

CALIBRATION

- Consider binary classification with a probabilistic score classifier

$$f(\mathbf{x}) = 2 \cdot \mathbb{1}_{[s(\mathbf{x}) \geq c]} - 1,$$

leading to the prediction random variable $\hat{y} = f(\mathbf{x})$. Let $\mathbf{S} = s(\mathbf{x})$ be the score random variable.

- f is calibrated iff $P(y = 1 \mid \mathbf{S} = s) = s$ for all $s \in [0, 1]$.
- Different *post-processing* methods have been proposed for the purpose of calibration, i.e., to construct a *calibration function*

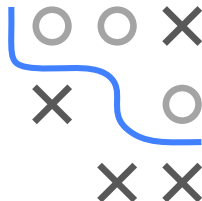
$$C : \mathbb{S} \rightarrow [0, 1],$$

such that $C(s(\mathbf{x}))$ is well-calibrated. Here, \mathbb{S} is the possible score set of the classifier (the image of s).

- For learning C , a set of *calibration data* is used:

$$\mathcal{D}_{cal} = \{(s^{(1)}, y^{(1)}), \dots, (s^{(N)}, y^{(N)})\} \subset \mathbb{S} \times \{-1, 1\}$$

- This data should be different from the training data used to learn the scoring classifier. Otherwise, there is a risk of introducing a bias.



EMPIRICAL BINNING AND PLATT SCALING

- *Binning* offers a first obvious approach: Partition \mathbb{S} into bins (intervals) B_1, \dots, B_M , and define $C(s) = \bar{p}_{J(s)}$, where $J(s)$ denotes the index of the bin of s (i.e., $s \in B_{J(s)}$), and

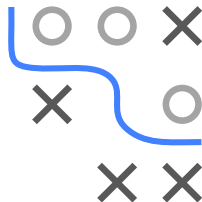
$$\bar{p}_m = \frac{\sum_{n=1}^N \mathbb{1}_{[s^{(n)} \in B_m, y^{(n)} = +1]}}{\sum_{n=1}^N \mathbb{1}_{[s^{(n)} \in B_m]}}$$

is the average proportion of positives in bin B_m .

- Another method is *Platt scaling*, which essentially applies logistic regression to predicted scores $s \in \mathbb{R}$, i.e., it fits a calibration function C such that

$$C(s) = \frac{1}{1 + \exp(\gamma + \theta \cdot s)},$$

minimizing log-loss on \mathcal{D}_{cal} .



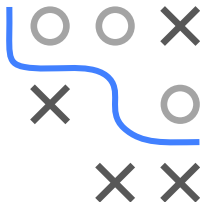
ISOTONIC REGRESSION

- The sigmoidal transformation fit by Platt scaling is appropriate for some methods (e.g., support vector machines) but not for others.
- *Isotonic regression* combines the nonparametric character of binning with Platt scaling's guarantee of monotonicity.
- Isotonic regression minimizes

$$\sum_{n=1}^N w_n (C(s^{(n)}) - y^{(n)})^2$$

subject to the constraint that C is isotonic: $C(s) \leq C(t)$ for $s < t$.

- Note that C is evaluated only at a finite number of points; in-between, one may (linearly) interpolate or assume a piecewise constant function.



PAIR-ADJACENT VIOLATORS ALGORITHM (PAVA)

- Let the scores observed for calibration be sorted (and without ties), such that

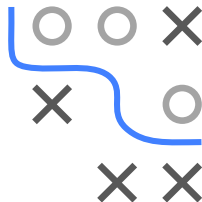
$$s^{(1)} < s^{(2)} < \dots < s^{(N)}.$$

We then seek values $c_1 \leq c_2 \leq \dots \leq c_N$ which minimize

$$\sum_{n=1}^N w_n (c_n - y^{(n)})^2.$$

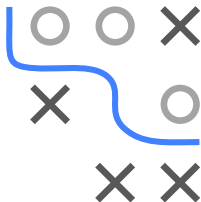
- Initialize one block B_n for each observation $(s^{(n)}, y^{(n)})$; the value of the block is $c(B_n) = y^{(n)}$ and the width is $w(B_n) = 1$.
- A merge operation combines two blocks B' and B'' into a new block B with width $w(B) = w(B') + w(B'')$ and value

$$c = \frac{w(B')c(B') + w(B'')c(B'')}{w(B') + w(B'')}.$$

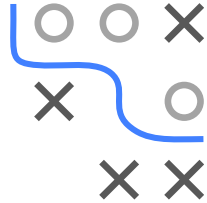
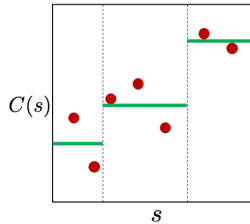
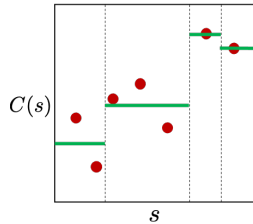
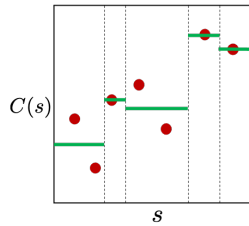
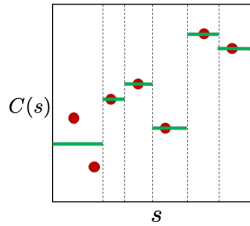
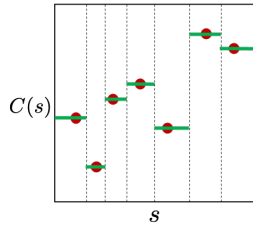


PAIR-ADJACENT VIOLATORS ALGORITHM (PAVA)

- PAVA iterates the following steps (the description is somewhat simplified to avoid notational overload):
 - (1) Find the first violating pair, namely, adjacent blocks B_i and B_{i+1} such that $c_i > c_{i+1}$; if there is no such pair, then stop.
 - (2) Merge B_i and B_{i+1} into a new block B .
 - (3) If $c(B) < c(B_{i-1})$ for the left neighbor block B_{i-1} , merge also these blocks and continue doing so until no more violations are encountered.
 - (4) Continue with (1).



PAIR-ADJACENT VIOLATORS ALGORITHM (PAVA)



PAIR-ADJACENT VIOLATORS ALGORITHM (PAVA)

- Note that, in the case of binary classification, the target values $y^{(n)}$ are all in $\{0, 1\}$:

