - Tom Fawcett & Rich Roberts for some of the ROC graphs