

Optimizing ROC Curves with a Sort-Based Surrogate Loss for Binary Classification and Changepoint Detection, arXiv:2107.01285

Toby Dylan Hocking — toby.hocking@nau.edu
joint work with my student Jonathan Hillman
Machine Learning Research Lab — <http://ml.nau.edu>



Come to SICCS! Graduate Research Assistantships available!

Problem Setting and Related Work

Proposed algorithm

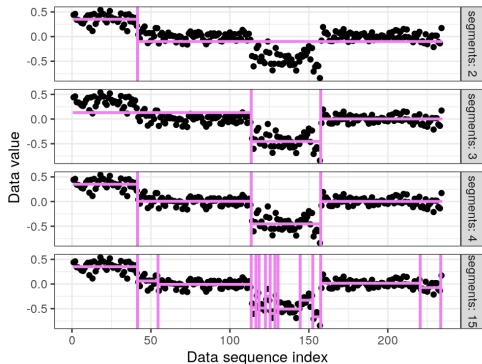
Empirical results

Discussion and Conclusions

Problem: unsupervised changepoint detection

- ▶ We are given a data sequence z_1, \dots, z_T measured at T points over time/space.
- ▶ Ex: DNA copy number data for cancer diagnosis, $z_t \in \mathbb{R}$.
- ▶ The penalized changepoint problem is

$$\arg \min_{u_1, \dots, u_T \in \mathbb{R}} \sum_{t=1}^T (u_t - z_t)^2 + \lambda \sum_{t=2}^T I[u_{t-1} \neq u_t].$$

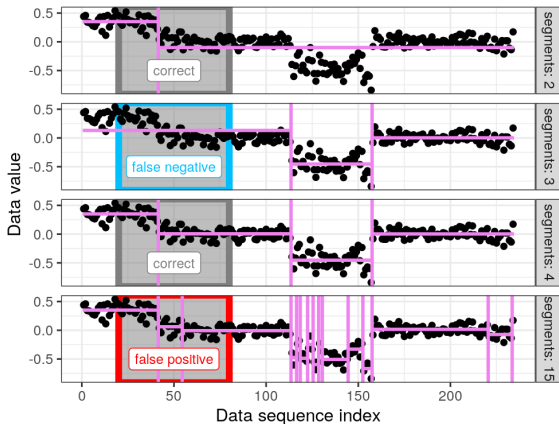


Larger penalty λ results in fewer changes/segments.

Smaller penalty λ results in more changes/segments.

Problem: weakly supervised changepoint detection

- ▶ We are given a data sequence \mathbf{z} with labeled regions L .
- ▶ We compute features $\mathbf{x} = \phi(\mathbf{z}) \in \mathbf{R}^p$ and want to learn a function $f(\mathbf{x}) = -\log \lambda \in \mathbf{R}$ that minimizes label error.

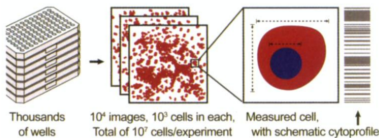


Problem: supervised binary classification

- ▶ Given pairs of inputs $\mathbf{x} \in \mathbb{R}^p$ and outputs $y \in \{0, 1\}$ can we learn $f(\mathbf{x}) = y$?
- ▶ Example: email, \mathbf{x} = bag of words, y = spam or not.
- ▶ Example: images. Jones *et al.* PNAS 2009.

