Scale Matters

On the many uses of calibration in machine learning

Peter.Flach@bristol.ac.uk

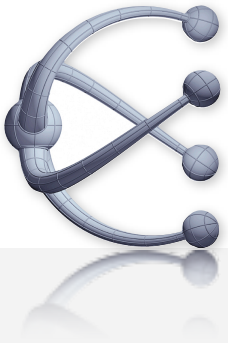
Peter A. Flach

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

Department of Computer Science

Intelligent Systems Laboratory

University of Bristol



Abstract

- Calibration is the process of adjusting measurements to a standard scale.
- In machine learning it is most commonly understood in relation to the class probability estimates of a probabilistic classifier:
    - we say that a classifier is well-calibrated if among all instances receiving a probability estimate *p* for a particular class, the proportion of instances having the class in question is approximately *p*.
- The advantage of a well-calibrated classifier is that near-optimal decision thresholds can be directly derived from the operating condition (class and cost distribution).

Abstract

- In this talk I explore various methods for classifier calibration, including the isotonic regression method that relates to ROC analysis.

- I will discuss how these methods can be applied to single features, resulting in a very general framework

  - in which features carry class information, and

  - categorical features can be turned into real-valued ones and *vice versa*.

Abstract

- I will also discuss an alternative notion of calibration whereby a classifier's score quantifies the proportion of positive predictions it makes at that threshold.

- I will introduce the ROL curve, a close companion of ROC curves that allows to quantify the loss at a particular predicted positive rate.

- Rate-calibrated classifiers have an expected loss that is linearly related to AUC, which vindicates AUC as a coherent measure of classification performance

    - contrary to recent claims in the literature (David Hand, MLj 2008).

Outline

I. Getting started

- ROC curve, AUC, ROC convex hull, logistic and isotonic calibration, …

II. Feature transformation and calibration

- Discretisation, adding a scale to arbitrary features

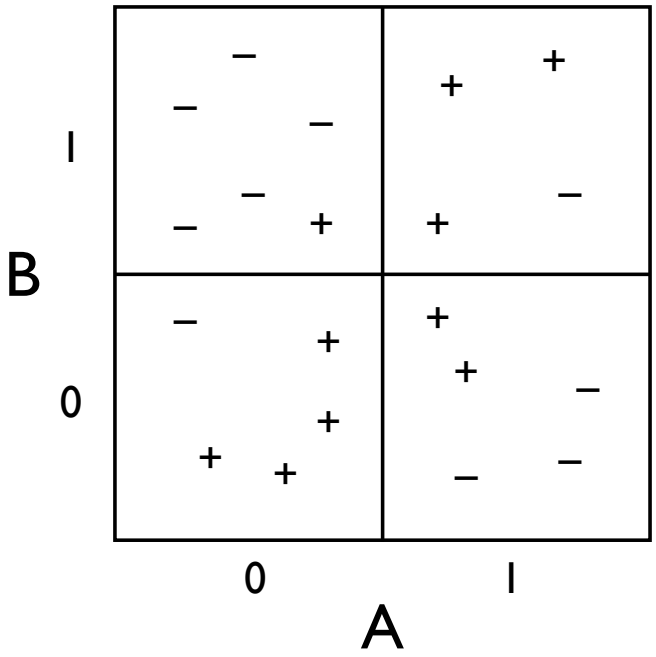
III. Rate calibration and AUC as a performance metric

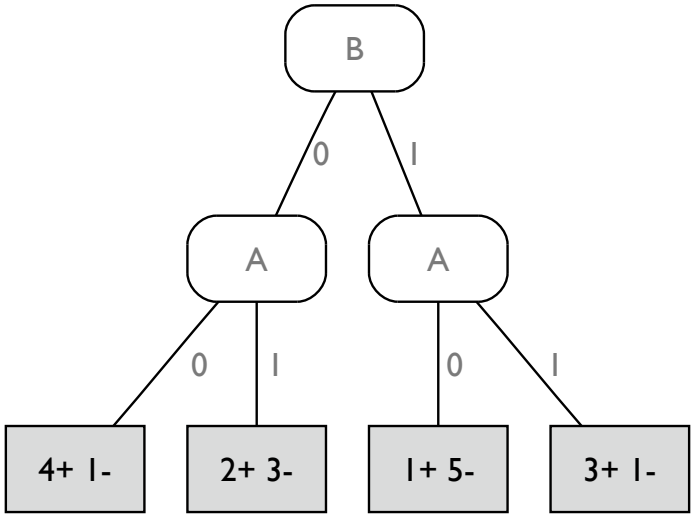
- The ROL curve, the relation between AUC and expected loss

I Getting started

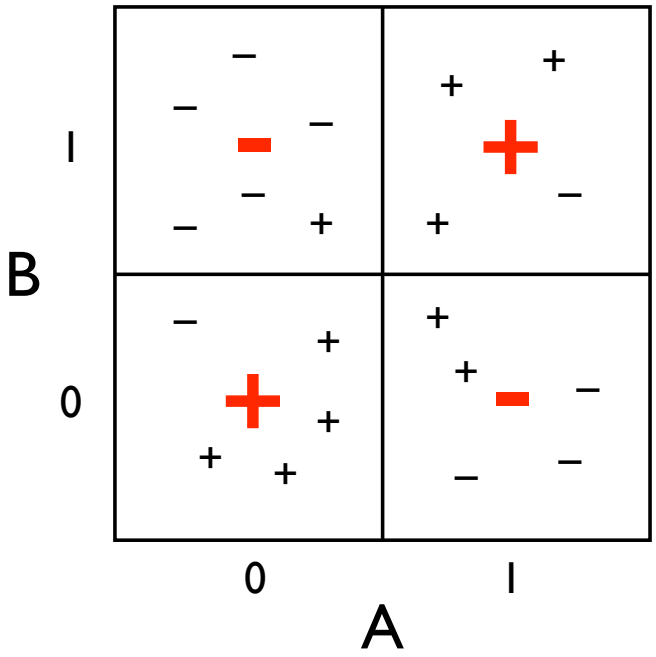
- Classification, ranking, probability estimation
- ROC curve, convex hull, concavities
- Calibration: parametric and non-parametric
- Class imbalance, cost sensitivity

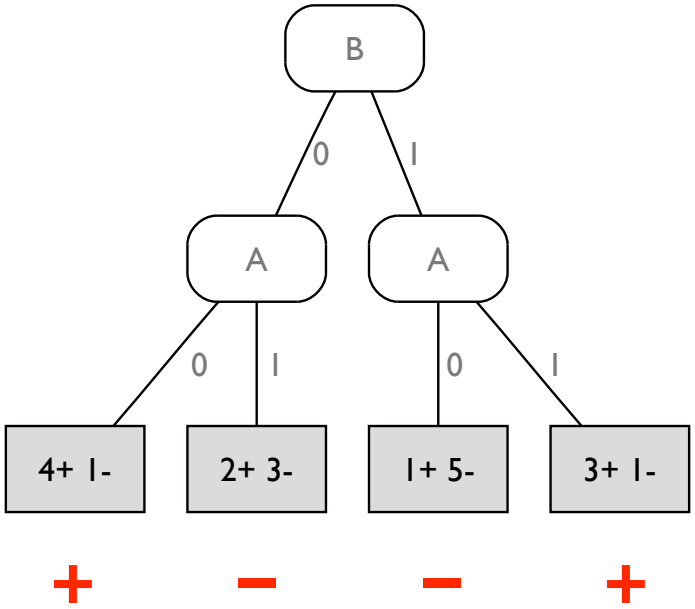
Decision tree classifier





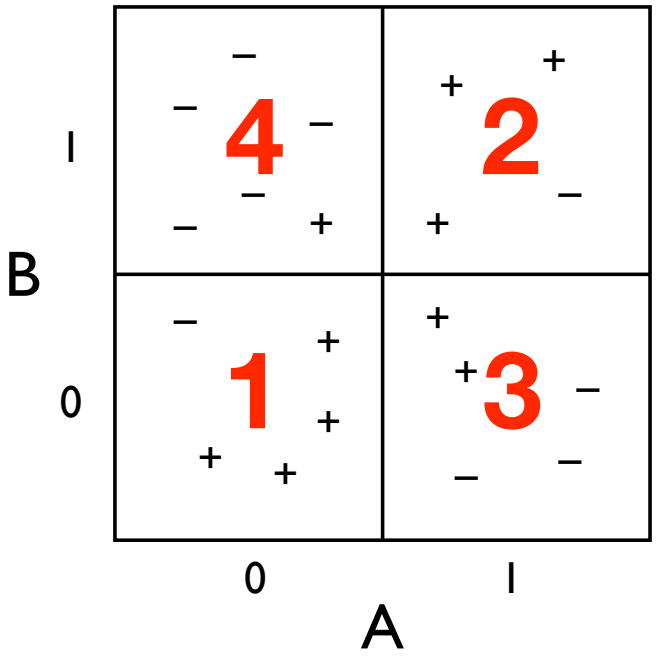
Decision tree classifier

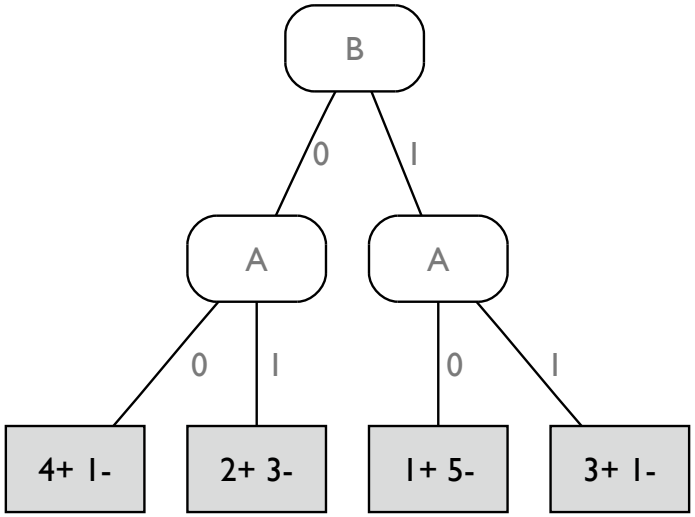




Labels obtained by majority vote decision rule.

