Beta calibration:

a well-founded and easily implemented improvement on logistic calibration for binary classifiers

Meelis Kull, Telmo Silva Filho, Peter Flach

AISTATS 2017

University of BRISTOL

TARTU ÜLIKOOL 1632 UNIVERSITAS TARTUENSIS

VIRTUS IMPAVIDA

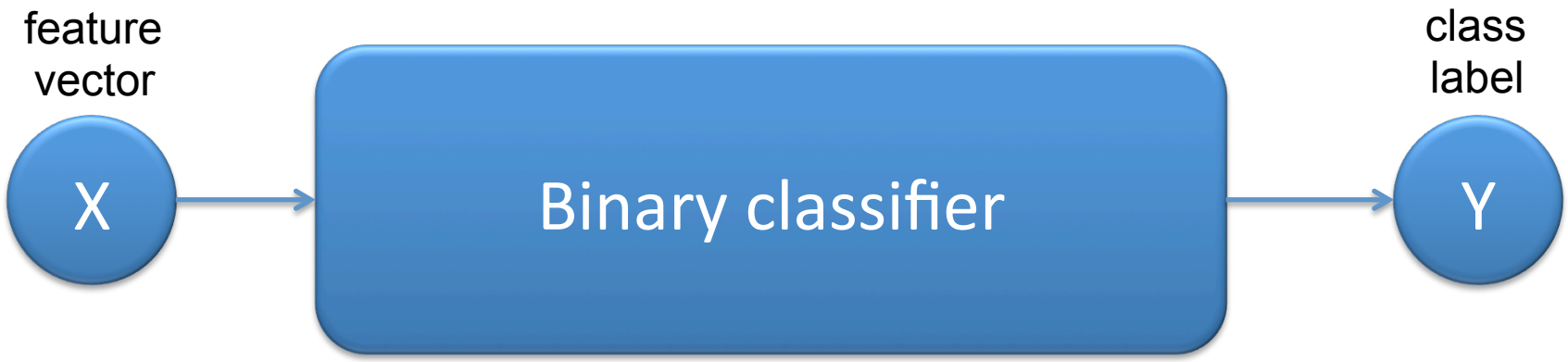
sphere

irc-sphere.ac.uk

EPSRC Engineering and Physical Sciences Research Council

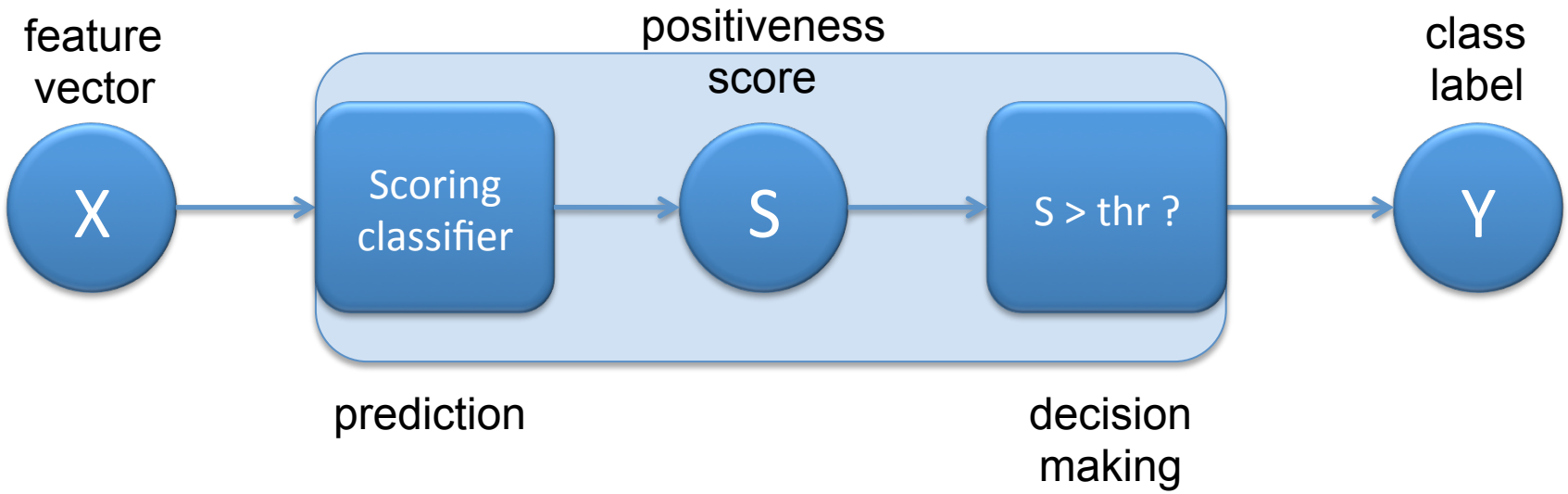
www.irc-sphere.ac.uk

Prediction confidence of binary classifiers



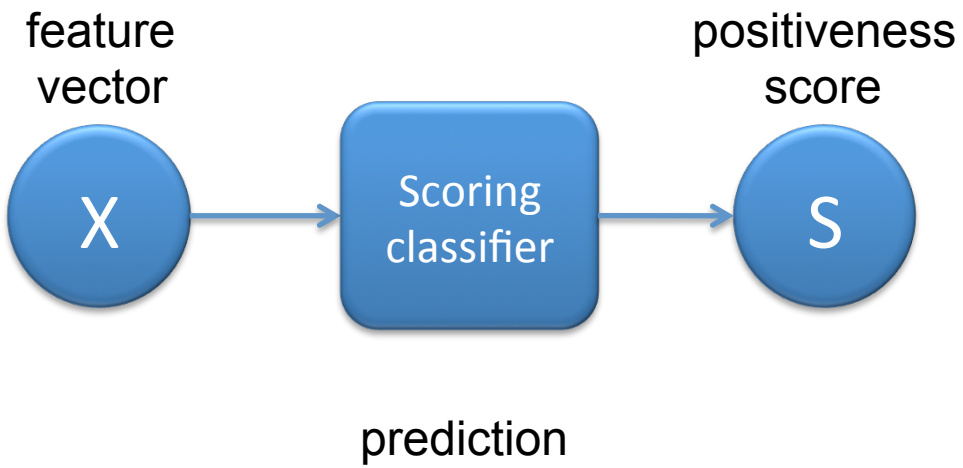
- Confidence use cases:
  - Abstaining classifier
  - Costs unknown during training
  - Human decision-maker
- Separation of prediction and decision-making