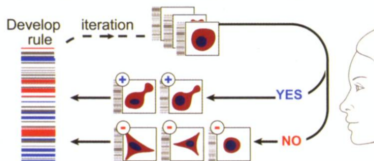
A Automated Cell Image Processing

Cytoprofile of 500+ features measured for each cell



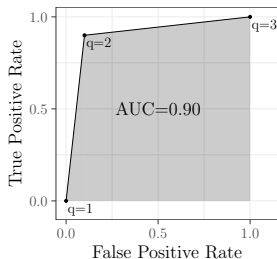
B Iterative Machine Learning

System presents cells to biologist for scoring, in batches



Receiver Operating Characteristic (ROC) curve

- ▶ Binary classification algo gives predictions $[\hat{y}_1, \hat{y}_2, \hat{y}_3, \hat{y}_4]$.
- ▶ Each point on the ROC curve is the FPR/TPR if you add some constant c to the predictions, $[\hat{y}_1 + c, \hat{y}_2 + c, \hat{y}_3 + c, \hat{y}_4 + c]$.
- ▶ Best point in ROC space is upper left (0% FPR, 100% TPR).
- ▶ Maximizing Area Under the ROC curve (AUC) is a common objective for binary classification, especially for imbalanced data (example: 99% positive, 1% negative labels).

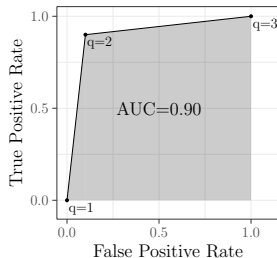
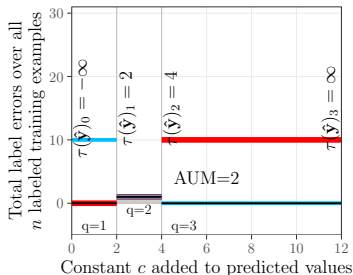


In binary classification, ROC curve is monotonic increasing.

- ▶ AUC=1 best.
- ▶ AUC=0.5 for constant prediction (usually worst).

Area Under ROC curve, synthetic example

- ▶ label = $[1, 0, 0, \dots, 1, 1, 0]$ (20 labels, 10 positive, 10 negative).
- ▶ predictions = $[-4, -4, -4, \dots, -2, -2, -2]$.
- ▶ No constant added $c = 0$, $q = 1$, everything predicted negative, so no false positives, but no true positives.
- ▶ Add $c = 3 \Rightarrow [-1, -1, -1, \dots, 1, 1, 1]$, 1 FP and 9 TP, $q = 2$.
- ▶ Add $c = 5 \Rightarrow [1, 1, 1, \dots, 3, 3, 3]$, all FP and TP, $q = 3$.



Problem Setting and Related Work

Proposed algorithm

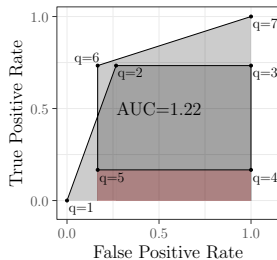
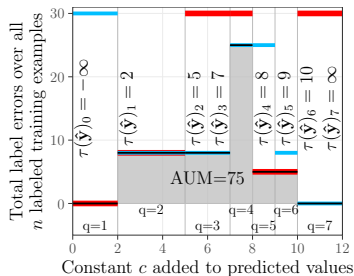
Empirical results

Discussion and Conclusions

Looping ROC curve, simple synthetic example

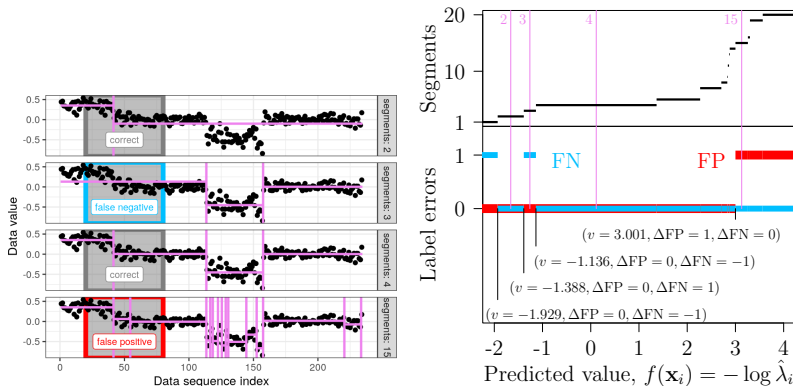
If ROC curve has loops, AUC can be greater than one.

- ▶ Dark grey area double counted.
- ▶ Red area negative counted.
- ▶ Do we want to maximize AUC?
- ▶ Minimize Area Under Min (AUM) instead, which encourages ROC points in upper left.