Decision tree ranker

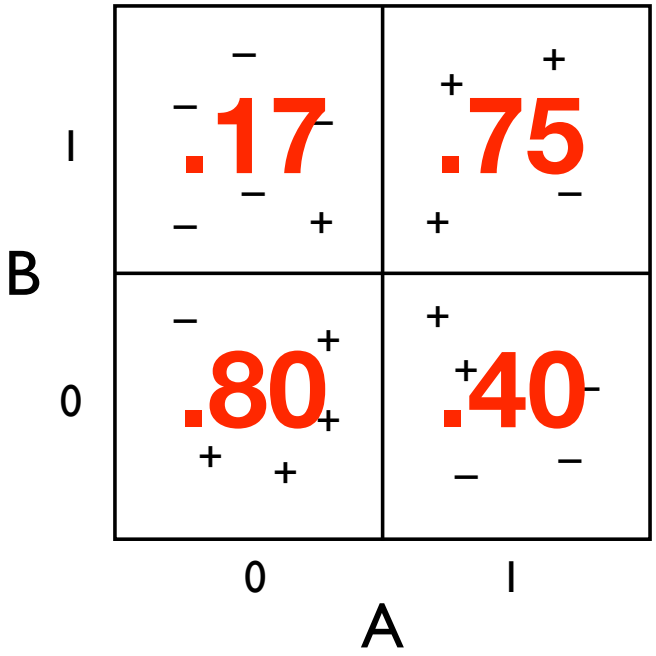


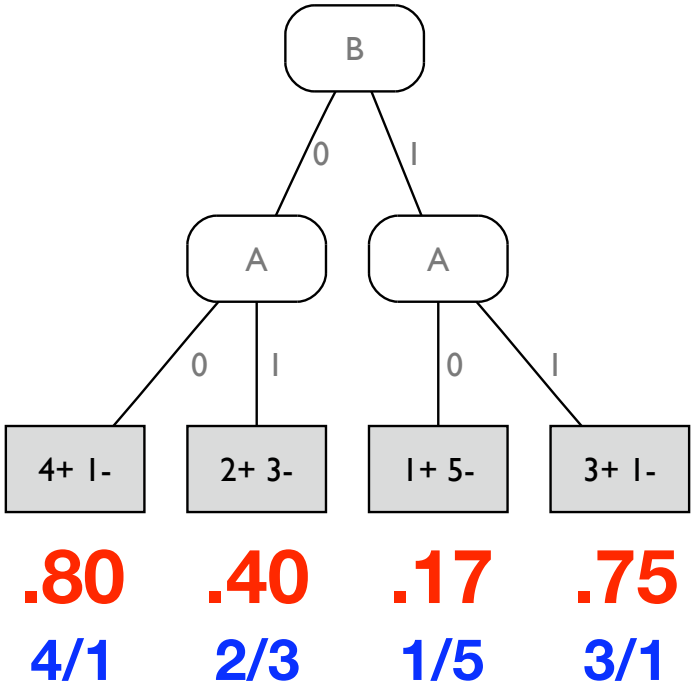


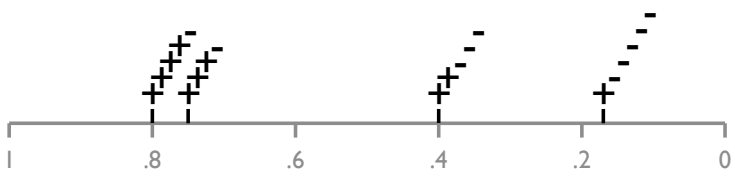
1 3 4 2

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

Decision tree probability estimator

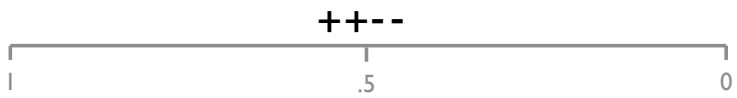




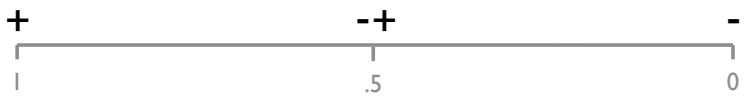


Classification ≠ ranking ≠ probability estimation

- Better probabilities ≠ better ranking

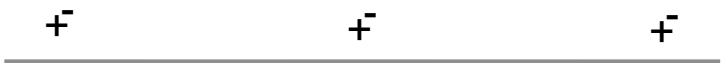


  - no ranking errors, mean squared error ≈ 0.25



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

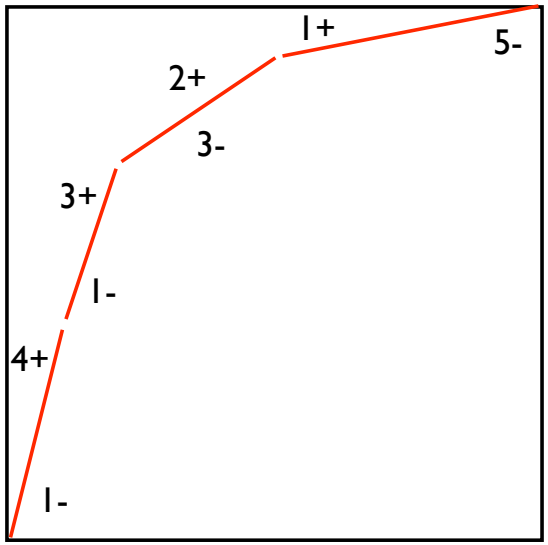
- Better classification ≠ better ranking

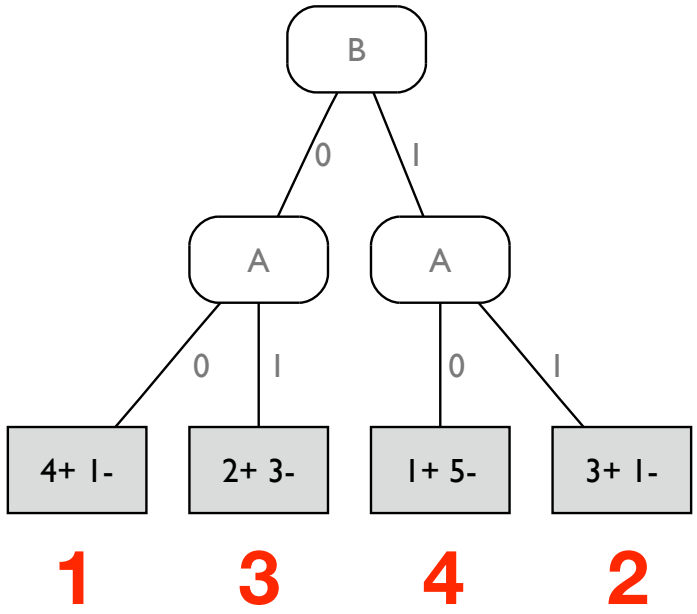


  - 4.5 ranking errors, 3 classification errors

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

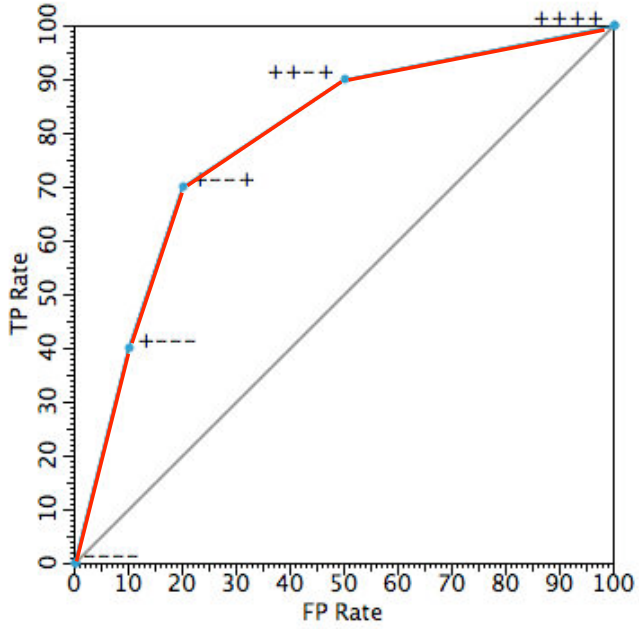
Visualising ranking performance

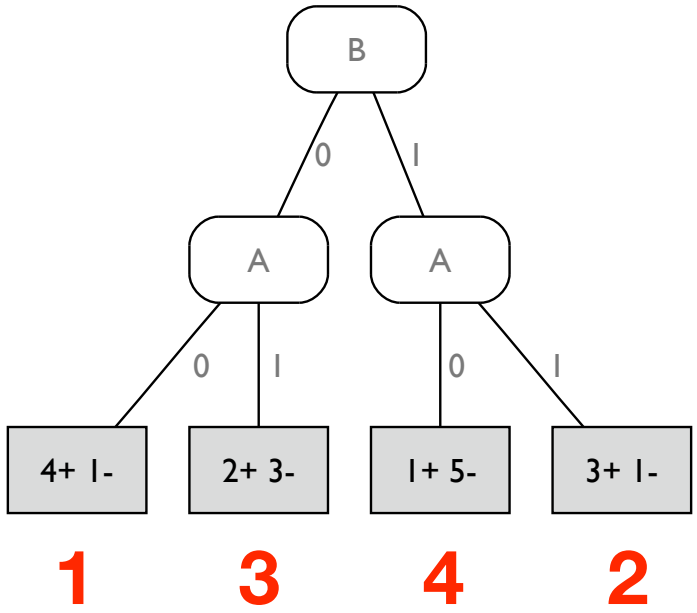




Each leaf is visualised by a line segment; by stacking these line segments in the ranking order we can keep track of cumulative performance (aka Lorenz curve or ROC curve).

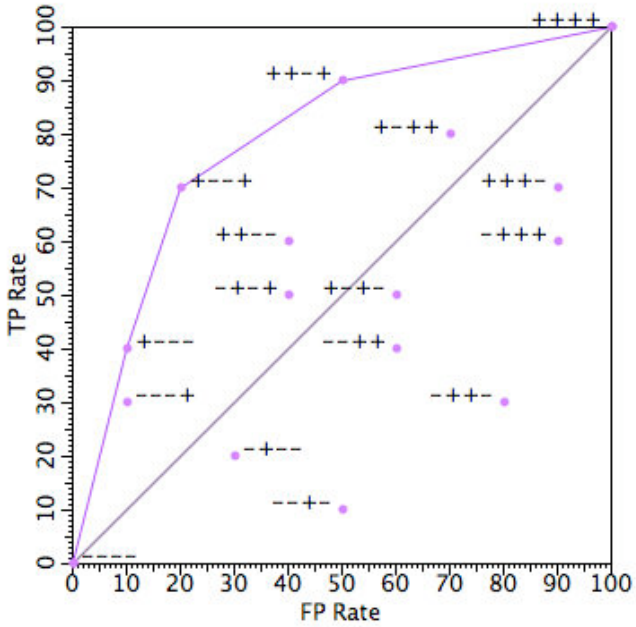
Visualising ranking performance (2)

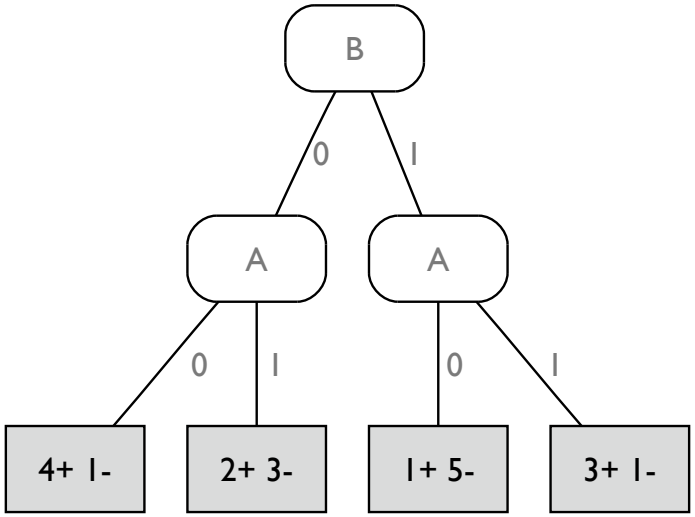




Counts on the axes mean that slopes represent *posterior odds*; normalising these by the number of positives/negatives means that slopes represent *likelihood ratios* instead.

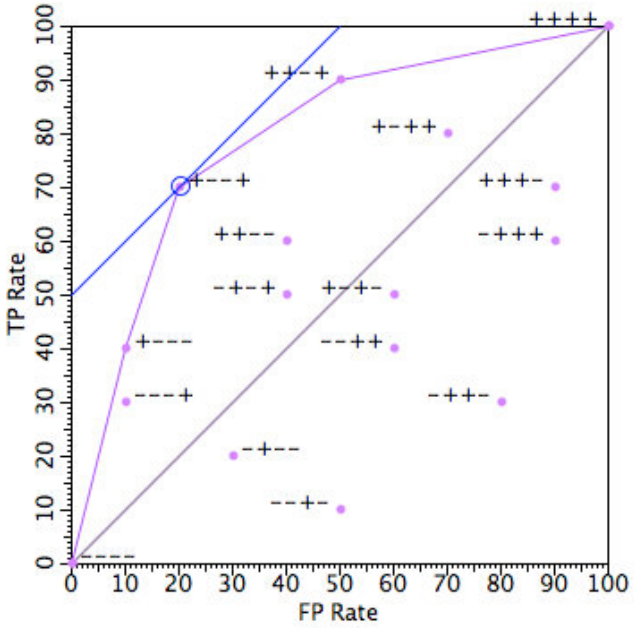
All possible tree labellings

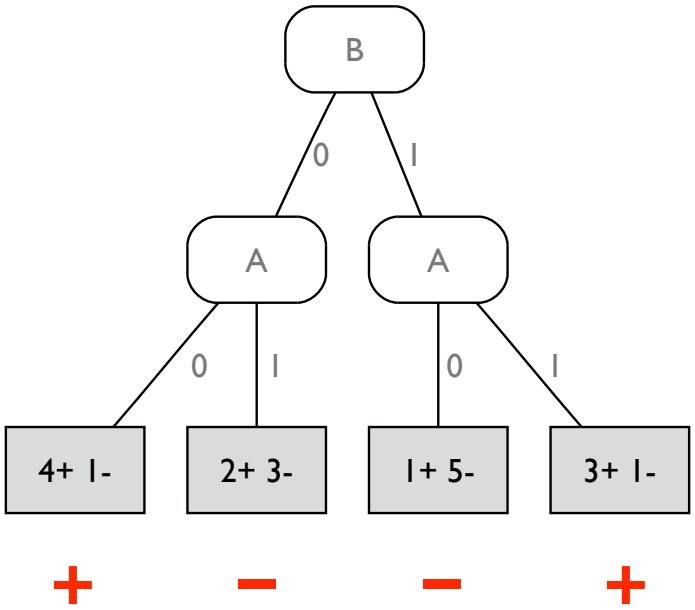




A tree with *n* leaves has 2^n possible labellings, which summarise all possible model behaviours. Notice that a labelling and its opposite (e.g., +—–+ and –++–) are each other's mirror image in ROC space (through (1/2,1/2)).