Score quantifies confidence in label



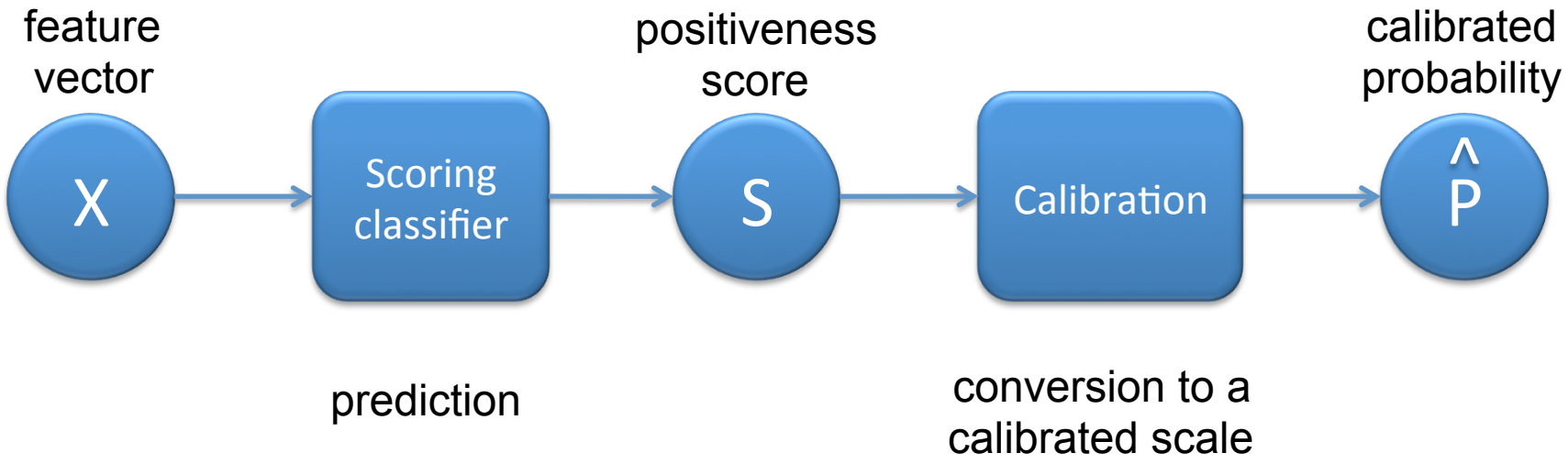
- Higher positiveness score S means more likely Y=1
  - SVM: signed distance to the decision boundary
  - Neural net: positive class probability from softmax
  - Naïve Bayes: positive class probability

Score quantifies confidence in label



- Higher positiveness score S means more likely Y=1
  - SVM: signed distance to the decision boundary
  - Neural net: positive class probability from softmax
  - Naïve Bayes: positive class probability
- Problem: the scale of S is model-dependent
  - How many positives among instances with S=0.8 ?

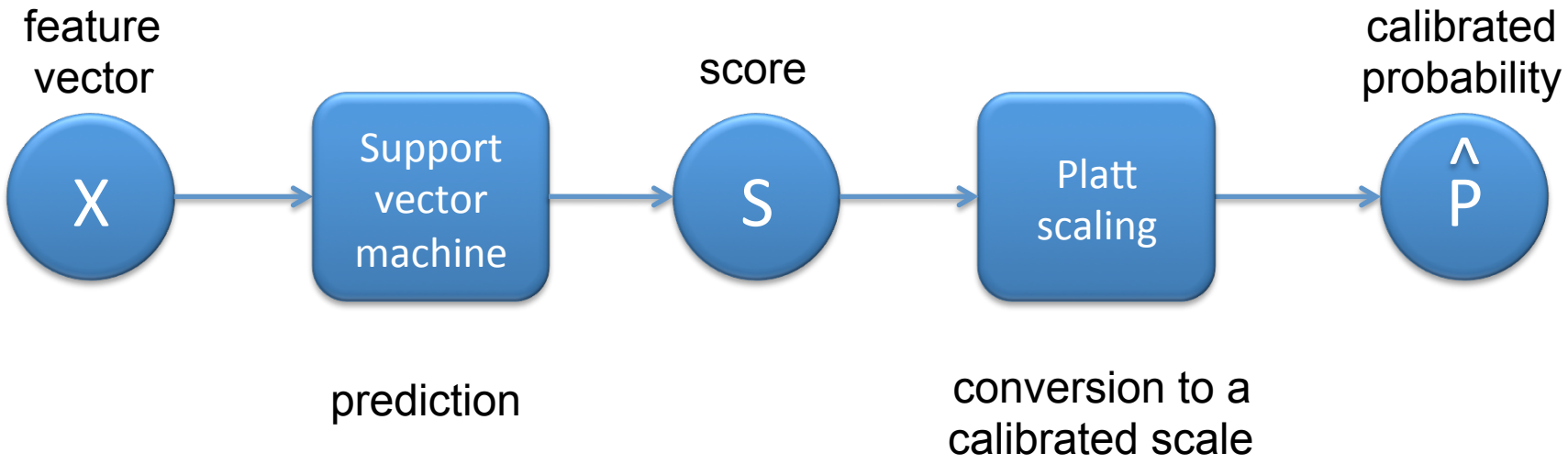
Calibration converts to a meaningful scale



- Calibrated probabilities match with expected frequencies
  - E.g. among $\hat{P}=0.8$ there are 80% positive, 20% negative
  - Calibrated on the training data vs on the test data
- Perfect calibration map: $\hat{P}=P(Y=1|S)$
- Bayes optimal: $\hat{P}=P(Y=1|X)$

Logistic calibration (a.k.a. Platt scaling)

J. Platt (2000). Probabilities for SV machines.
In A. Smola, P. Bartlett, B. Schölkopf, D. Schuurmans (Eds.),
Advances in Large Margin Classifiers, pp.61–74, MIT Press.