Real data example with non-monotonic label error

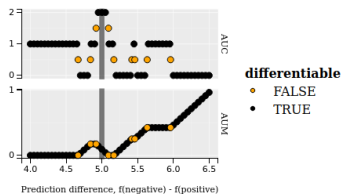
ROC curve loops result from non-monotonic FP/FN functions, but do these occur in real data? Yes, in supervised changepoint problems.



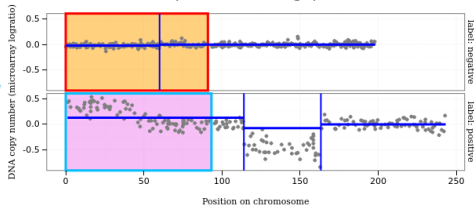
Optimal changepoint model may have non-monotonic error (for example FN), because changepoints at model size s may not be present in model $s + 1$.

Real data example, interactive AUC/AUM demo

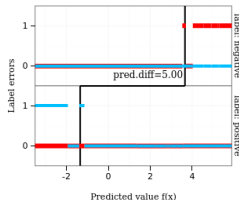
Overview, select difference



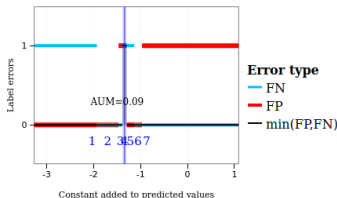
Data, labels, predicted changepoint models



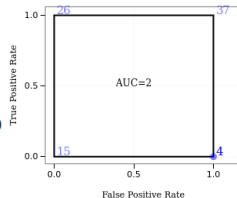
Example error functions



Total error, select interval

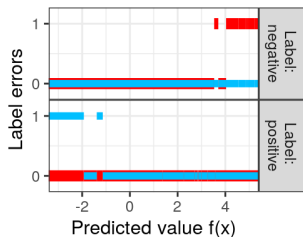


ROC curve, select point



<http://bl.ocks.org/tdhock/raw/de3979318d5255dd6e21ff907e2f3fb4/>

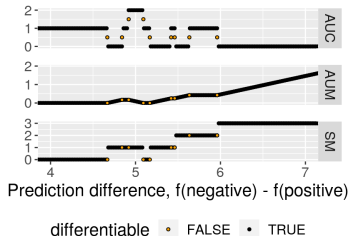
Real data example with AUC greater than one



Error type

FN

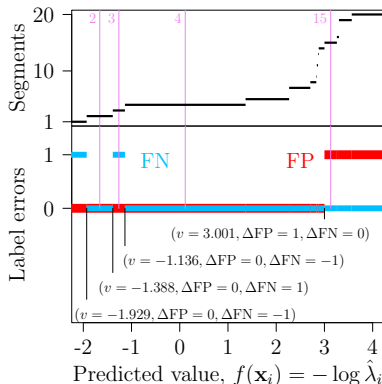
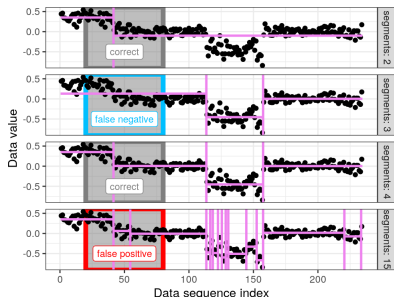
FP



- ▶ $n = 2$ labeled changepoint problems.
- ▶ $\text{AUC} = 2$ when prediction difference = 5.
- ▶ $\text{AUM} = 0$ implies $\text{AUC} = 1$.
- ▶ AUM is convex relaxation of non-convex Sum of Min (SM).
- ▶ AUM is differentiable almost everywhere.
- ▶ Main new idea: compute the gradient of this function and use it for learning.

Algorithm inputs: predictions and label error functions

- ▶ Each observation $i \in \{1, \dots, n\}$ has a predicted value $\hat{y}_i \in \mathbb{R}$.
- ▶ Breakpoints $b \in \{1, \dots, B\}$ used to represent label error via tuple $(v_b, \Delta FP_b, \Delta FN_b, \mathcal{I}_b)$.
- ▶ There are changes $\Delta FP_b, \Delta FN_b$ at predicted value $v_b \in \mathbb{R}$ in error function $\mathcal{I}_b \in \{1, \dots, n\}$.