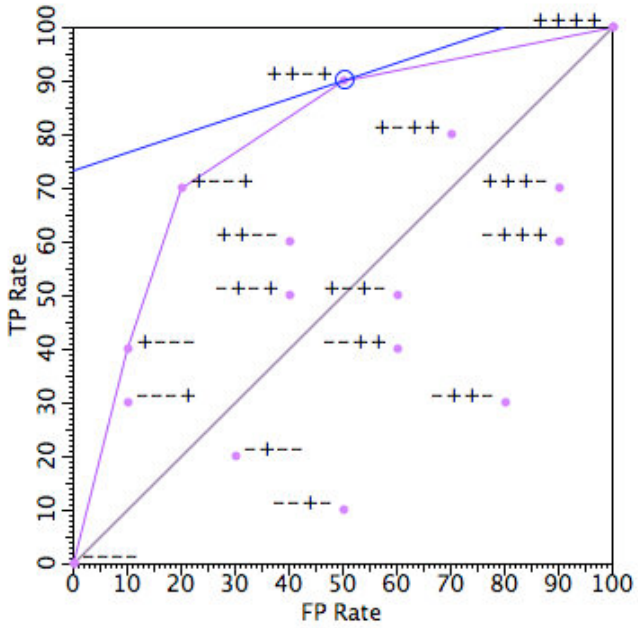
Choosing the optimal labelling

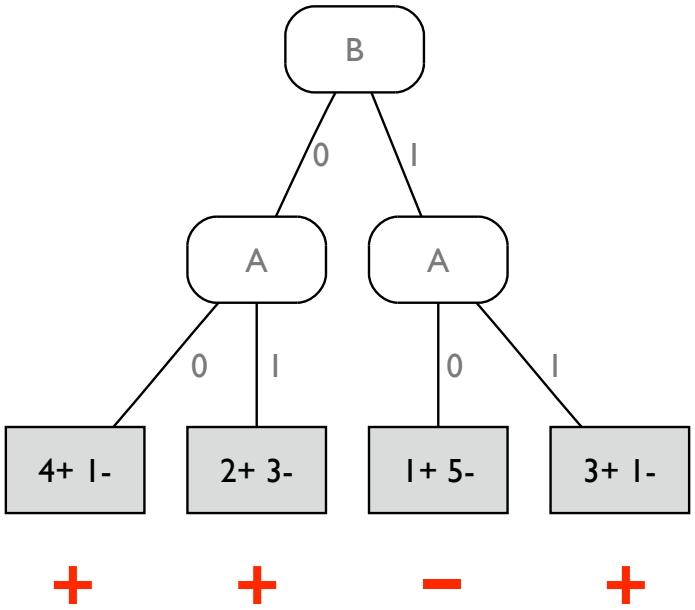




The above labelling is optimal for uniform prior odds (i.e., positives and negatives are equally prevalent/important)

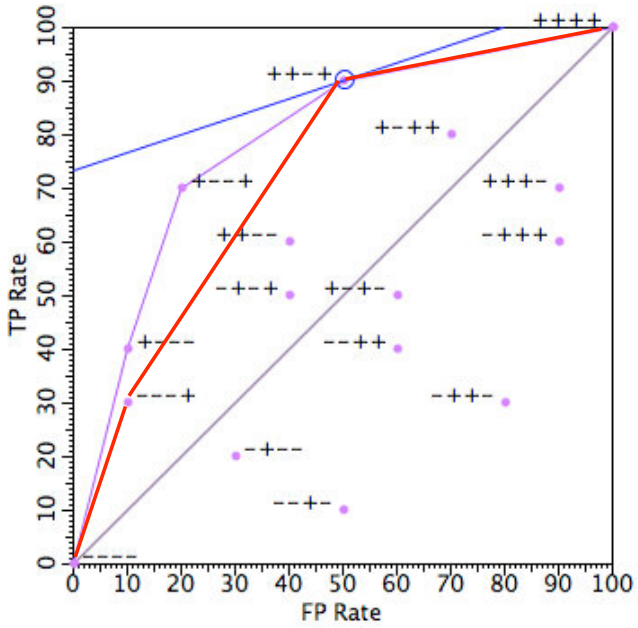
Choosing the optimal labelling (2)

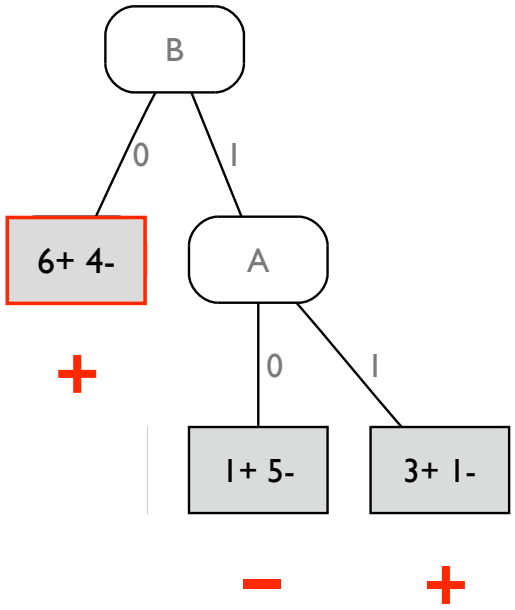




The second leaf is relabelled + if positives are three times as prevalent/important as negatives; notice that this effectively prunes the left subtree.

Pruning considered harmful...



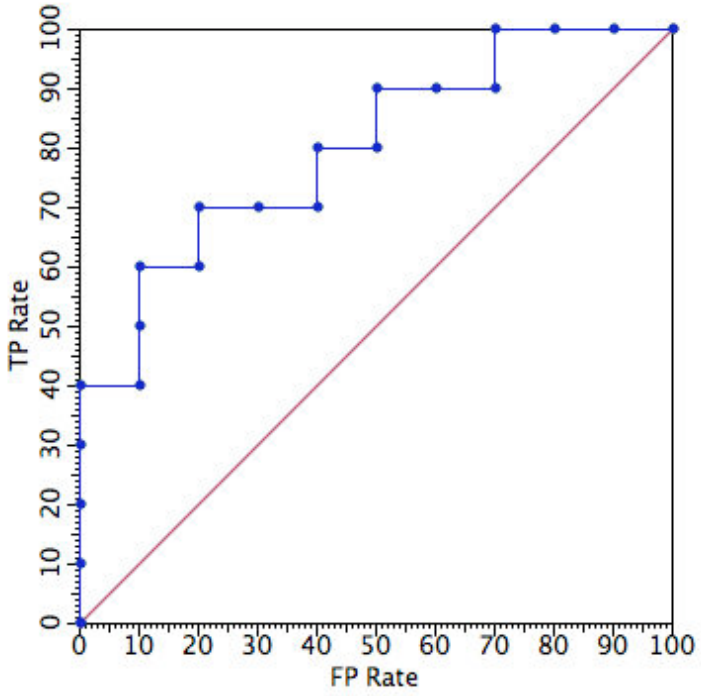


However, notice that pruning *decreases* ranking performance, as measured by the area under the curve (AUC, estimates the probability that a random positive is ranked before a random negative).

From a ranking to a ROC curve

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

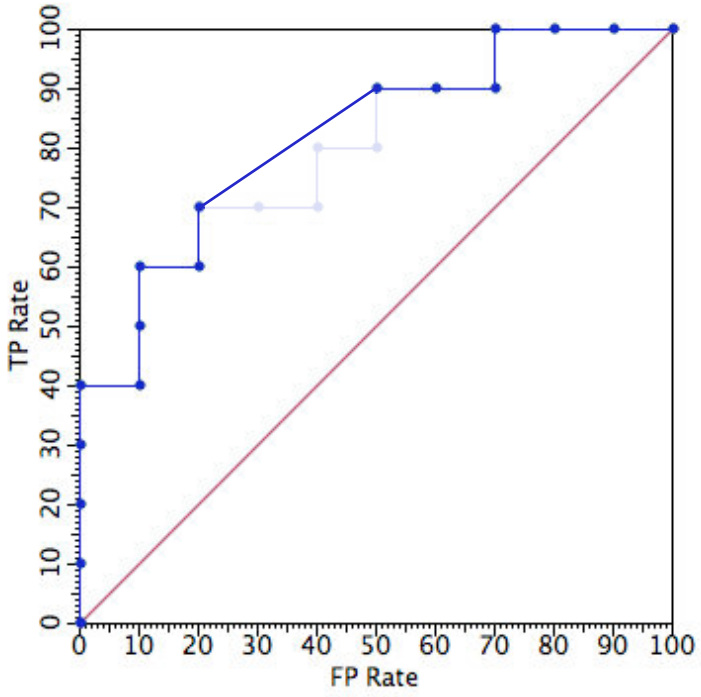
- start in (0,0)
- get the next instance in the ranking
  - if it is positive, move 1/Pos up
  - if it is negative, move 1/Neg right



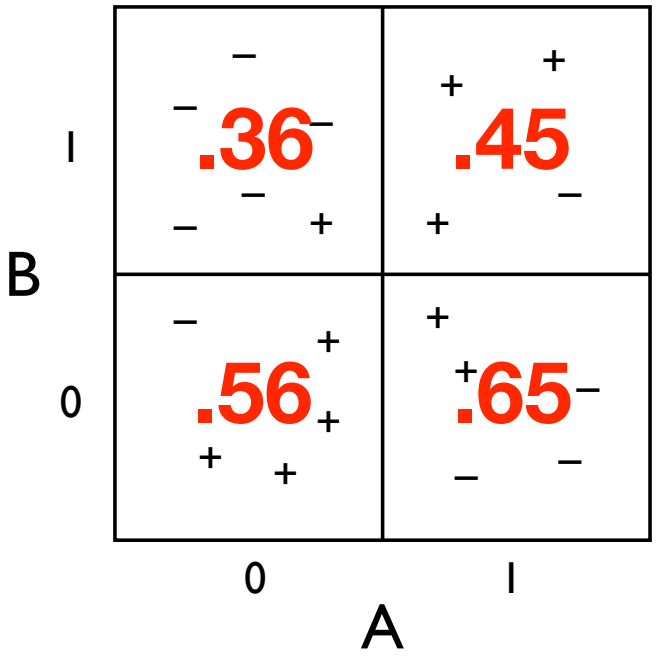
From a ranking to a ROC curve

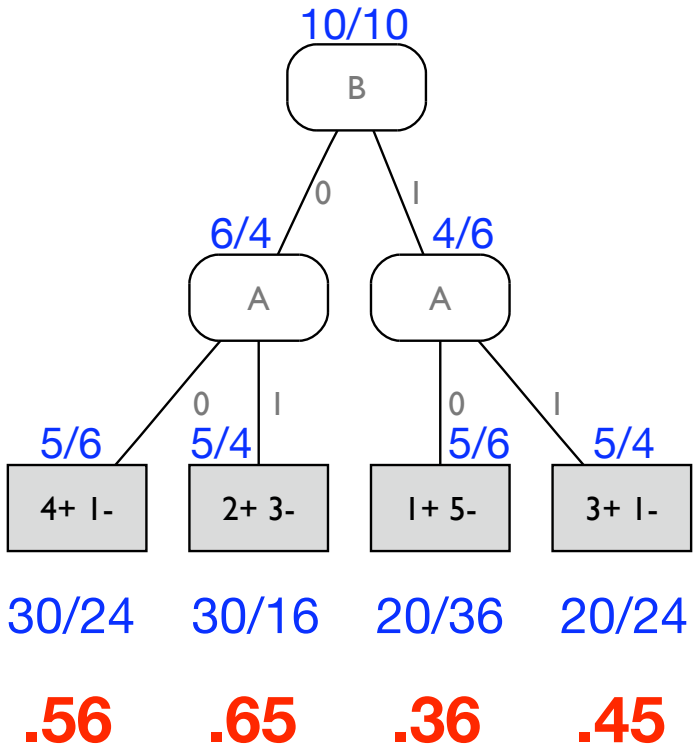
+ + + + + - + + - + -+-+ - - +- - -

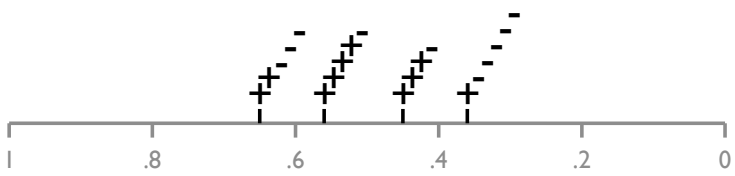
- start in (0,0)
- get the next instance in the ranking
  - if it is positive, move 1/Pos up
  - if it is negative. move 1/Neg right
  - make diagonal move in case of ties