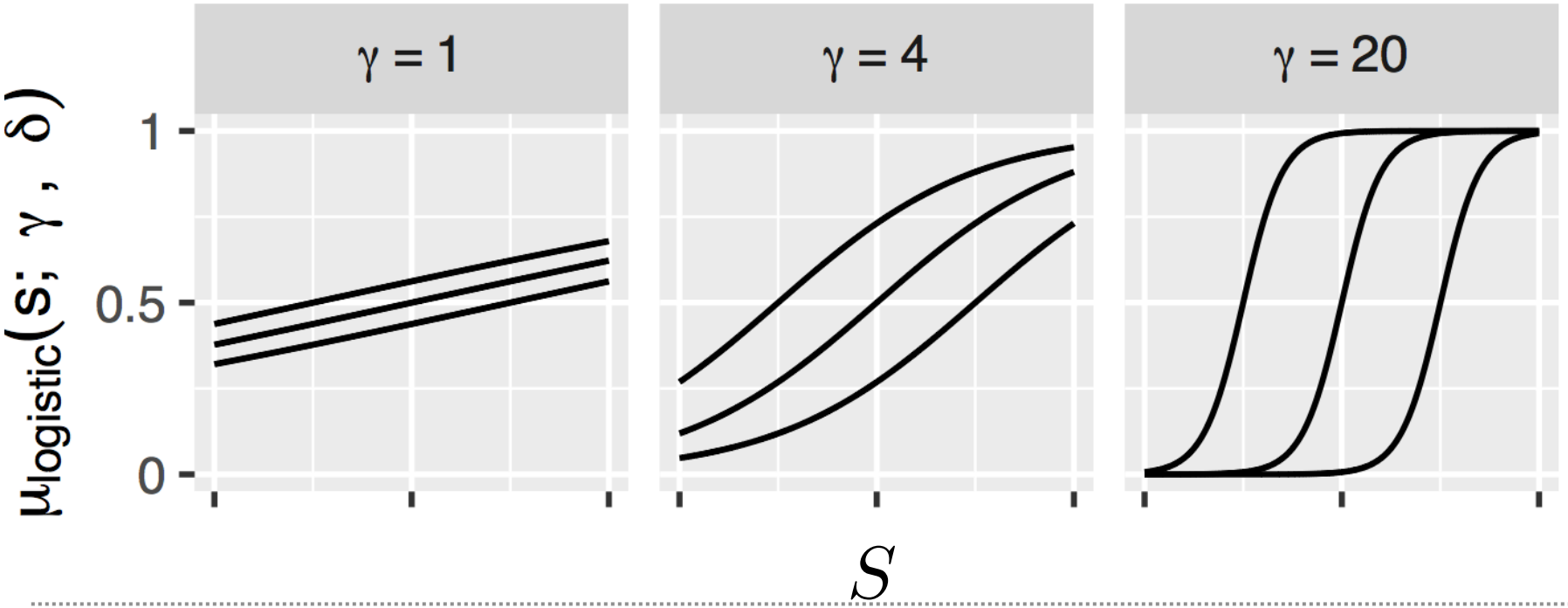


- Logistic sigmoid to convert $[-\infty, +\infty]$ to $[0, 1]$

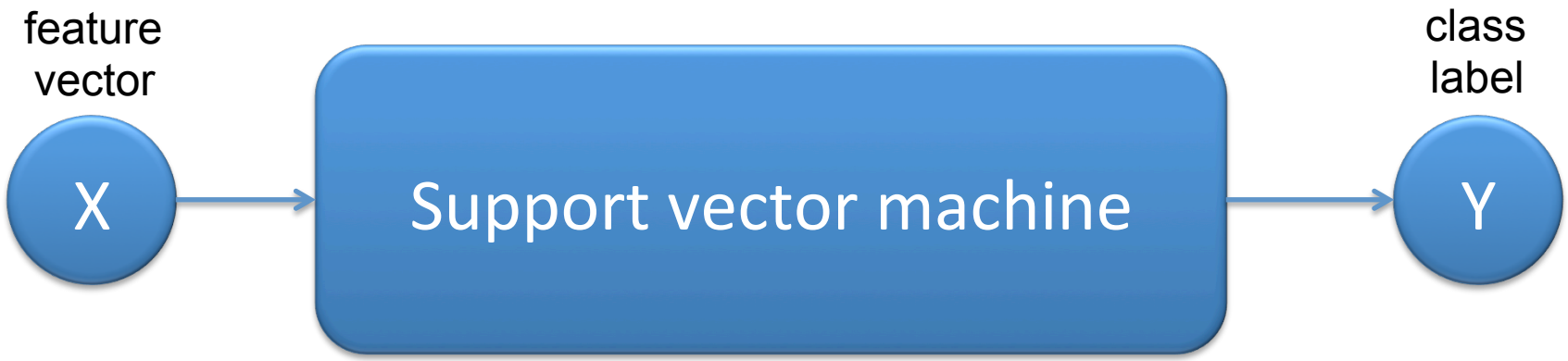
$$\mu_{logistic}(S; \gamma, \delta) = \frac{1}{1 + 1/(\exp(\gamma \cdot S + \delta))}$$

Logistic calibration map family

$$\mu_{logistic}(S;\gamma,\delta) = \frac{1}{1+1/(\exp(\gamma\cdot S+\delta))}$$

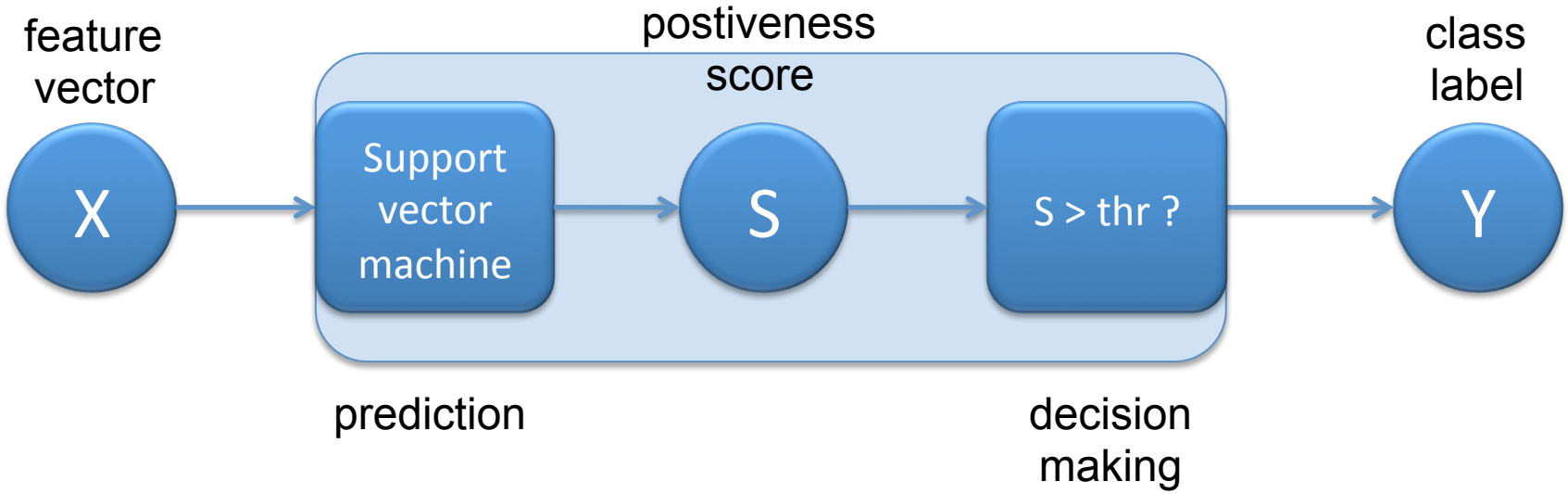


How to fit logistic calibration

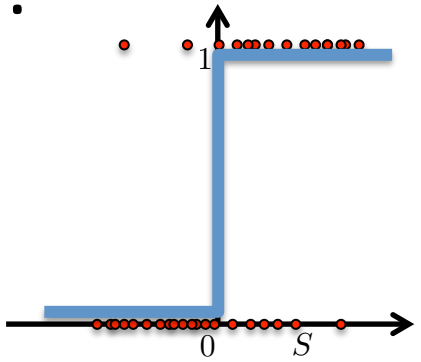


- How to get probabilities from SVM?

