Optimizing Combinatorial and Non-decomposable Metrics with ExactBoost

Daniel Csillag*, Carolina Piazza†, Thiago Ramos*, João Vitor Romano*, Roberto Oliveira*, Paulo Orenstein*

* Instituto de Matemática Pura e Aplicada, Rio de Janeiro, Brazil, † Princeton University, Princeton, USA

Summary

Combinatorial and non-decomposable losses pose computational and theoretical challenges, with many classification algorithms resorting to surrogates. In this work, we:

- introduce a fast and exact stagewise optimization algorithm, dubbed ExactBoost, that boosts stumps to the actual loss function;
- develop a novel extension of margin theory to the non-decomposable setting, recovering a guarantee previously available only to decomposable losses (Bartlett et al. 1998; Koltchinskii and Panchenko 2002; Schapire and Freund 2013);
- bound the generalization error of ExactBoost for metrics with different levels of non-decomposability, e.g. area under the ROC curve (AUC) and Kolmogorov-Smirnov (KS);
- empirically evaluate ExactBoost's performance against surrogate-based and exact algorithms available, showing it is generally superior, especially as an ensembler.

Overview of ExactBoost

The main idea behind ExactBoost is stagewise optimization tailored to a margin-adjusted empirical loss. Take $(X_1, y_1), \dots, (X_n, y_n) \overset{\text{iid}}{\sim} \mathcal{D}$ and a loss $\widehat{L}$ that is invariant under rescaling and translation in its first argument. The goal is to find a score function where higher scores $S(X_i)$ should indicate higher likelihood of $y_i = 1$. It will be assumed that, after T rounds, a score has the form $S(X_i) = \sum_{t=1}^{T} w_t h_t(X_i)$, with $w_t \geq 0, \sum_{t=1}^{T} w_t = 1$, $h_t \in \mathcal{H}$, and the set of weak learners $\mathcal{H}$ are binary stumps of the form

$$\mathcal{H} = \left\{ \pm \mathbf{1}_{[X_{(j)} \leq \xi]} \pm \mathbf{1}_{[X_{(j)} > \xi]} \ : \ \xi \in \mathbb{R}, j \in [p] \right\}. \qquad (1)$$

In order to attenuate overfitting, we consider the margin-adjusted loss $\widehat{L}_\theta(\mathbf{S}, \mathbf{y}) = \widehat{L}(\mathbf{S} - \theta\mathbf{y}, \mathbf{y})$, where $\theta > 0$ is a margin parameter.

The method works by picking (ξ_t, j_t, a_t, b_t) that solves

$$\min_{\xi \in \mathbb{R}, j \in [p], \tilde{a}, \tilde{b} \in \mathbb{R}} \widehat{L}\Big(\mathbf{S}_{t-1} + \tilde{a}\mathbf{1}_{[X_{(j)} \leq \xi]} + \tilde{b}\mathbf{1}_{[X_{(j)} > \xi]} - (1 + |\tilde{b} - \tilde{a}|/2)\theta\mathbf{y}, \mathbf{y}\Big), \qquad (2)$$

and setting $\mathbf{S}_t = \mathbf{S}_{t-1} + a_t\mathbf{1}_{[X_{(j_t)} \leq \xi_t]} + b_t\mathbf{1}_{[X_{(j_t)} > \xi_t]}$.

The resulting algorithm is called ExactBoost, as it is based on the exact loss function provided rather than a surrogate. Note it takes as input an initial set of scores which could be any starting point, including scores trained by other learning models.

Algorithm 1: ExactBoost

```
function ExactBoost(data $(\mathbf{X}, \mathbf{y})$, initial scores $S_0$,
    margin $\theta$, bins $b$, iterations $T$, estimator runs $E$)
for $e \in \{1, \dots, E\}$ do
  $S_e \leftarrow S_0$
  for $t \in \{1, \dots, T\}$ do
    $\mathbf{X}^s, \mathbf{y}^s \leftarrow$ subsample $\mathbf{X}, \mathbf{y}$
    for $j \in \{1, \dots, p\}$ do
      $\widehat{L}(h) \leftarrow \widehat{L}_\theta(S_e(\mathbf{X}^s_{(j)}) + h(\mathbf{X}^s_{(j)}), \mathbf{y}^s)$
      $h_j \leftarrow \operatorname{argmin}_h \widehat{L}(h)$ //Algorithm 2
    end for
    $h \leftarrow \operatorname{argmin}_{h_j} \widehat{L}_\theta(S_e(\mathbf{X}^s) + h_j(\mathbf{X}^s), \mathbf{y}^s)$
    $S'_e \leftarrow S_e + h$
    if $\widehat{L}_\theta(S'_e(\mathbf{X}), \mathbf{y}) \leq \widehat{L}_\theta(S_e(\mathbf{X}), \mathbf{y})$ then
      $S_e \leftarrow S'_e$
      $S_e \leftarrow (S_e - \min S_e)/(\max S_e - \min S_e)$
    end if
  end for
end for
return mean$(S_1, \dots, S_E)$
```