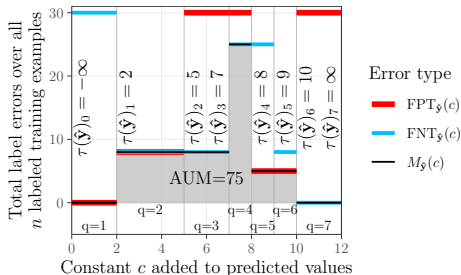


Algorithm computes total FP and FN for each threshold/constant added to predicted values

- ▶ Breakpoint threshold, $t_b = v_b - \hat{y}_{\mathcal{I}_b} = \tau(\hat{\mathbf{y}})_q$ for some q .
- ▶ Total error before/after each breakpoint can be computed via sort and modified cumsum:

$$\underline{\text{FP}}_b = \sum_{j:t_j < t_b} \Delta \text{FP}_j, \quad \overline{\text{FP}}_b = \sum_{j:t_j \leq t_b} \Delta \text{FP}_j,$$

$$\underline{\text{FN}}_b = \sum_{j:t_j \geq t_b} -\Delta \text{FN}_j, \quad \overline{\text{FN}}_b = \sum_{j:t_j > t_b} -\Delta \text{FN}_j.$$

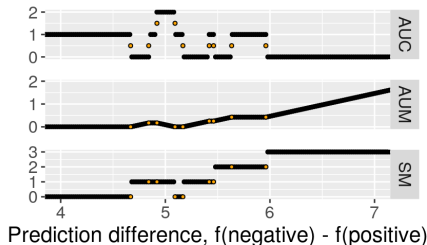


Algorithm computes two directional derivatives

- ▶ Gradient only defined when function is differentiable, but AUM is not everywhere (see below).
- ▶ Directional derivatives defined everywhere.

$$\nabla_{\mathbf{v}(-1,i)} \text{AUM}(\hat{\mathbf{y}}) = \sum_{b:\mathcal{I}_b=i} \min\{\overline{\text{FP}}_b, \overline{\text{FN}}_b\} - \min\{\overline{\text{FP}}_b - \Delta\text{FP}_b, \overline{\text{FN}}_b - \Delta\text{FN}_b\},$$

$$\nabla_{\mathbf{v}(1,i)} \text{AUM}(\hat{\mathbf{y}}) = \sum_{b:\mathcal{I}_b=i} \min\{\underline{\text{FP}}_b + \Delta\text{FP}_b, \underline{\text{FN}}_b + \Delta\text{FN}_b\} - \min\{\underline{\text{FP}}_b, \underline{\text{FN}}_b\}.$$



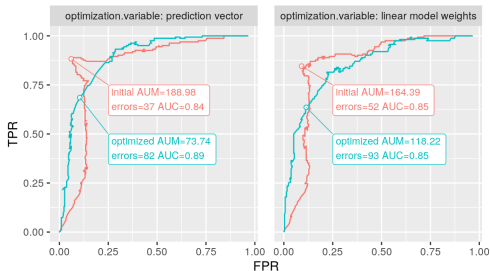
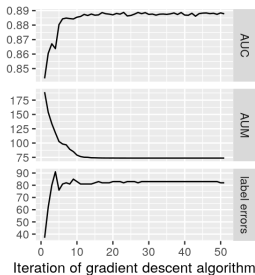
Problem Setting and Related Work

Proposed algorithm

Empirical results

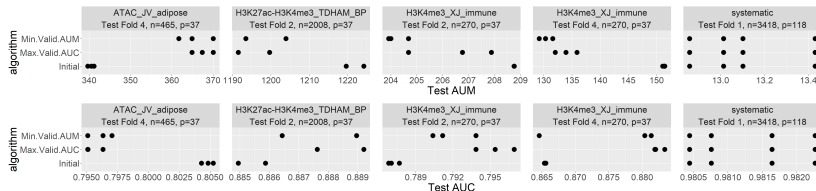
Discussion and Conclusions

Train set ROC curves for a real changepoint problem



- ▶ Left/middle: changepoint problem initialized to prediction vector with min label errors, gradient descent on prediction vector.
- ▶ Right: linear model initialized by minimizing regularized convex loss (surrogate for label error), gradient descent on weight vector.