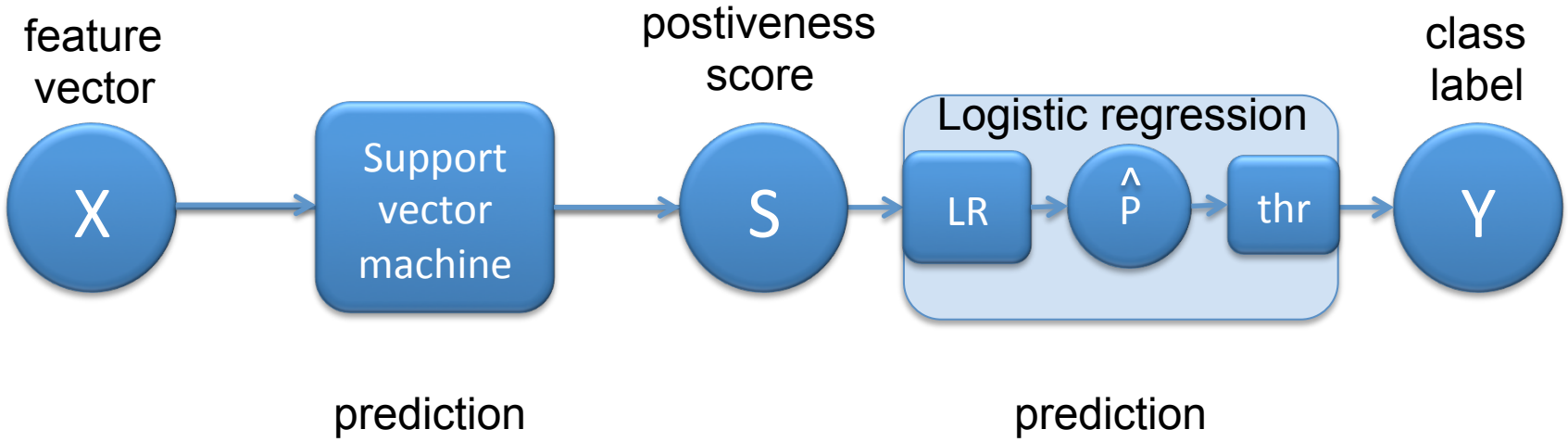
How to fit logistic calibration



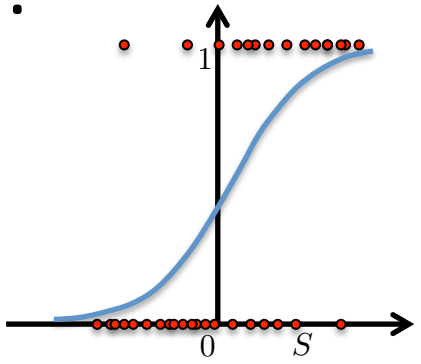
- How to get probabilities from SVM?
  1. Get the positiveness scores S



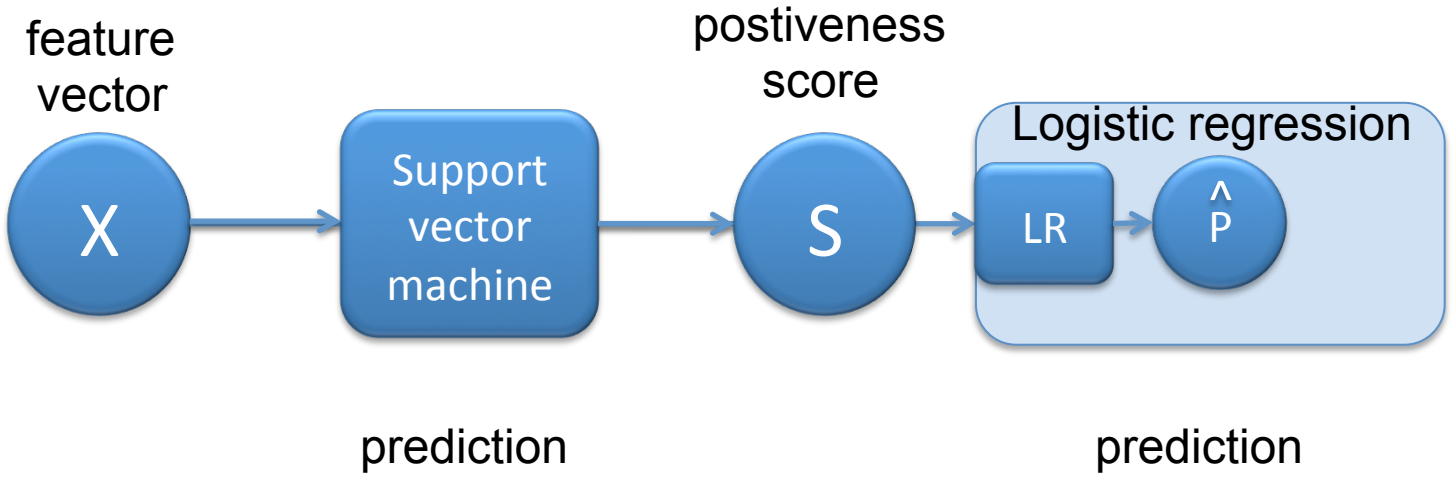
How to fit logistic calibration



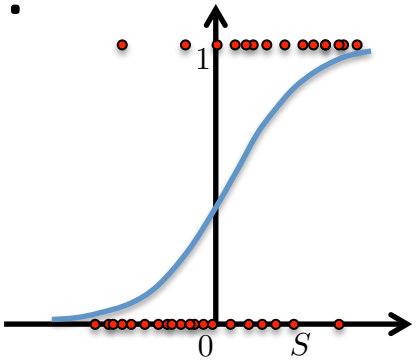
- How to get probabilities from SVM?
  1. Get the positiveness scores S
  2. Replace fixed step function by logistic regression fitting