Algorithm 2: Iterative Minimization

```
function Min(loss $\hat{L}_\theta$, data $\mathbf{X}_{(j)}$, labels $\mathbf{y}$, scores $S$,
margin $\theta$)
$\Xi \leftarrow [\min \mathbf{X}_{(j)}, \max \mathbf{X}_{(j)}]$
$A \leftarrow [-1, 1]$; $B \leftarrow [-1, 1]$
for $k \in \{1, \dots, c\}$ do
  $l_\star \leftarrow +\infty$
  for bisections $(\Xi^{(b)}, A^{(b)}, B^{(b)})$ do
    for $i \in \{1, \dots, n\}$ do
      $s \leftarrow S + A^{(b)}\mathbf{1}_{[\mathbf{X}_{(j)} \leq \Xi^{(b)}]} + B^{(b)}\mathbf{1}_{[\mathbf{X}_{(j)} > \Xi^{(b)}]}$
      $\mathbf{s}_i \leftarrow \underline{S_i}$ if $y_i = 0$ otherwise $\overline{S_i}$
    end for
    $l_\star \leftarrow \widehat{L}_\theta(\mathbf{s}, \mathbf{y})$ if $\widehat{L}_\theta(\mathbf{s}, \mathbf{y}) < l_\star$
  end for
end for
$I_\Xi \leftarrow \{\underline{\Xi}, \overline{\Xi}\}$; $I_A \leftarrow \{\underline{A}, \overline{A}\}$; $I_B \leftarrow \{\underline{B}, \overline{B}\}$
$S(a, b, \xi) \leftarrow S + a\mathbf{1}_{[\mathbf{X}_{(j)} \leq \xi]} + b\mathbf{1}_{[\mathbf{X}_{(j)} > \xi]}$
$(\xi_\star, a_\star, b_\star) \leftarrow \operatorname{argmin}_{\xi \in I_\Xi, a \in I_A, b \in I_B} \widehat{L}_\theta(S(a, b, \xi), \mathbf{y})$
return $S + a_\star\mathbf{1}_{[\mathbf{X}_{(j)} \leq \xi_\star]} + b_\star\mathbf{1}_{[\mathbf{X}_{(j)} > \xi_\star]}$
```

Theoretical Results

Notation Let $\mathcal{D}$ be a probability distribution over $(X, y) \in \mathbb{R}^p \times \{0, 1\}$, and $\mathcal{D}_0$ (respectively, $\mathcal{D}_1$) denote the conditional distribution of X when $y = 0$ (respectively, 1). Conditionally on n_1 and n_0 respectively, the subsamples $\mathbf{X}_1 := (X_i : i \in [n], y_i = 1)$ and $\mathbf{X}_0 := (X_i : i \in [n], y_i = 0)$ are iid from $\mathcal{D}_1$ and $\mathcal{D}_0$. Score functions $S : \mathbb{R}^p \to [-1, 1]$ are convex combinations of elements in a family of measurable functions $\mathcal{H} : \mathbb{R}^p \to [-1, 1]$. Denote by $\mathcal{R}_n(\mathcal{H})$ the Rademacher complexity $\mathcal{H}$ with respect to $\mathcal{D}$ and $\mathcal{R}_{n,y}(\mathcal{H})$ for $y \in \{0, 1\}$ the complexities with respect to $\mathcal{D}_0$ and $\mathcal{D}_1$. Note $\mathcal{R}_n(\mathcal{H}) = O(\sqrt{\log p/n})$ when the set of weak learners $\mathcal{H}$ is as in (1).