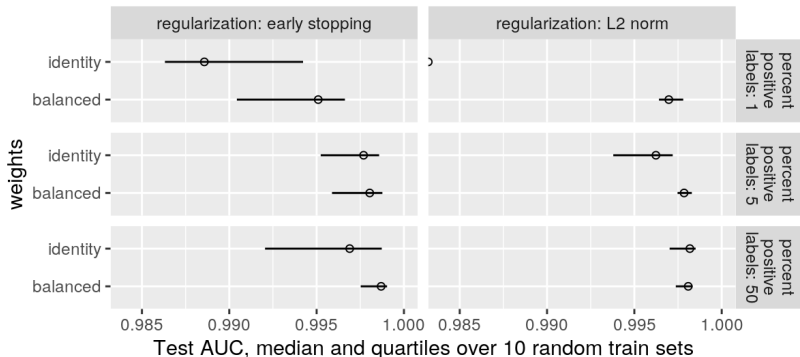
Learning algorithm results in better test AUC/AUM for changepoint problems



- ▶ Five changepoint problems (panels from left to right).
- ▶ Two evaluation metrics (AUM=top, AUC=bottom).
- ▶ Three algorithms (Y axis), Initial=Min regularized convex loss (surrogate for label error),
Min.Valid.AUM/Min.Valid.AUC=AUM gradient descent with early stopping regularization.

Standard logistic loss fails for highly imbalanced labels

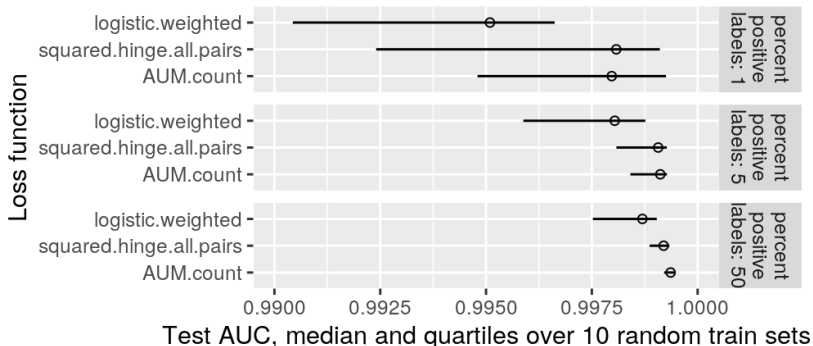
Comparing logistic regression models (control experiment)



- ▶ Test set has 50% positive, 50% negative labels.
- ▶ Train set has variable class imbalance (panels top to bottom).
- ▶ Loss is $\ell[f(x_i), y_i]w_i$ with $w_i = 1$ for identity weights, $w_i = 1/N_{y_i}$ for balanced, ex: 1% position means $w_i \in \{1/10, 1/990\}$.

Learning algorithm competitive for unbalanced binary classification

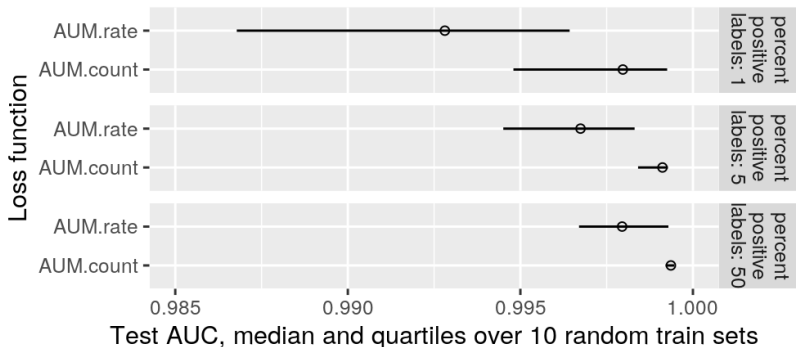
(b) AUM compared to baselines



- ▶ Squared hinge all pairs is a classic/popular surrogate loss function for AUC optimization.
- ▶ All linear models with early stopping regularization.