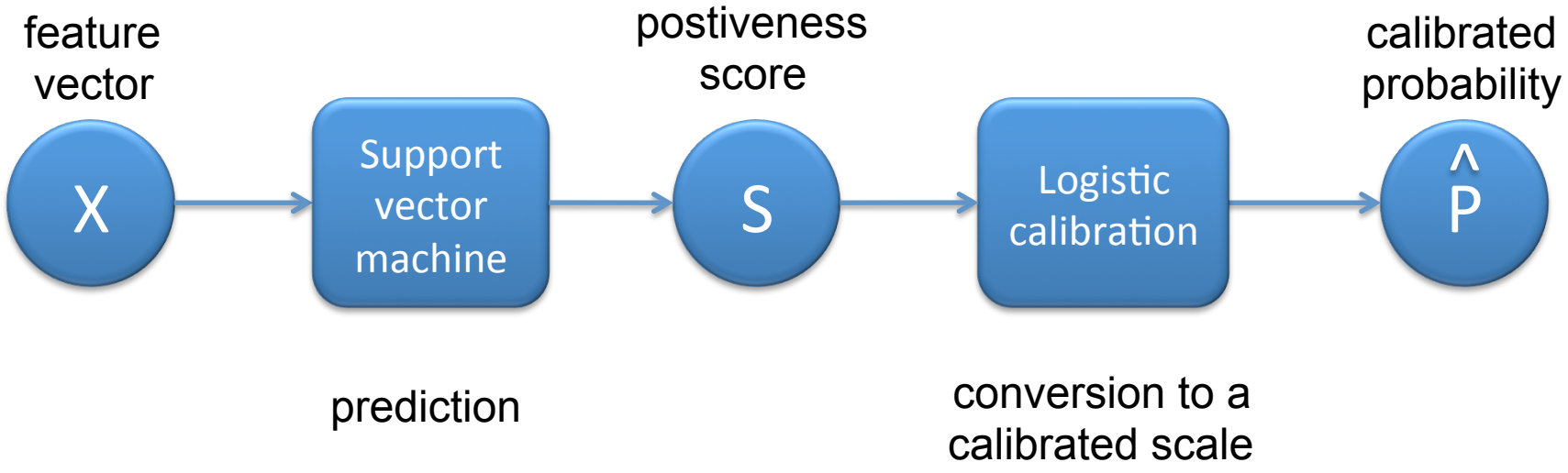
How to fit logistic calibration



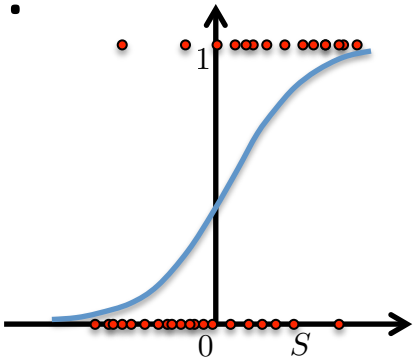
- How to get probabilities from SVM?
  1. Get the positiveness scores S
  2. Replace fixed step function by logistic regression fitting



How to fit logistic calibration



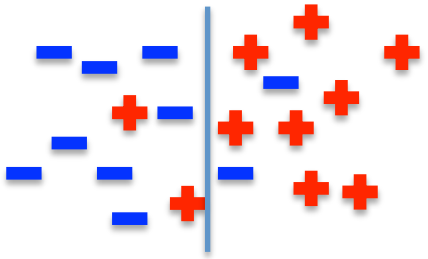
- How to get probabilities from SVM?
  1. Get the positiveness scores S
  2. Replace fixed step function by logistic regression fitting

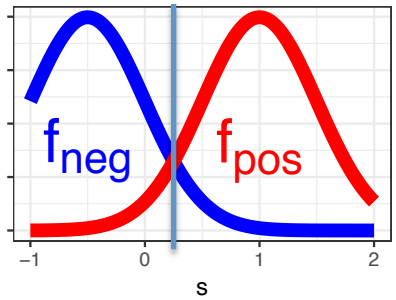


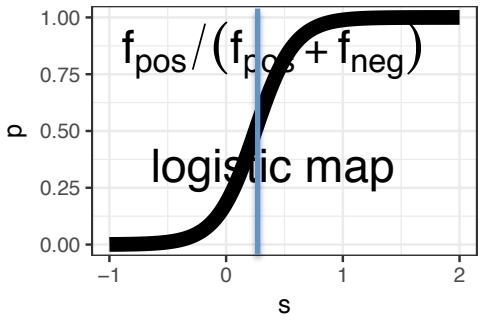
Logistic calibration is perfect if:

- Assume:
  - $p(S \mid Y=+1) \sim \mathcal{N}(\mu_{pos}, \sigma^2)$
  - $p(S \mid Y=-1) \sim \mathcal{N}(\mu_{neg}, \sigma^2)$
  - $p(Y=+1) = p(Y=-1) = 0.5$
- Then:
  - $p(Y=+1 \mid S) = \dfrac{f_{pos}}{f_{pos} + f_{neg}}$

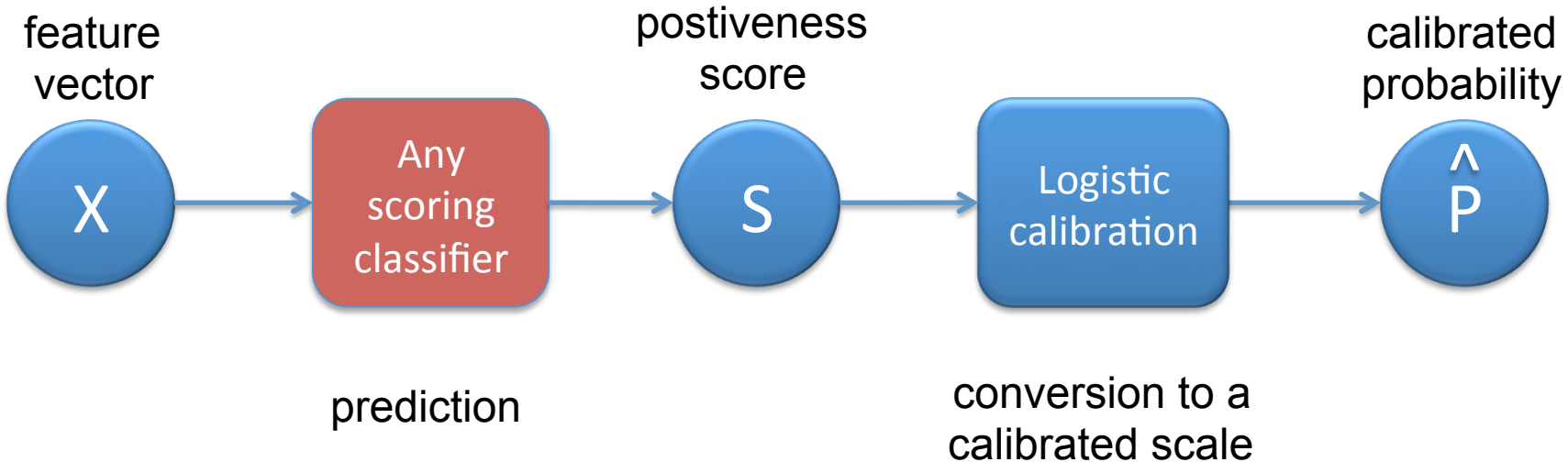
$$= \frac{1}{1 + 1/(\exp(\gamma \cdot S + \delta))}$$