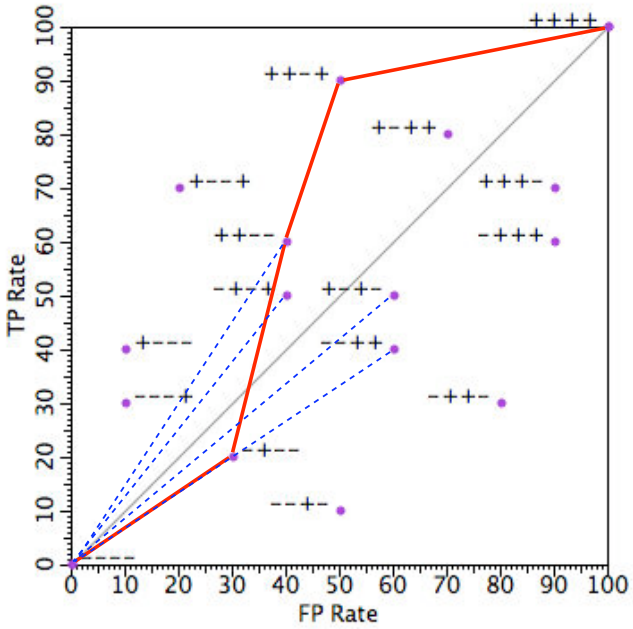
Naive Bayes probability estimator

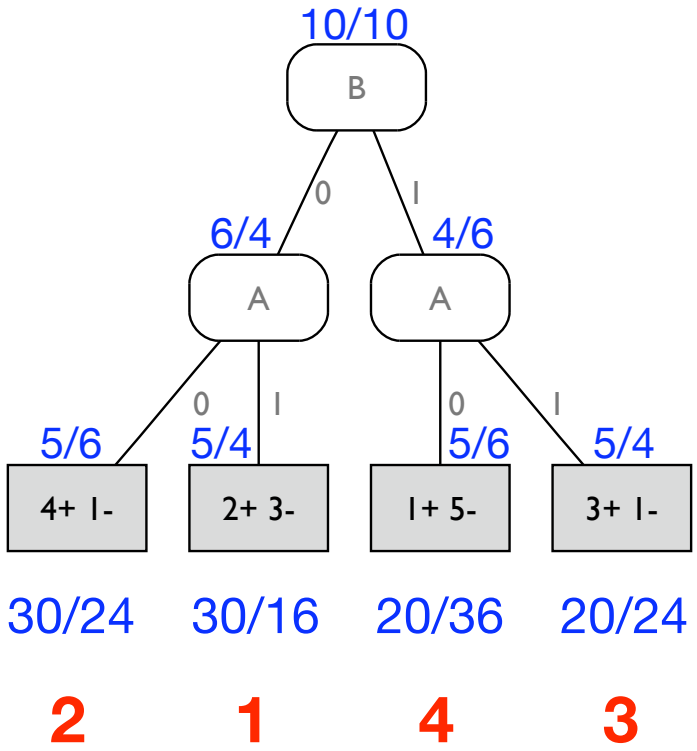






Naive Bayes ROC curve



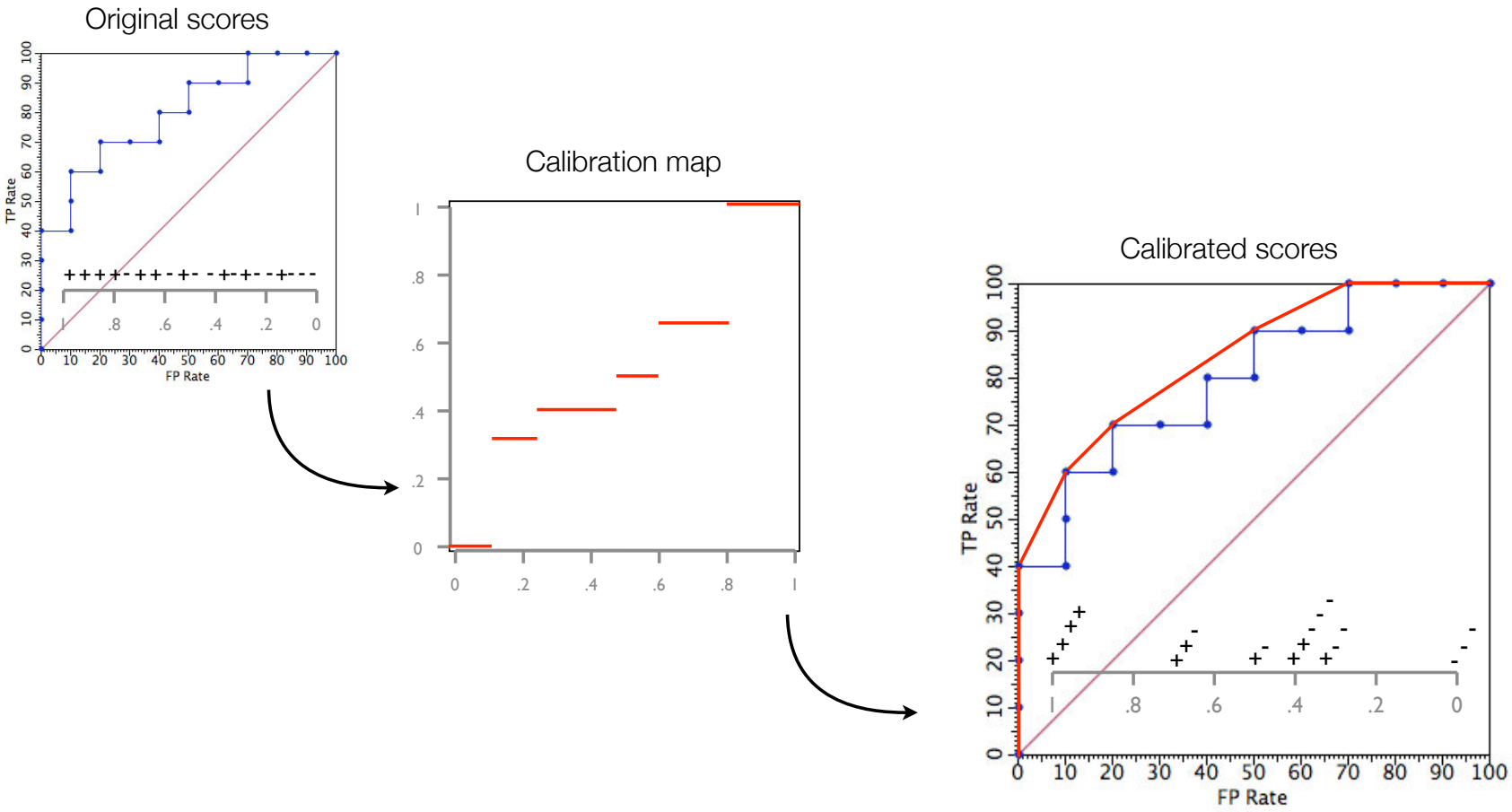


The concavity is caused by misleading marginal probabilities (*cf*. A=1, B=0). Repairing this would require access to the true joint probabilities.

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

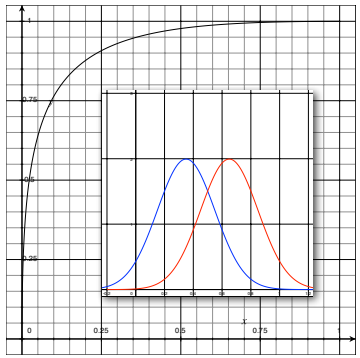
Isotonic calibration = pool adjacent violators



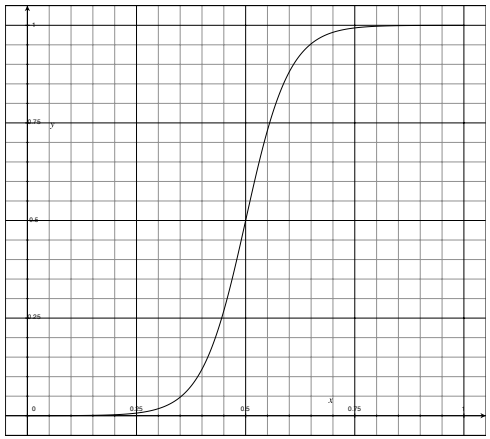
Piecewise constant calibration map leads to more ties in the ranking.

Parametric alternative: logistic calibration

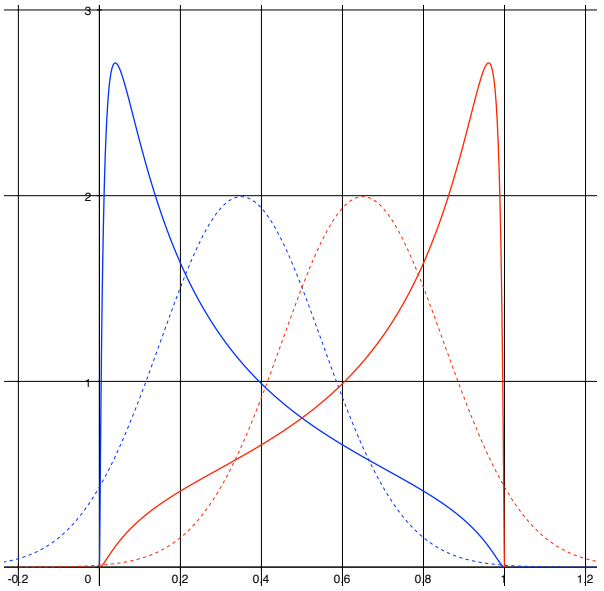
Normally distributed scores



Calibration map: logistic function

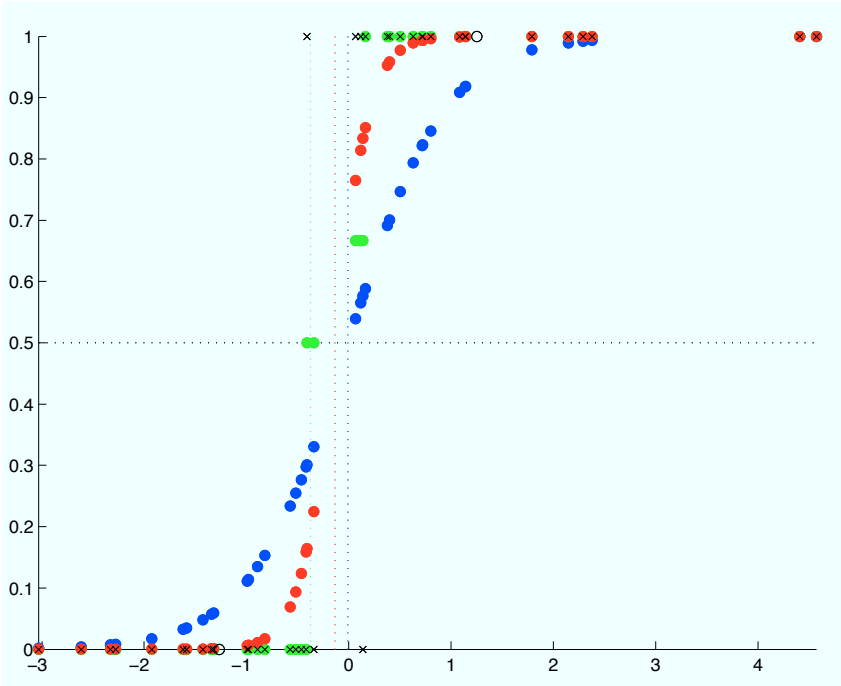


Score distributions after calibration



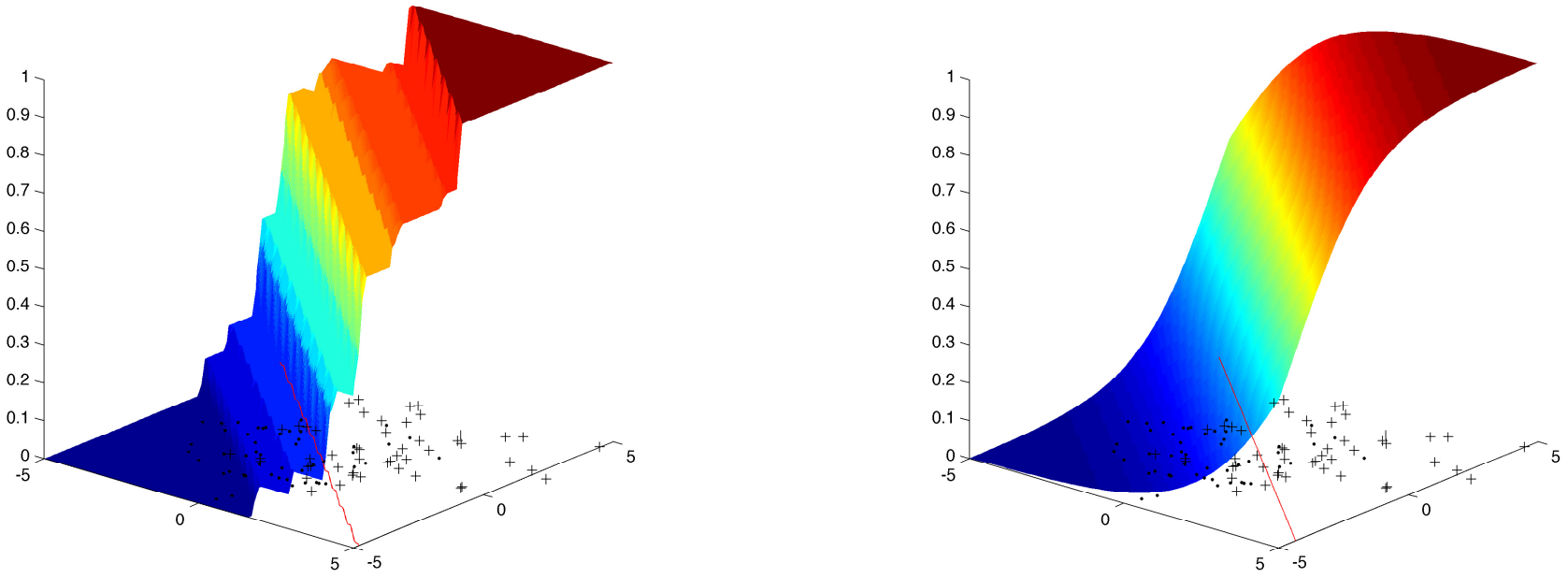
Logistic regression optimises this directly.

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

Some notation

	Predicted		*Total*
Actual	TP	FN	Pos
	FP	TN	Neg

- Pos: number of positives
- Neg: number of negatives
- TP, FP, TN, FN: number of true/false positives/negatives
- Acc: number of correctly classified examples (Acc = TP+TN)
- Err: number of incorrectly classified examples (Err = FP+FN)
- π_0: proportion of positives (π_0 = Pos/(Pos+Neg))
- π_1: proportion of negatives ($\pi_1 = 1-\pi_0$)
- tpr, fpr, tnr, fnr: true/false positive/negative rates (tpr = TP/Pos, fpr = FP/Neg, tnr = 1–fpr, fnr = 1–tpr)
- acc: proportion of correctly classified examples ($acc = \pi_0 * tpr + \pi_1 * tnr$)
- err = 1–acc

Class imbalance or cost sensitivity?