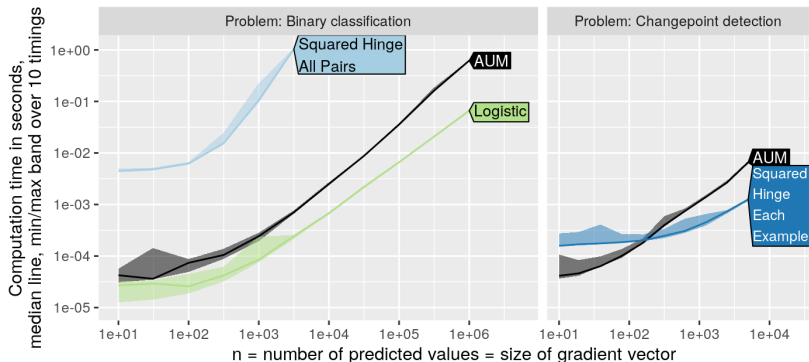
Error rate loss is not as useful as error count loss

(a) Comparing AUM variants



- ▶ AUM.count is as described previously: error functions used to compute $\text{Min}(\text{FP}, \text{FN})$ are absolute label counts.
- ▶ AUM.rate is a variant which uses normalized error functions, $\text{Min}(\text{FPR}, \text{FNR})$.
- ▶ Both linear models with early stopping regularization.

Comparable computation time to other loss functions



- ▶ Logistic $O(n)$.
- ▶ AUM $O(n \log n)$.
- ▶ Squared Hinge All Pairs $O(n^2)$.
- ▶ Squared Hinge Each Example $O(n)$.

Problem Setting and Related Work

Proposed algorithm

Empirical results

Discussion and Conclusions

Conclusions, Pre-print arXiv:2107.01285

- ▶ ROC curves are used to evaluate binary classification and changepoint detection algorithms.
- ▶ In changepoint detection there can be loops in ROC curves, so maximizing AUC may not be desirable.
- ▶ Instead we propose to minimize a new loss, $AUM = \text{Area Under Min}(FP, FN)$.
- ▶ We propose new algorithm for efficient AUM and directional derivative computation.
- ▶ Empirical results provide evidence that learning using AUM minimization results in AUC maximization.
- ▶ Future work: sort-based surrogates for all pairs loss functions (binary classification, information retrieval).

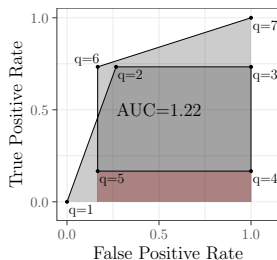
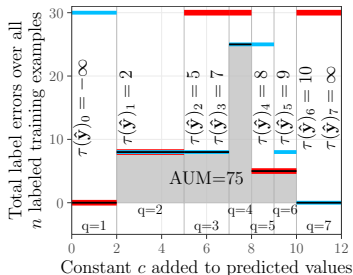
Thanks!



Contact: toby.hocking@nau.edu

More notation

- ▶ First let $\{(\text{fpt}(\hat{\mathbf{y}})_q, \text{fnt}(\hat{\mathbf{y}})_q, \tau(\hat{\mathbf{y}})_q)\}_{q=1}^Q$ be a sequence of Q tuples, each of which corresponds to a point on the ROC curve (and an interval on the fn/fp error plot).
- ▶ For each q the $\text{fpt}(\hat{\mathbf{y}})_q, \text{fnt}(\hat{\mathbf{y}})_q$ are false positive/negative totals at that point (in that interval) whereas $\tau(\hat{\mathbf{y}})_q$ is the upper limit of the interval.
- ▶ The limits are increasing, $-\infty = \tau(\hat{\mathbf{y}})_0 < \dots < \tau(\hat{\mathbf{y}})_Q = \infty$.
- ▶ Then we define $m(\hat{\mathbf{y}})_q = \min\{\text{fpt}(\hat{\mathbf{y}})_q, \text{fnt}(\hat{\mathbf{y}})_q\}$ as the min of fp and fn totals in that interval.