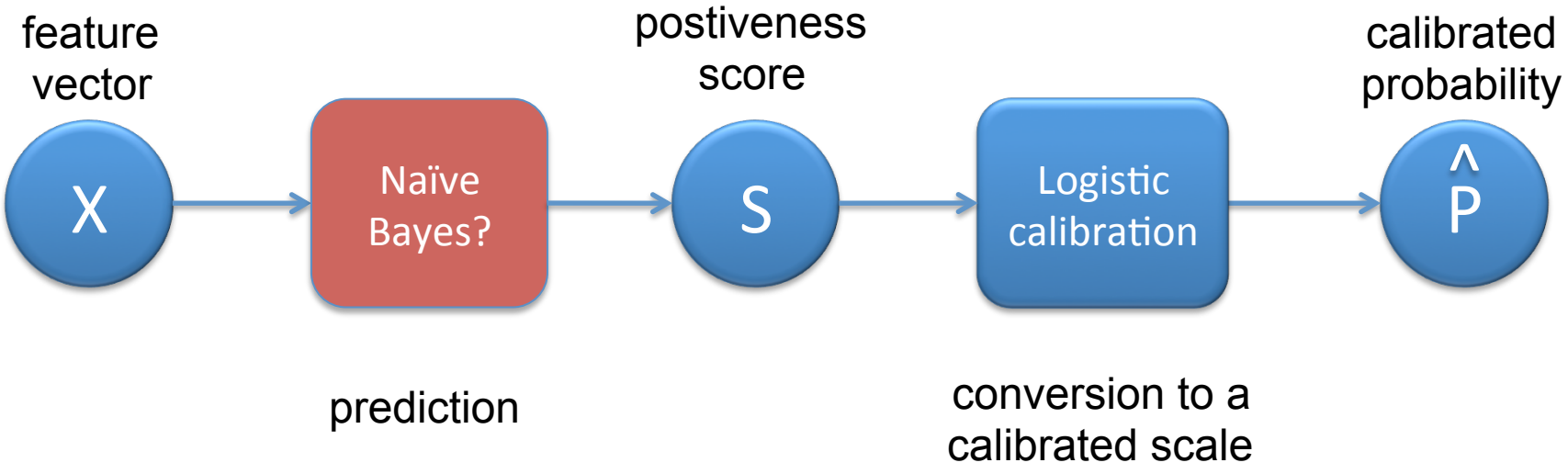


Logistic calibration is widespread

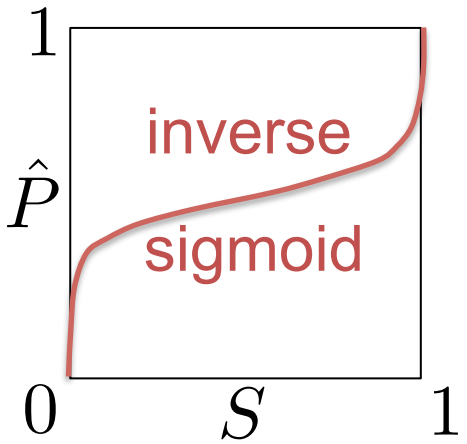


- If scores are already probabilities
  - Then Gaussian assumption questionable

Logistic calibration after Naïve Bayes?



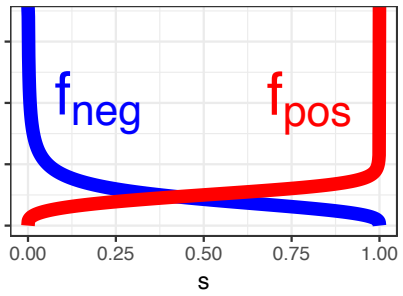
- If scores are already probabilities
  - Then Gaussian assumption questionable
- Naïve Bayes pushes probabilities towards extremes
  - Particularly with many correlated features

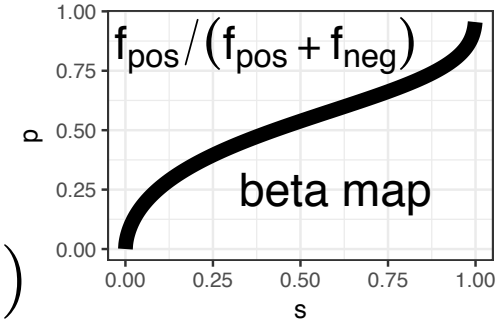


Beta calibration from first principles:

- Assume:
  - $p(S \mid Y=+1) \sim Beta(\alpha_{pos}, \beta_{pos})$
  - $p(S \mid Y=-1) \sim Beta(\alpha_{neg}, \beta_{neg})$
  - $p(Y=+1) = p(Y=-1) = 0.5$
- Then:
  - $p(Y=+1 \mid S) = \dfrac{f_{pos}}{f_{pos} + f_{neg}}$