- c_0, c_1: misclassification cost for positive/negative
  - Cost-sensitive loss is $c_0\pi_0$*fnr + $c_1\pi_1$*fpr
  - To express this on a scale commensurate to error rate it is convenient to set $c_0+c_1=2$ and $c = c_0/2$, $1-c = c_1/2$.
- The combined operating condition is ***skew*** $z = c_0\pi_0/(c_0\pi_0+c_1\pi_1)$
  - Uniform misclassification costs: $z = \pi_0$
  - Uniform class distribution: $z = c$

Posterior odds or likelihood ratio?

- Decision trees estimate posterior odds. This is most convenient if the class distribution doesn't change and the misclassification costs define the operating condition.

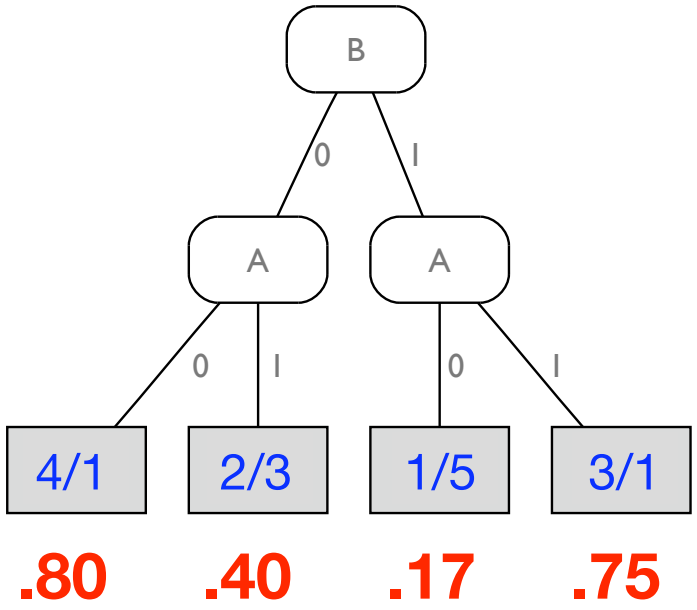
- If both class and cost distributions are likely to change, it is better to work with the likelihood ratio (i.e., slope of the ROC curve) and skew as operating condition.

- Working with likelihood ratios means rebalancing the classes:

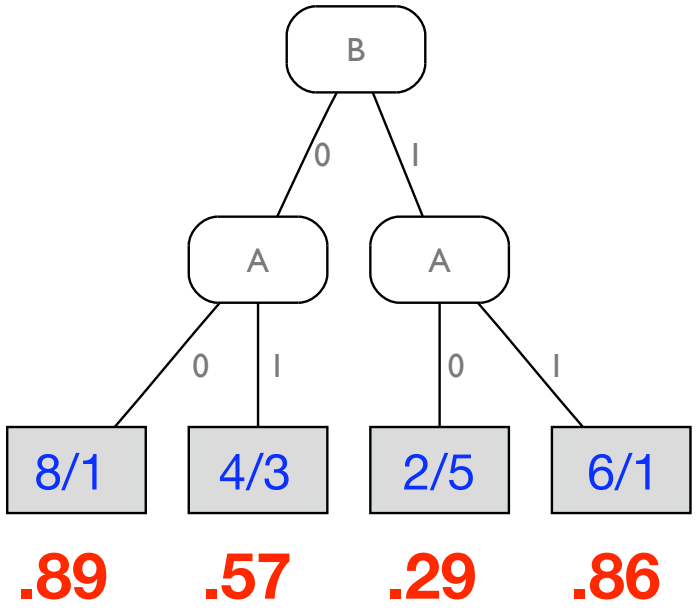
  - posterior odds po = lr * π_0/π_1

  - likelihood ratio lr = po * π_1/π_0

Posterior odds or likelihood ratio (2)



(Re)balanced classes



Unbalanced classes

Probabilities or log-odds?

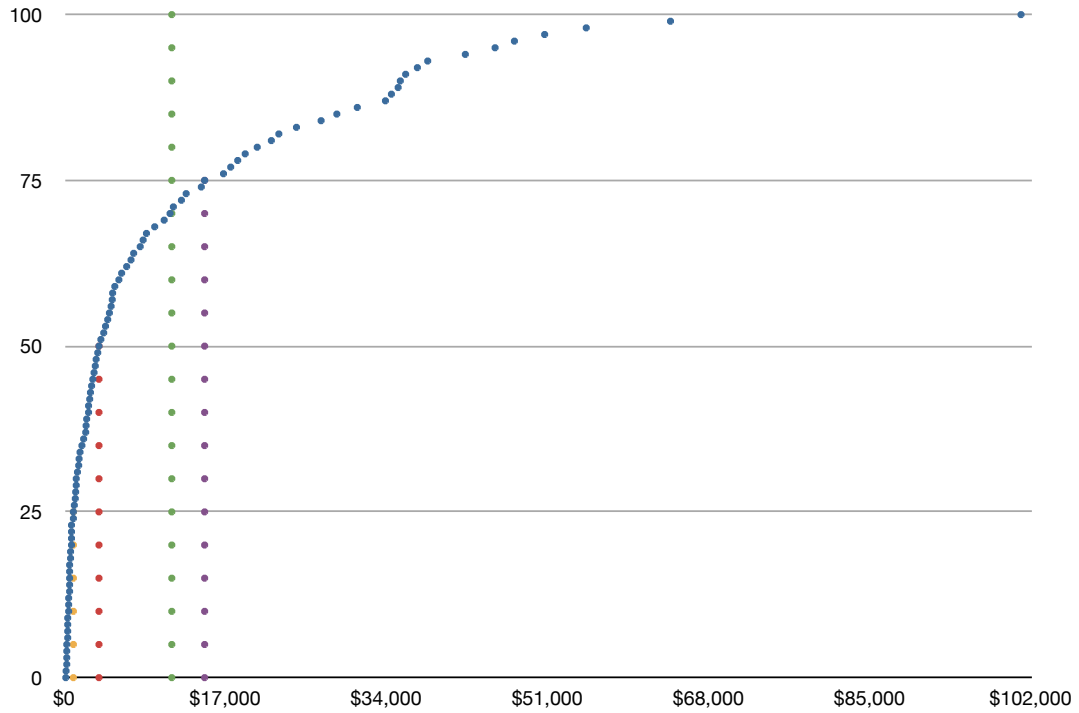
- Likelihood ratios and posterior odds can be translated to class probabilities. All these are expressed on a multiplicative scale.
  - e.g. $p = \mathrm{lr} / (\mathrm{lr}+1)$
- Sometimes an additive scale is preferred for further processing. In that case we can convert probability p to $\ln p/(1-p) = \ln \mathrm{lr}$.
  - Intuitive interpretation is signed distance from the decision boundary.
  - Allows taking arithmetic mean etc.
  - Useful for feature calibration.

II Feature transformation and calibration

Kind	*Order*	*Scale*	*Tendency*	*Dispersion*	*Shape*
Categorical	×	×	mode	n/a	n/a
Ordinal	√	×	median	quantiles	n/a
Real-valued	√	√	mean	range, interquartile range, variance, standard deviation	skewness, kurtosis

↓ to, from →	*Real-valued*	*Ordinal*	*Categorical*	*Boolean*
Real-valued	**normalisation**	**calibration**	**calibration**	**calibration**
Ordinal	**discretisation**	**ordering**	**ordering**	**ordering**
Categorical	**discretisation**	**unordering**	**grouping**	
Boolean	**thresholding**	**thresholding**	**binarisation**	

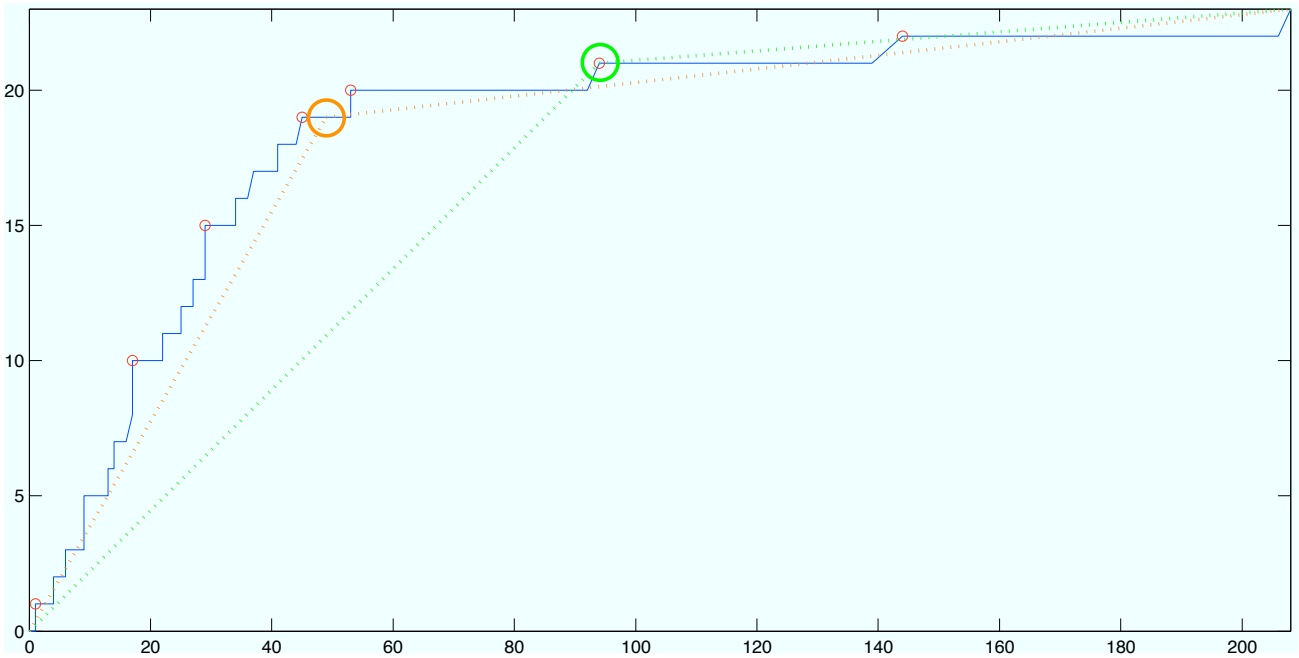
Example feature: country GDP per capita



First quartile - median - mean - third quartile

(data from Wolfram Alpha)

Class-based splitting: Euro vs non-Euro countries



The **orange** split sets the threshold equal to the mean, selecting 19 Euro countries and 49 non-Euro countries. The **green** split sets the threshold equal to the median, selecting 21 Euro countries and 94 non-Euro countries.

Feature discretisation by recursive partitioning

Algorithm 10.1: RecPart(S, f, Q) – supervised discretisation by means of recursive partitioning.

Input : set of labelled instances S ranked on feature values $f(x)$; scoring function Q.

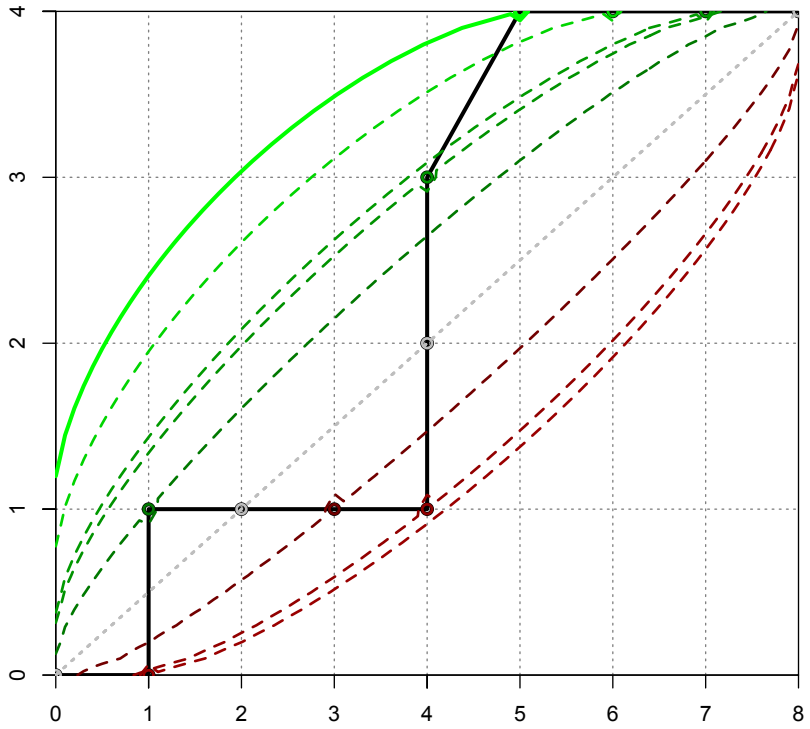
Output : sequence of thresholds $t_1, \ldots, t_{k-1}$.

```
1 if stopping criterion applies then return ∅;
2 Split S into S_l and S_r using threshold t that optimises Q ;
3 T_l = RecPart(S_l, f, Q);
4 T_r = RecPart(S_r, f, Q);
5 return T_l ∪ {t} ∪ T_r;
```

This is best-known in combination with an MDL-based heuristic (Fayyad & Irani, 1992) but can be used with many other performance metrics.