Define the populational AUC and the populational KS losses as

$$\mathrm{AUC}(S) := 1 - \Pr\{S(X) > S(X')\},$$
$$\mathrm{KS}(S) := 1 - \sup_{t \in \mathbb{R}} \left(\Pr_{X \sim \mathcal{D}_0} \{S(X) \leq t\} - \Pr_{X \sim \mathcal{D}_1} \{S(X) \leq t\} \right).$$

Theorem 1. *Given $\theta > 0$, $\delta \in (0, 1)$, and a class of functions $\mathcal{H}$ from $\mathbb{R}^p$ to $[-1, 1]$, the following holds with probability at least $1 - \delta$: for all score functions $S : \mathbb{R}^p \to [-1, 1]$ obtained as convex combinations of the elements of $\mathcal{H}$,*

$$\mathrm{AUC}(S) \leq \widehat{\mathrm{AUC}}_\theta(S) + \frac{4}{\theta}\zeta_{\mathrm{AUC}}(\mathcal{H}) + \sqrt{\frac{2\log(1/\delta)}{\min\{n_0, n_1\}}},$$

where $\zeta_{\mathrm{AUC}}(\mathcal{H}) = \mathcal{R}_{\min\{n_0,n_1\},0}(\mathcal{H}) + \mathcal{R}_{\min\{n_0,n_1\},1}(\mathcal{H})$.

For ExactBoost, where $\mathcal{H}$ is given by (1), the theorem implies that the score S produced by the algorithm satisfies $\mathrm{AUC}(S) \leq \widehat{\mathrm{AUC}}_\theta(S) + o(1)$ with high probability whenever $\min\{n_0, n_1\} \gg \theta^{-2} \log p$. This result can be extended to similar pairwise losses.

Theorem 2. *Given $\theta > 0$, $\delta \in (0, 1)$, and a class of functions $\mathcal{H}$ from $\mathbb{R}^p$ to $[-1, 1]$, the following holds with probability at least $1 - \delta$: for all score functions $S : \mathbb{R}^p \to [-1, 1]$ obtained as convex combinations of the elements of $\mathcal{H}$,*

$$\mathrm{KS}(S) \leq \widehat{\mathrm{KS}}_\theta(S) + \frac{8}{\theta}\zeta_{\mathrm{KS}}(\mathcal{H}) + \sqrt{\frac{\log(2/\delta)}{2}} \left(\frac{1}{\sqrt{n_0}} + \frac{1}{\sqrt{n_1}} \right),$$

where $\zeta_{\mathrm{KS}}(\mathcal{H}) = \mathcal{R}_{n_0,0}(\mathcal{H}) + \mathcal{R}_{n_1,1}(\mathcal{H}) + n_0^{-1/2} + n_1^{-1/2}$.

In words, a score that achieves a small margin-adjusted KS loss should, with high probability, have good performance on the population. In the specific case of ExactBoost, the theorem above yields with probability greater than $1 - \delta$,

$$\mathrm{KS}(S) \leq \widehat{\mathrm{KS}}_\theta(S) + C\sqrt{\frac{\theta^{-2}(1 + \log p) + \log(2/\delta)}{\min\{n_0, n_1\}}},$$

with $C > 0$ a universal constant. Treating δ as fixed, good training performance on the margin-adjusted loss leads to good generalization when the number of positive and negative examples in the data are much larger than $\theta^{-2} \log p$.

Subsampling Subsampling can help ExactBoost avoid overfitting. The next proposition is helpful in controlling its impact in the optimization procedure for some losses.

Proposition 1. *Let $\widehat{L}$ be either the $\widehat{\mathrm{AUC}}$ or the $\widehat{\mathrm{KS}}$ loss. Consider a subset of indices $I = I_0 \cup I_1 \subset [n]$ chosen independently and uniformly at random with equal number of positive and negative cases, $|I_0| = |I_1| = k$. Let h_R be the optimal stump over the reduced sample $\{(X_j, y_j)\}_{j \in I}$ and score S and h_* the optimal stump over the entire sample $\{(X_i, y_i)\}_{i \in [n]}$. Then,*

$$\mathbb{E}[\widehat{L}(S + h_R)] \leq \widehat{L}(S + h_*) + \frac{e}{k},$$

where the expectation is over the choice of I.

Using random subsets of observations in ExactBoost leads to an expected error close to optimal when the subset has a balanced proportion of positive and negative examples.