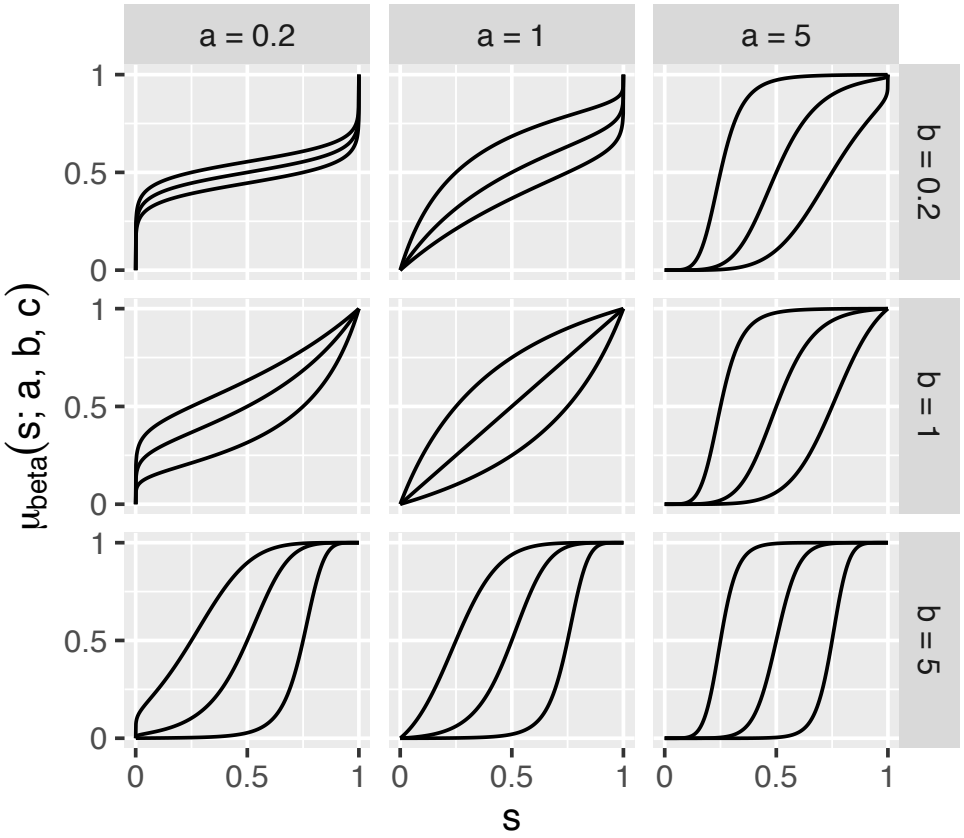
$$= \frac{1}{1 + 1/(e^c \frac{S^a}{(1-S)^b})} =: \mu_{beta}(S; a, b, c)$$





Beta calibration map family

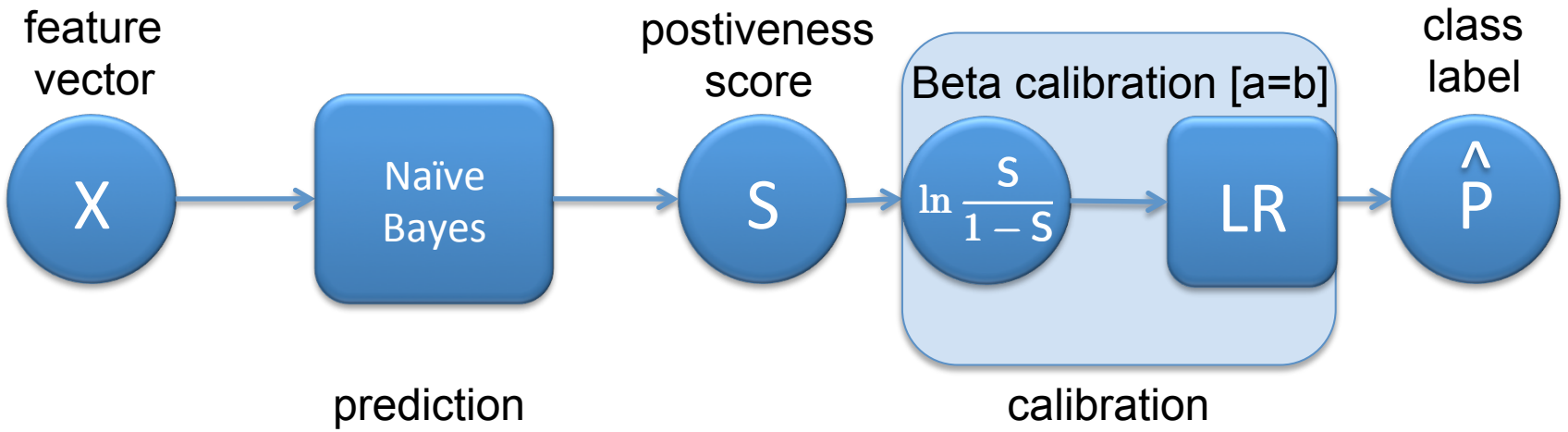
$$\mu_{beta}(S; a, b, c) = \frac{1}{1 + 1/(e^{c}\frac{S^{a}}{(1-S)^{b}})}$$



Logistic calibration vs Beta calibration

$$\mu_{logistic}(S;\gamma,\delta) = \frac{1}{1+1/(\exp(\gamma\cdot S+\delta))} \qquad \mu_{beta}(S;a,b,c) = \frac{1}{1+1/(e^{c}\frac{S^{a}}{(1-S)^{b}})}$$

$$\mu_{logistic}(\ln\frac{S}{1-S};a,c) = \mu_{beta}(S;a,a,c)$$



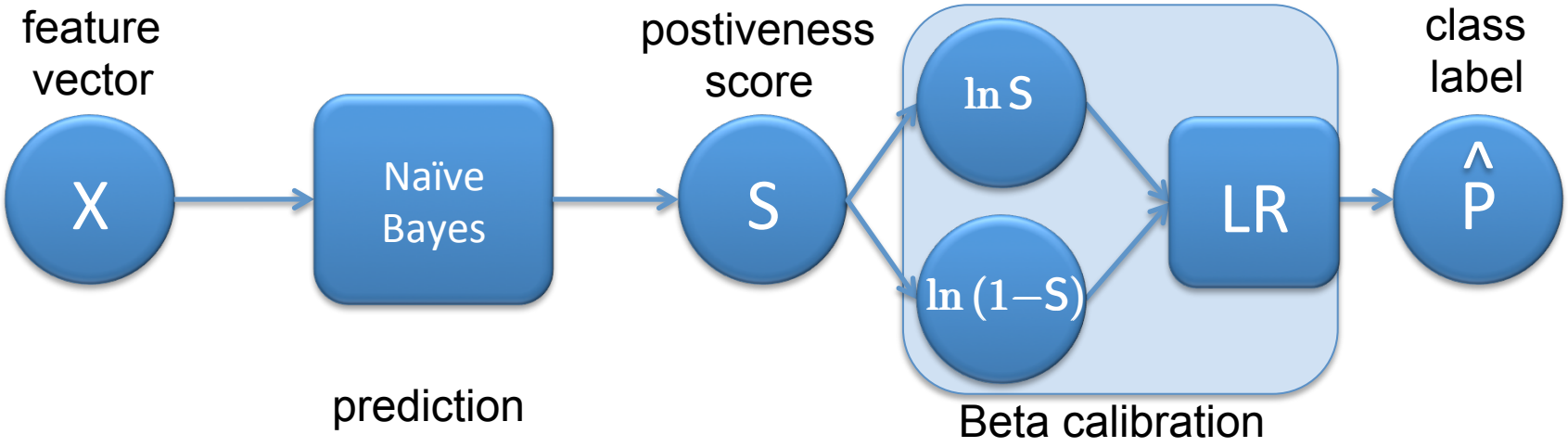
- 2-parameter beta calibration can be implemented as logistic calibration of the log odds