Feature discretisation: example

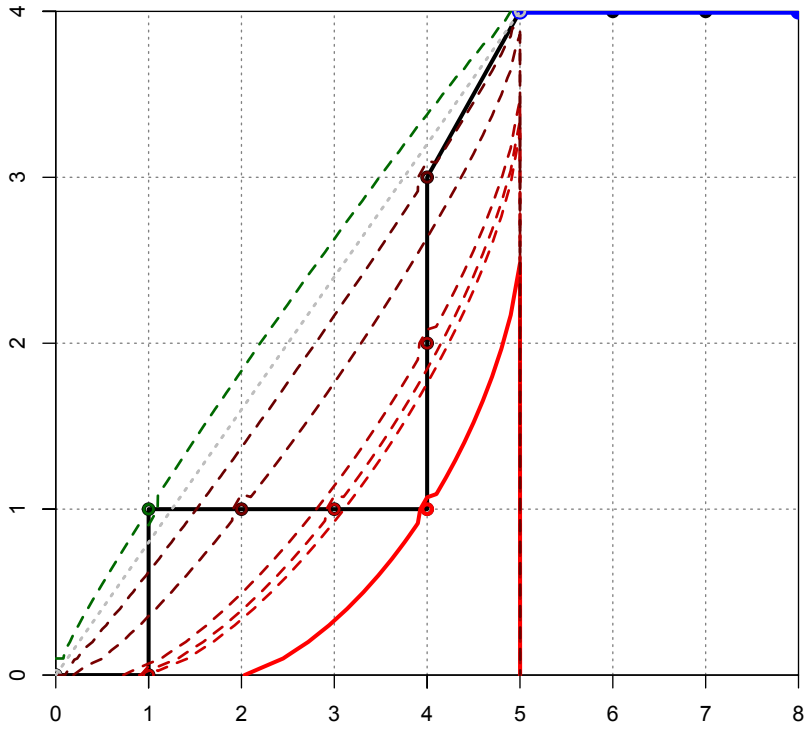
Instance	*Value*	*Class*
e_1	−5.0	−
e_2	−3.1	+
e_3	−2.7	−
e_4	0.0	−
e_5	7.0	−
e_6	7.1	+
e_7	8.5	+
e_8	9.0	−
e_9	9.0	+
e_{10}	13.7	−
e_{11}	15.1	−
e_{12}	20.1	−



Curved lines show information gain isometrics: further away from the diagonal means higher information gain.

Feature discretisation: example (2)

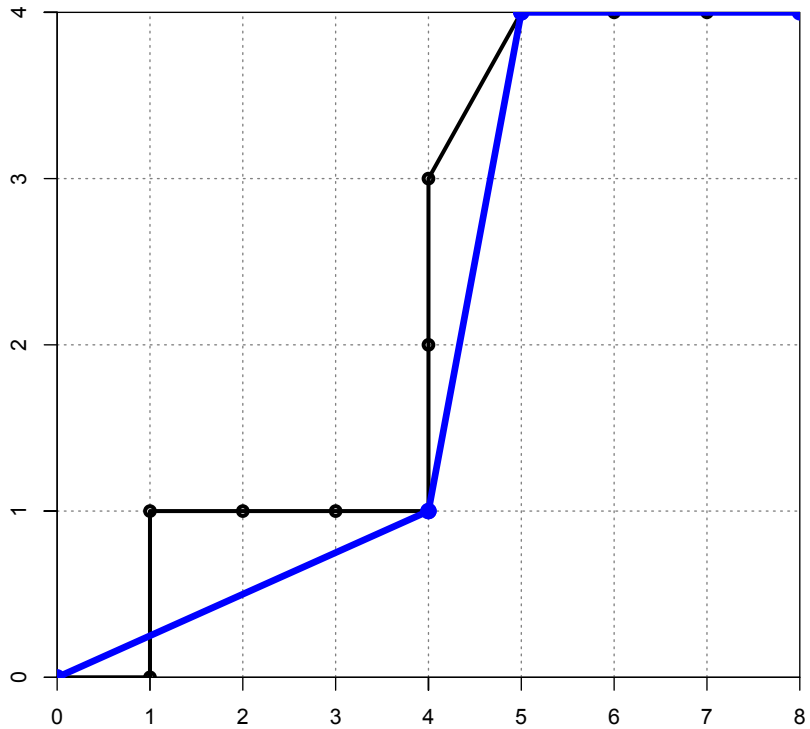
Instance	*Value*	*Class*
e_1	-5.0	$-$
e_2	-3.1	$+$
e_3	-2.7	$-$
e_4	0.0	$-$
e_5	7.0	$-$
e_6	7.1	$+$
e_7	8.5	$+$
e_8	9.0	$-$
e_9	9.0	$+$
e_{10}	13.7	$-$
e_{11}	15.1	$-$
e_{12}	20.1	$-$



The best split results in one pure bin; partitioning proceeds recursively on the remainder.

Recursive partitioning as feature calibration

Instance	*Value*	*Class*	*Calibrated value*
e_1	−5.0	−	
e_2	−3.1	+	
e_3	−2.7	−	0.33
e_4	0.0	−	
e_5	7.0	−	
e_6	7.1	+	
e_7	8.5	+	
e_8	9.0	−	0.86
e_9	9.0	+	
e_{10}	13.7	−	
e_{11}	15.1	−	0.00
e_{12}	20.1	−	

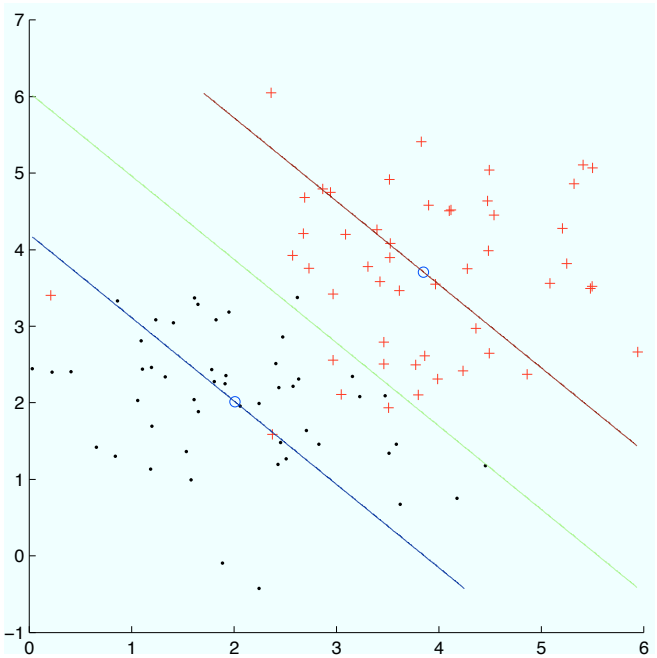


1st bin: po = 1/4 (*p*=0.20); lr = 1/4 * 8/4 = 1/2 (*p*=0.33)
2nd bin: po = 3/1 (*p*=0.75); lr = 3/1 * 8/4 = 6/1 (*p*=0.86)
3d bin: po = 0/3 (*p*=0.00); lr = 0/3 * 8/4 = 0/3 (*p*=0.00)

Variants of feature calibration

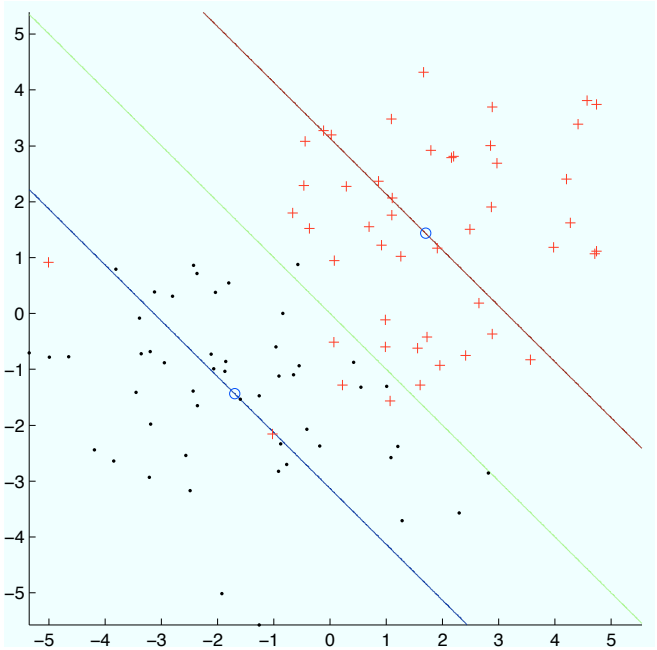
- Tracing the ROC curve by means of recursive partitioning
- Tracing the ROC convex hull -> ordinal calibration
- Logistic calibration

Logistic feature calibration



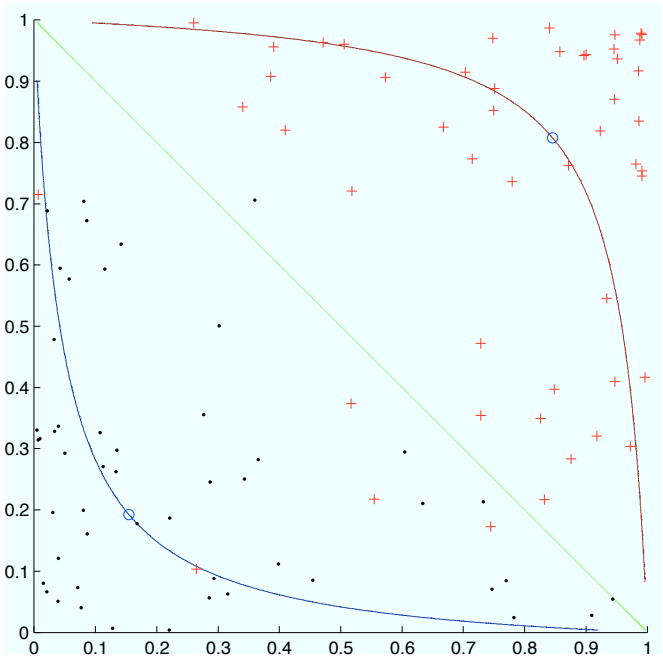
Difference-between-means classifier

Logistic feature calibration (2)



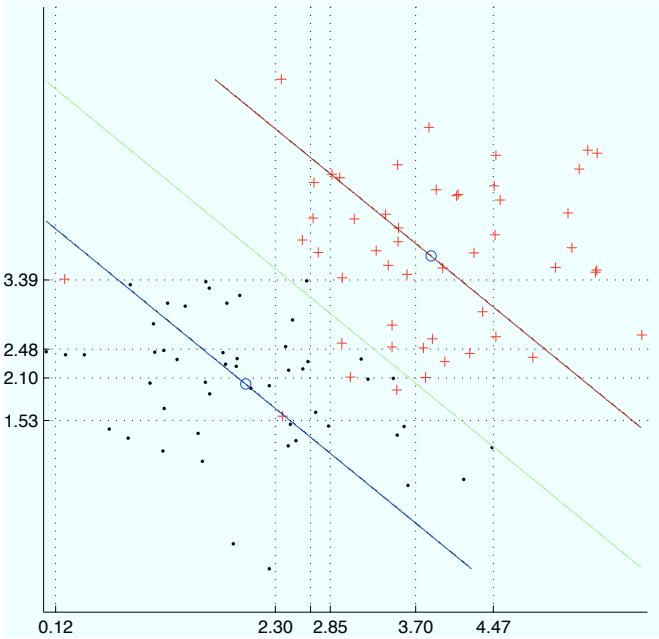
Calibrated features in log-odds space

Logistic feature calibration (3)



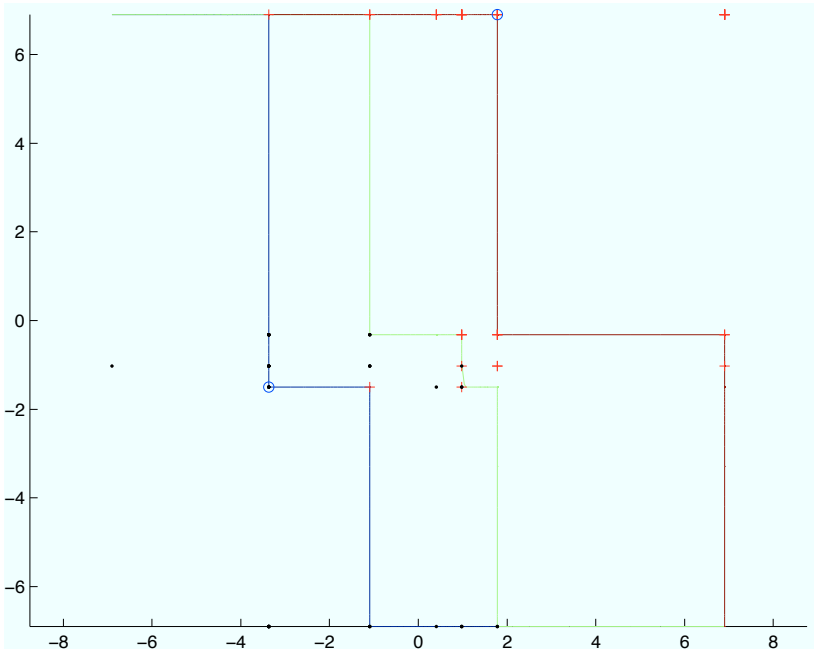
Calibrated features in probability space

Isotonic feature calibration



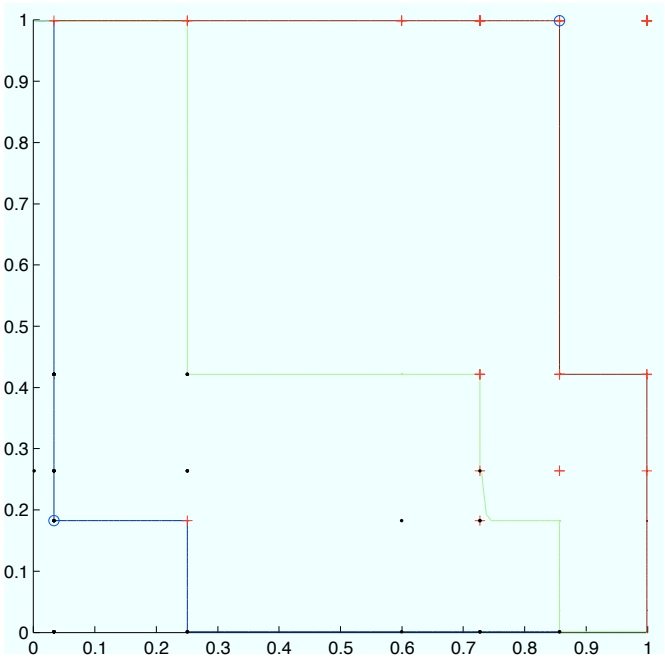
Convex hull discretisation (NB. respects bin order)

Isotonic feature calibration (2)



Calibrated features in log-odds space

Isotonic feature calibration (3)



Calibrated features in probability space

Why calibrated features are useful

- They extend the scope of models that require real-valued features (e.g. linear models) to include ordinal and categorical features.
- They allow unsupervised methods such as *K*-means or PCA to take class labels into account.
- Naive Bayes is a case in point, as its model can be *entirely* explained as feature calibration into log-odds space.
- Non-parametric calibration provides a useful alternative that leads to many novel variants of existing methods.

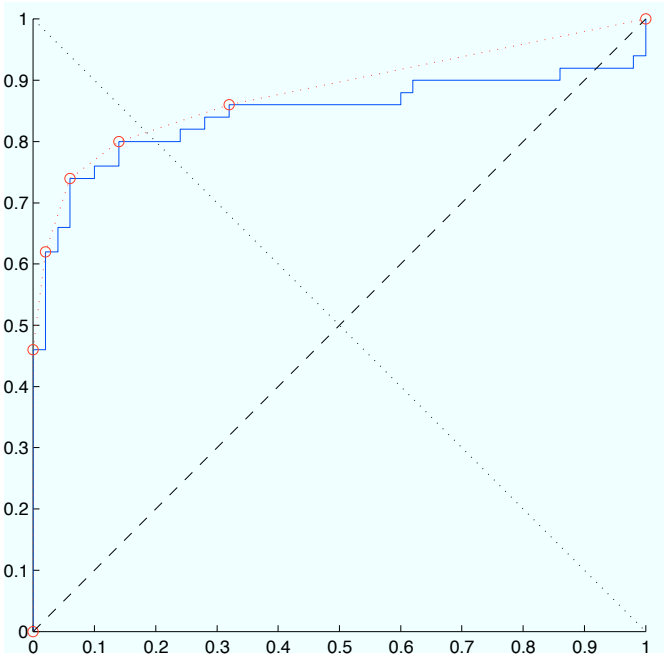
III Rate calibration and AUC as performance metric

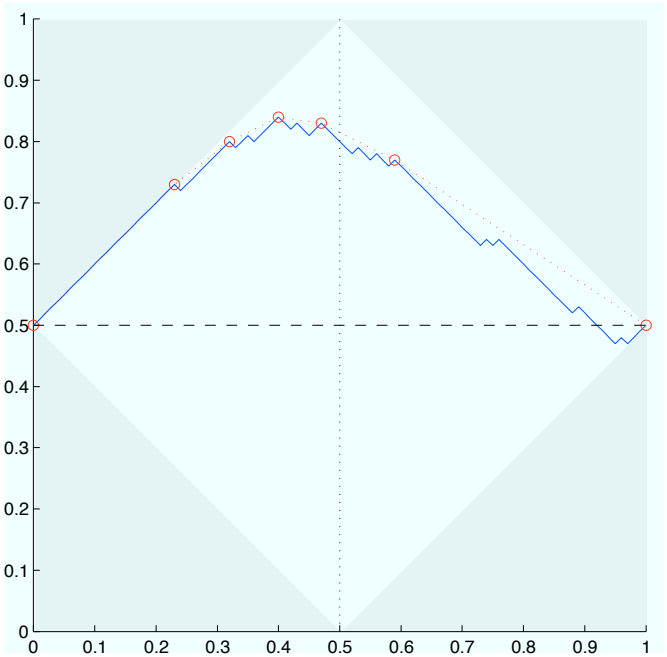
- It is often said that AUC aggregates classification performance across decision thresholds. But it is not entirely clear what this means exactly.

- David Hand has recently shown that a particular kind of expected loss can be related to AUC, but in a model-dependent way. This has led him to say that AUC is fundamentally incoherent as a measure of classification performance.

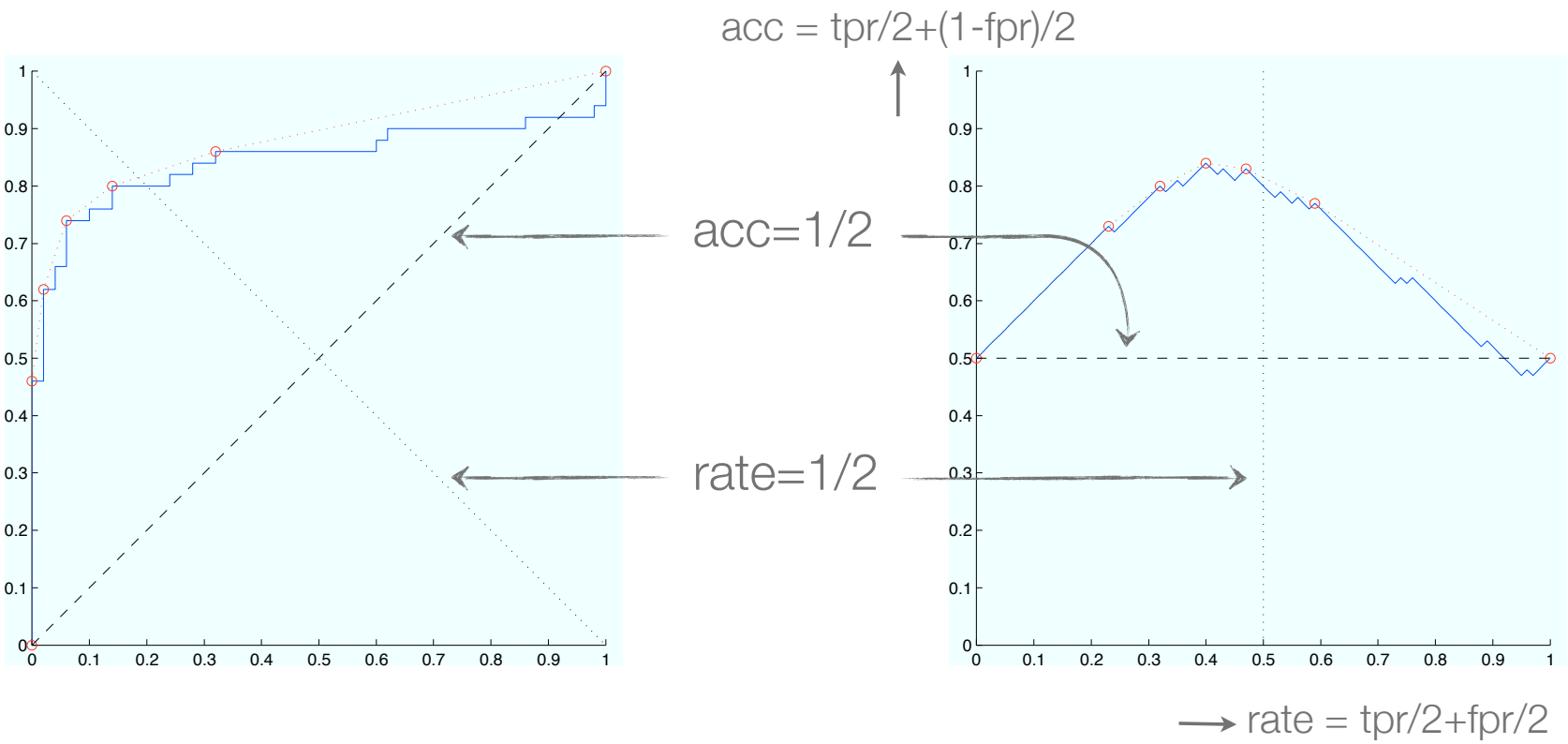
- I will show that AUC is linearly related to a slightly different notion of expected loss, and therefore coherent.