L1 relaxation interpretation

Our proposed loss function is

$$\text{AUM}(\hat{\mathbf{y}}) = \sum_{q=2}^{Q-1} [\tau(\hat{\mathbf{y}})_q - \tau(\hat{\mathbf{y}})_{q-1}] m(\hat{\mathbf{y}})_q.$$

It is an L1 relaxation of the following non-convex **Sum of Min(FP,FN)** function,

$$\text{SM}(\hat{\mathbf{y}}) = \sum_{q=2}^{Q-1} I[\tau(\hat{\mathbf{y}})_q \neq \tau(\hat{\mathbf{y}})_{q-1}] m(\hat{\mathbf{y}})_q = \sum_{q: \tau(\hat{\mathbf{y}})_q \neq \tau(\hat{\mathbf{y}})_{q-1}} m(\hat{\mathbf{y}})_q.$$

Definition of data set, notations

- ▶ Let there be a total of B breakpoints in the error functions over all n labeled training examples.
- ▶ Each breakpoint $b \in \{1, \dots, B\}$ is represented by the tuple $(v_b, \Delta FP_b, \Delta FN_b, \mathcal{I}_b)$, where the $\mathcal{I}_b \in \{1, \dots, n\}$ is an example index, and there are changes $\Delta FP_b, \Delta FN_b$ at predicted value $v_b \in \mathbb{R}$ in the error functions.
- ▶ For example in binary classification, there are $B = n$ breakpoints (same as the number of labeled training examples); for each breakpoint $b \in \{1, \dots, B\}$ we have $v_b = 0$ and $\mathcal{I}_b = b$. For breakpoints b with positive labels $y_b = 1$ we have $\Delta FP = 0, \Delta FN = -1$, and for negative labels $y_b = -1$ we have $\Delta FP = 1, \Delta FN = 0$.
- ▶ In changepoint detection we have more general error functions, which may have more than one breakpoint per example.

Proposed algorithm uses sort to compute AUM and directional derivatives

- 1: **Input:** Predictions $\hat{\mathbf{y}} \in \mathbb{R}^n$, breakpoints in error functions $v_b, \Delta FP_b, \Delta FN_b, \mathcal{I}_b$ for all $b \in \{1, \dots, B\}$.
- 2: Zero the $AUM \in \mathbb{R}$ and directional derivatives $\mathbf{D} \in \mathbb{R}^{n \times 2}$.
- 3: $t_b \leftarrow v_b - \hat{y}_{\mathcal{I}_b}$ for all b .
- 4: $s_1, \dots, s_B \leftarrow \text{SORTEDINDICES}(t_1, \dots, t_B)$.
- 5: Compute $\underline{FP}_b, \overline{FP}_b, \underline{FN}_b, \overline{FN}_b$ for all b using s_1, \dots, s_B .
- 6: **for** $b \in \{2, \dots, B\}$ **do**
- 7: $AUM += (t_{s_b} - t_{s_{b-1}}) \min\{\underline{FP}_b, \overline{FN}_b\}$.
- 8: **for** $b \in \{1, \dots, B\}$ **do**
- 9: $\mathbf{D}_{\mathcal{I}_b,1} += \min\{\overline{FP}_b, \overline{FN}_b\} - \min\{\overline{FP}_b - \Delta FP_b, \overline{FN}_b - \Delta FN_b\}$
- 10: $\mathbf{D}_{\mathcal{I}_b,2} += \min\{\underline{FP}_b + \Delta FP_b, \underline{FN}_b + \Delta FN_b\} - \min\{\underline{FP}_b, \underline{FN}_b\}$
- 11: **Output:** AUM and matrix \mathbf{D} of directional derivatives.

► Overall $O(B \log B)$ time due to sort.