Beta calibration via logistic regression

$$\mu_{logreg}(S', S''; a, b, c) = \frac{1}{1 + 1/(\exp(a \cdot S' + b \cdot S'' + c))}$$

$$\mu_{beta}(S; a, b, c) = \frac{1}{1 + 1/(e^c \frac{S^a}{(1-S)^b})}$$

$$\mu_{logreg}(\ln S, -\ln(1-S); a, b, c) = \mu_{beta}(S; a, b, c)$$



- 3-parameter beta calibration can be implemented as logistic regression on features $\ln S$ and $\ln(1-S)$

Experimental setup

- 41 UCI datasets
- 10x5-fold cross-validation
- Calibration with 3-fold internal cross-validation, predictions averaged [same as Platt 2000]
- Models:
  - NB : Naïve Bayes
  - Ada-O : Adaboost original
  - Ada-S : Adaboost in Scikit (SAMME)
- Evaluation measures: proper scoring rules
  - Brier score (squared error) $BS = \mathbb{E}[(\hat{P} - \mathbb{1}_{Y=1})^2]$
  - Log-loss (cross-entropy) $LL = \mathbb{E}[-\ln(\hat{P})\,\mathbb{1}_{Y=1} - \ln(1-\hat{P})\,\mathbb{1}_{Y=-1}]$

Results (Adaboost-original, log-loss)

Table 2: Log-loss results with Ada-O classifier. Best results are marked in **bold** and subscripts indicate the ranks (before rounding to 3 decimal digits).

dataset	uncalibrated	beta	isotonic	logistic
abalone	0.612_2	$\mathbf{0.612}_1$	0.626_4	0.614_3
autos	0.427_4	$\mathbf{0.283}_1$	0.416_3	0.287_2
balance	0.049_4	$\mathbf{0.037}_1$	0.041_3	0.038_2
car	0.114_2	$\mathbf{0.109}_1$	0.122_4	0.119_3
clevela	0.496_3	$\mathbf{0.417}_1$	0.523_4	0.429_2
credit-	0.360_3	0.341_2	0.444_4	$\mathbf{0.338}_1$
dermato	0.036_4	$\mathbf{0.019}_1$	0.035_3	0.021_2
diabete	0.506_3	$\mathbf{0.484}_1$	0.525_4	0.495_2
ecoli	0.326_4	$\mathbf{0.145}_1$	0.224_3	0.159_2
flare	$\mathbf{0.404}_1$	0.404_2	0.421_4	0.408_3
german	0.511_3	0.506_2	0.538_4	$\mathbf{0.506}_1$
glass	0.696_4	0.489_2	0.560_3	$\mathbf{0.485}_1$
heart-s	0.570_4	$\mathbf{0.427}_1$	0.557_3	0.443_2
hepatit	0.805_4	$\mathbf{0.392}_1$	0.431_3	0.411_2
horse	0.642_4	$\mathbf{0.413}_1$	0.524_3	0.418_2
ionosph	0.381_4	$\mathbf{0.203}_1$	0.296_3	0.227_2
iris	0.127_4	0.000_2	$\mathbf{0.000}_1$	0.000_3
landsat	0.041_2	$\mathbf{0.040}_1$	0.043_3	0.053_4
letter	0.037_3	0.035_2	$\mathbf{0.035}_1$	0.044_4
libras-	0.589_4	$\mathbf{0.100}_1$	0.184_3	0.112_2
lung-ca	0.193_4	$\mathbf{0.139}_1$	0.189_3	0.139_2
mfeat-k	0.062_4	$\mathbf{0.027}_1$	0.048_3	0.032_2
mfeat-m	0.040_4	$\mathbf{0.014}_1$	0.026_3	0.014_2
mfeat-z	0.095_4	$\mathbf{0.032}_1$	0.063_3	0.039_2
mushroo	0.000_4	0.000_3	0.000_2	$\mathbf{0.000}_1$
optdigi	0.035_2	$\mathbf{0.033}_1$	0.044_4	0.040_3
page-bl	0.093_2	$\mathbf{0.089}_1$	0.098_3	0.109_4
pendigi	0.017_2	$\mathbf{0.017}_1$	0.023_4	0.021_3
scene-c	0.384_4	$\mathbf{0.364}_1$	0.376_2	0.378_3
segment	0.020_3	$\mathbf{0.010}_1$	0.023_4	0.013_2
shuttle	0.000_2	$\mathbf{0.000}_1$	0.001_4	0.000_3
sonar	0.941_4	$\mathbf{0.404}_1$	0.486_3	0.440_2
spambas	0.162_2	$\mathbf{0.162}_1$	0.173_4	0.167_3
tic-tac	0.379_4	$\mathbf{0.333}_1$	0.340_3	0.339_2
vehicle	0.077_2	$\mathbf{0.068}_1$	0.131_4	0.077_3
vowel	0.076_2	$\mathbf{0.071}_1$	0.094_4	0.085_3
wavefor	0.253_2	$\mathbf{0.252}_1$	0.264_3	0.271_4
wdbc	0.256_4	$\mathbf{0.089}_1$	0.141_3	0.107_2
wpbc	1.016_4	0.500_2	$\mathbf{0.496}_1$	0.503_3
yeast	0.510_2	$\mathbf{0.509}_1$	0.543_4	0.514_3
zoo	0.132_4	0.013_3	$\mathbf{0.013}_1$	0.013_2
rank	3.20	1.27	3.12	2.41

Key findings

- Beta calibration better than logistic calibration:

	Naïve Bayes	Adaboost-O	Adaboost-S
Brier score	++	++	+
Log-loss	++	++	+

- Beta calibration better than isotonic calibration in log-loss:

	Naïve Bayes	Adaboost-O	Adaboost-S
Brier score	+	+	-
Log-loss	++	++	++

- Full 3-parameter version better than 2-parameter versions (but often non-significantly)

Availability

- https://betacal.github.io
  - Python package
  - R package
- Or, implement your own as logistic regression on the dataset with two features: $\ln s$ and $\ln(1-s)$

Next: Dirichlet calibration for multi-class

- Dirichlet distribution is the multi-class generalisation of beta distribution
- Beta calibration can also be generalised in a relatively straight-forward manner
- https://dircal.github.io (paper in preparation)
- Or, implement yourself as logistic regression on features $\ln p_1, \ln p_2, \ldots, \ln p_k$ where k is the number of classes

Summary

- Scoring classifiers
  - Allow for separation of prediction and decision making
- Calibrated probabilities
  - Provide a standard scale of scores
- Logistic calibration
  - Is perfect when score conditional densities are Gaussian
- Beta calibration
  - Is perfect when score conditional densities are Beta
- Beta calibration improves logistic calibration if scores are in [0,1] range :
  - Well-founded, easily implemented, better performance
  - https://betacal.github.io