From ROC curve to ROL curve

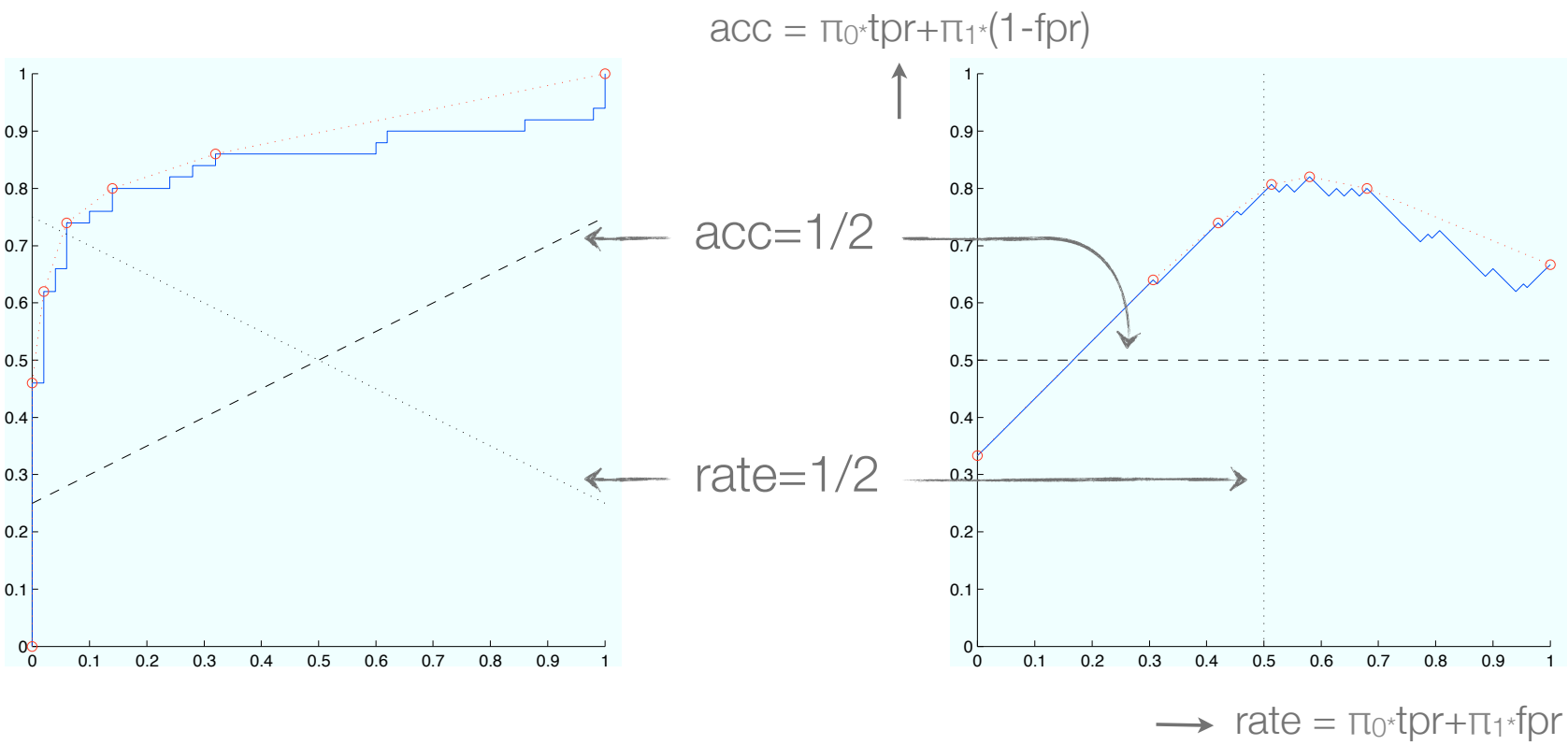




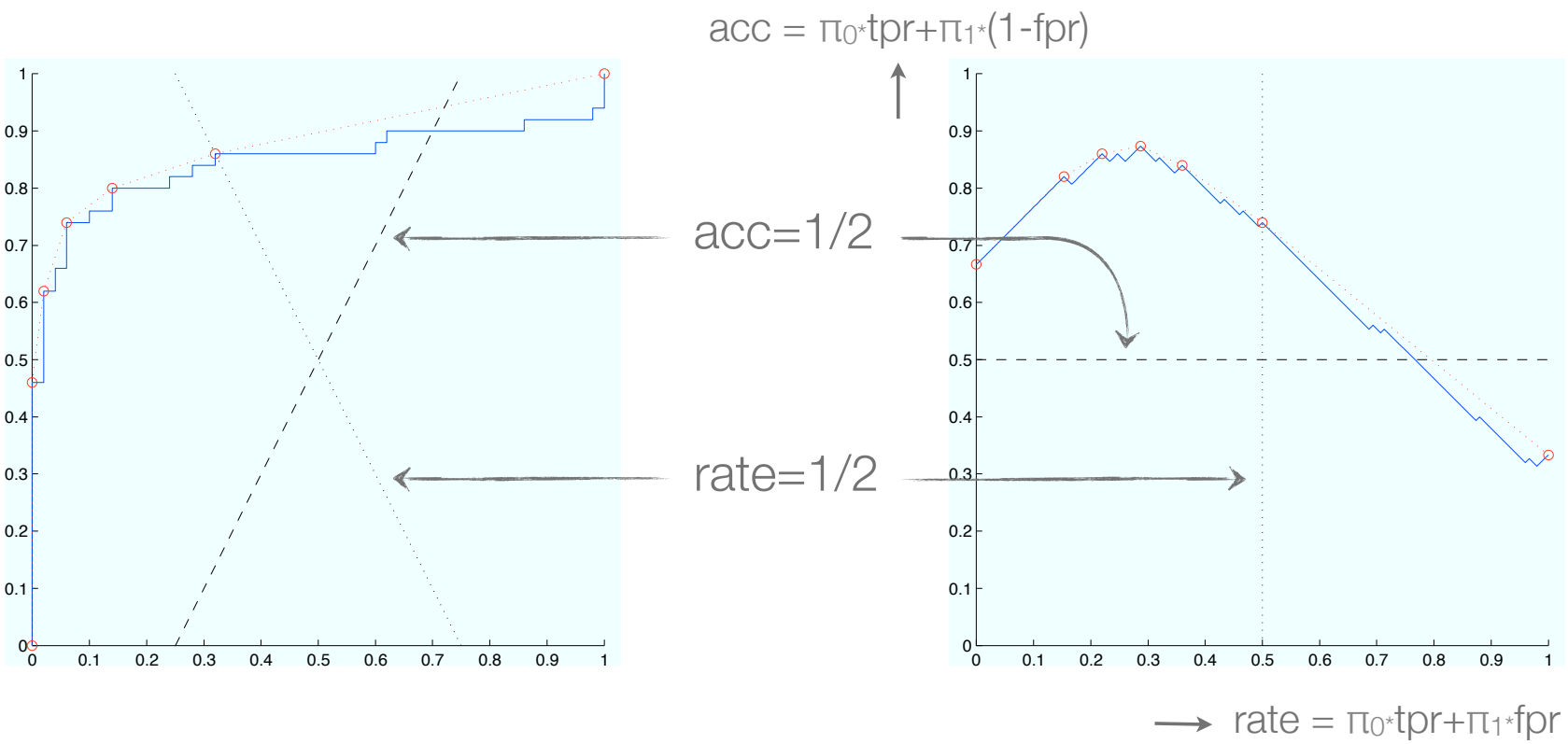
From ROC curve to ROL curve ($\pi_0 = \pi_1$)



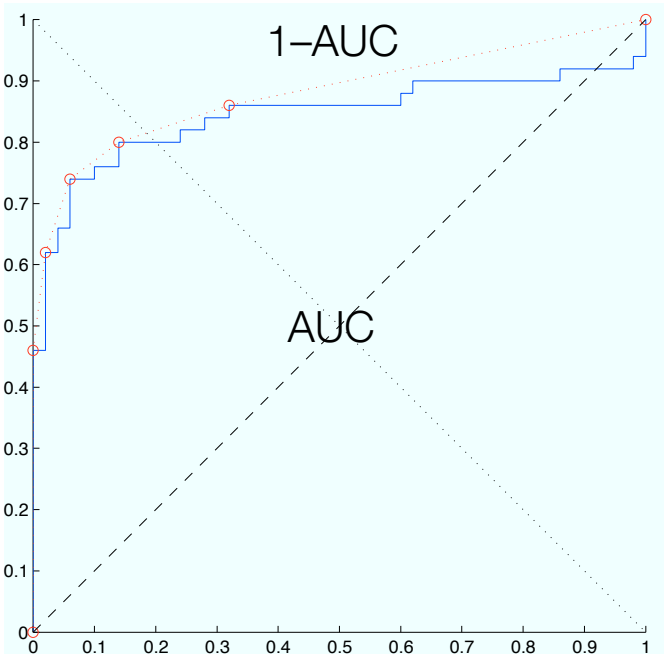
From ROC curve to ROL curve ($\pi_0 > \pi_1$)

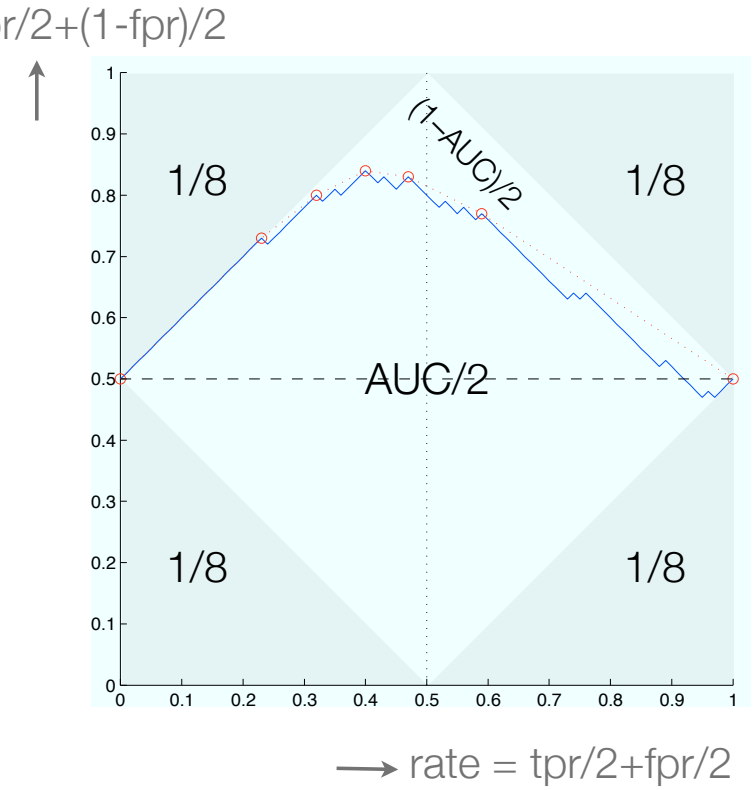


From ROC curve to ROL curve ($\pi_0 < \pi_1$)



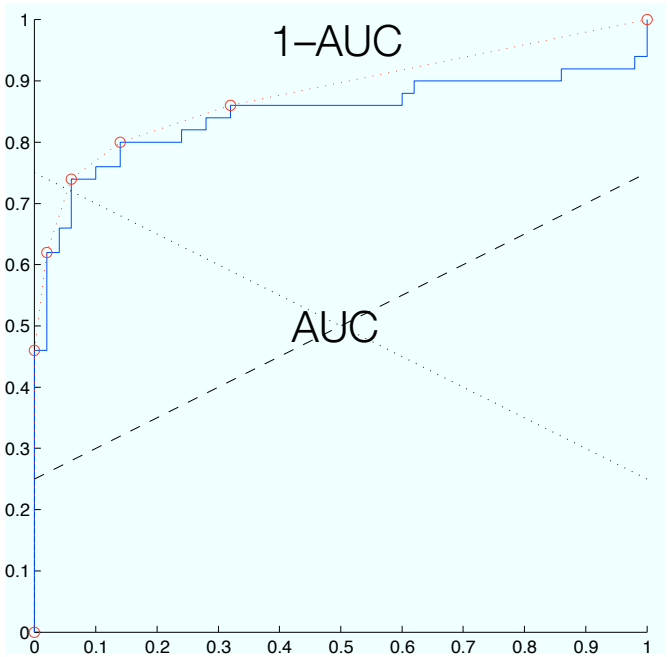
AUC and expected loss



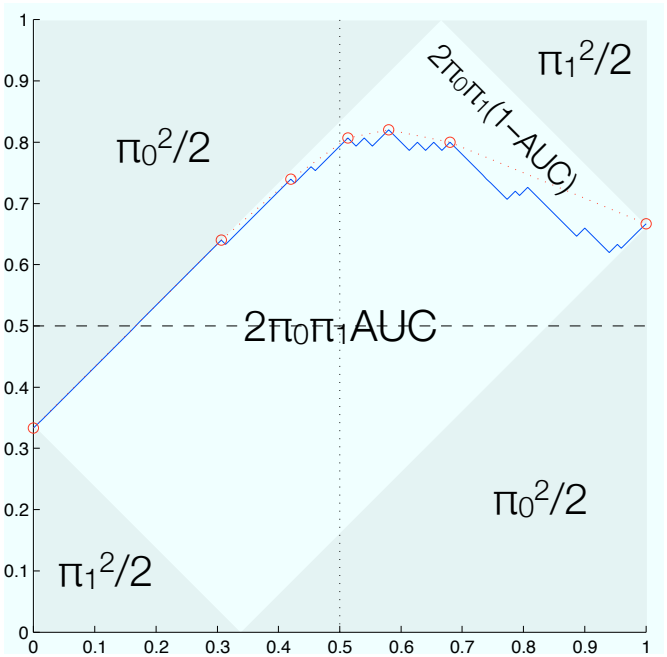


The expected loss for uniform rate is (1-AUC)/2+1/4 = (1-2AUC)/4+1/2.

AUC and expected loss



acc = π_0*tpr+π_1*(1-fpr)



rate = π_0*tpr+π_1*fpr

Expected loss for uniform rate is $2\pi_0\pi_1(1\text{-AUC})+\pi_0^2/2+\pi_1^2/2 = \pi_0\pi_1(1\text{-}2\text{AUC})+1/2$.

AUC as a 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 $\pi_0\pi_1(1-2AUC)+1/2$.

- This can be generalised to a cost-sensitive setting with c as operating condition, assuming that we set the rate independently of c

    - which is not clever but can be improved without changing the basic argument.

- But wait a minute: wasn't there a fundamental problem with AUC as a measure of classification performance?

David Hand's criticism of AUC (MLj, 2008)

- AUC can be interpreted as the expected true positive rate, averaged over all false positive rates.

- For any given classifier we don't have direct access to the false positive rate, and so we average over possible decision thresholds.

- The relationship between decision thresholds and operating conditions under which this threshold is optimal is model-specific, and so the way AUC aggregates performance over possible operating conditions is model-specific.

- Expectations over the operating condition are task-specific and not dependent on the model, and so AUC may make a model's classification performance look better or worse than it actually is.

Our response to Hand (ICML'11)

- Joint work with José Hernández-Orallo and César Ferri
- Hand's key assumption is that decision thresholds are set optimally.
  - This is a strong — and often unrealistic — assumption.
  - Under this assumption expected loss is not quantified by AUC, but by a different ROC-related measure (work in progress).
  - Hand's alternative, the *H* measure, is the area under the optimal cost curve.
- AUC is related to expected loss under several other threshold choice methods.
  - including the rate-driven one just considered.

Concluding remarks

- The scale on which its scores are expressed is an important — but often neglected — ingredient of a classifier. Calibration of that scale should be a routine operation. There are several options, depending on whether we want a multiplicative or additive scale, rebalance the classes, etc.

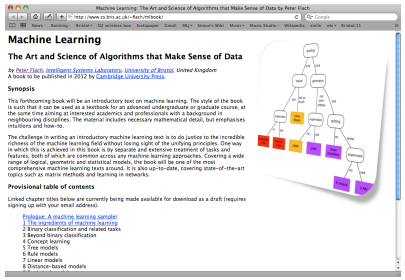
- Calibration is also a very useful tool in feature engineering, generalising discretisation which throws away order and scale.

- Calibration against predicted positive rate allows us to precisely formalise the interpretation of AUC as a classification performance metric.

- Acknowledgements:
  - José Hernández-Orallo and César Ferri for joint work on ROC analysis and the proper interpretation of AUC
  - Ronaldo Prati and Edson Takashi Matsubara for joint work on feature calibration

Two shameless plugs

- Many examples in this talk are from my forthcoming book:
  - Machine Learning: the art and science of algorithms that make sense of data. Cambridge University Press, 2012.
  - http://www.cs.bris.ac.uk/~flach/mlbook/



- In September 2012 the Bristol Intelligent Systems Lab hosts the European Conference on Machine Learning and Data Mining (ECML-PKDD).
  - http://www.ecmlpkdd2012.net/