Experiments

Benchmarks ExactBoost is compared to many learning algorithms: AdaBoost, k-nearest neighbors, logistic regression and random forest (via their Scikit-Learn implementation), gradient boosting (via XGBoost) and a 4-layer neural net (via TensorFlow). Methods that specifically optimize the performance metric are also considered: RankBoost (Freund et al. 2003) for AUC and DMKS (Fang and Chen 2019) for KS.

Hyperparameters Hyperparameters were fixed throughout the experiments. Baseline models were trained with the package-provided hyperparameters. Aided by experimental evidence on held-out datasets, ExactBoost uses $E = 250$ runs, $T = 50$ rounds, subsampling of 20% and margin of $\theta = 0.05$.

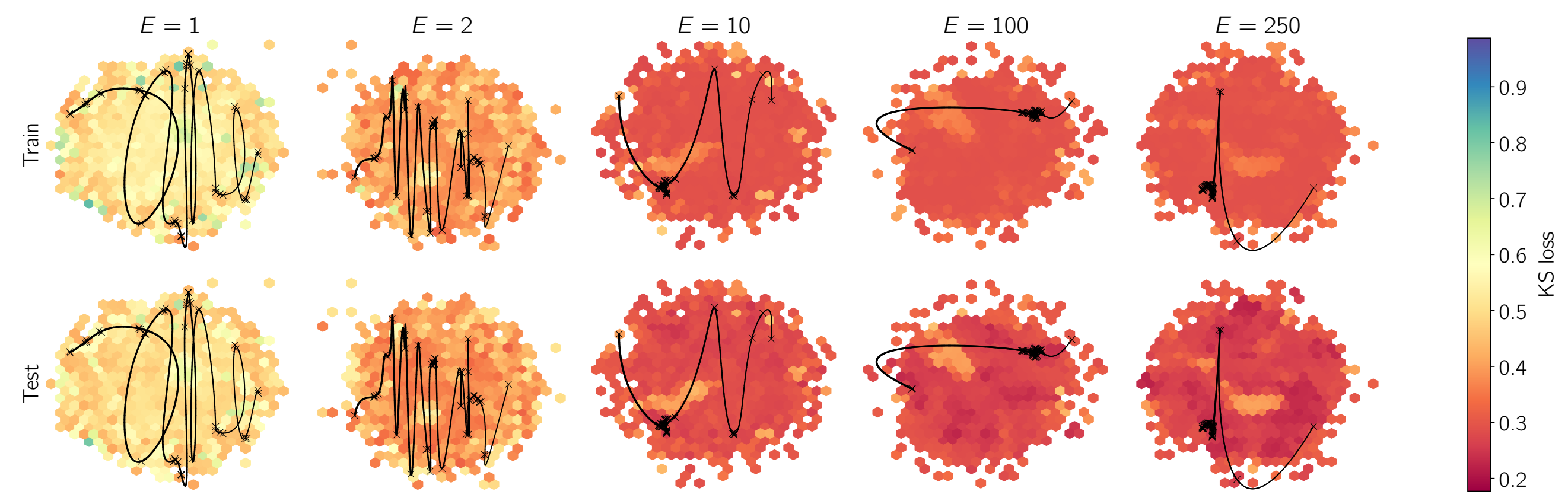

Fig 1: KS loss landscape highlighting ExactBoost's optimization trajectories on dataset heart via UMAP, going from left to right. More runs E lead to better train and test losses.

Datasets and timings Table 1 displays the main characteristics of each dataset, which span various applications, and range from balanced to imbalanced. Timing comparisons against exact benchmarks are also provided.

Dataset	Obs.	Features	Positives	RankBoost	DMKS
a1a	1605	119	24.6%	3.10x	34.50x
german	1000	20	70.0%	11.10x	2.00x
gisette	6000	5000	50.0%	OOT	36.70x
gmsc	150000	10	6.7%	OOT	87.50x
heart	303	21	45.9%	1.90x	23.70x
iono	351	34	64.1%	2.90x	4.70x
liver	145	5	37.9%	3.50x	17.40x

Table 1: Dataset properties and timings of algorithms relative to ExactBoost. ExactBoost is always faster ($> 1\times$). OOT indicates the time budget of 5 days was exceeded (see the paper for details on computational setup).

Results Figure 2 shows that ExactBoost has good performance against surrogate alternatives. Results in the paper show that ExactBoost is also generally better as a standalone estimator than loss-specific optimizers such as RankBoost and DMKS.

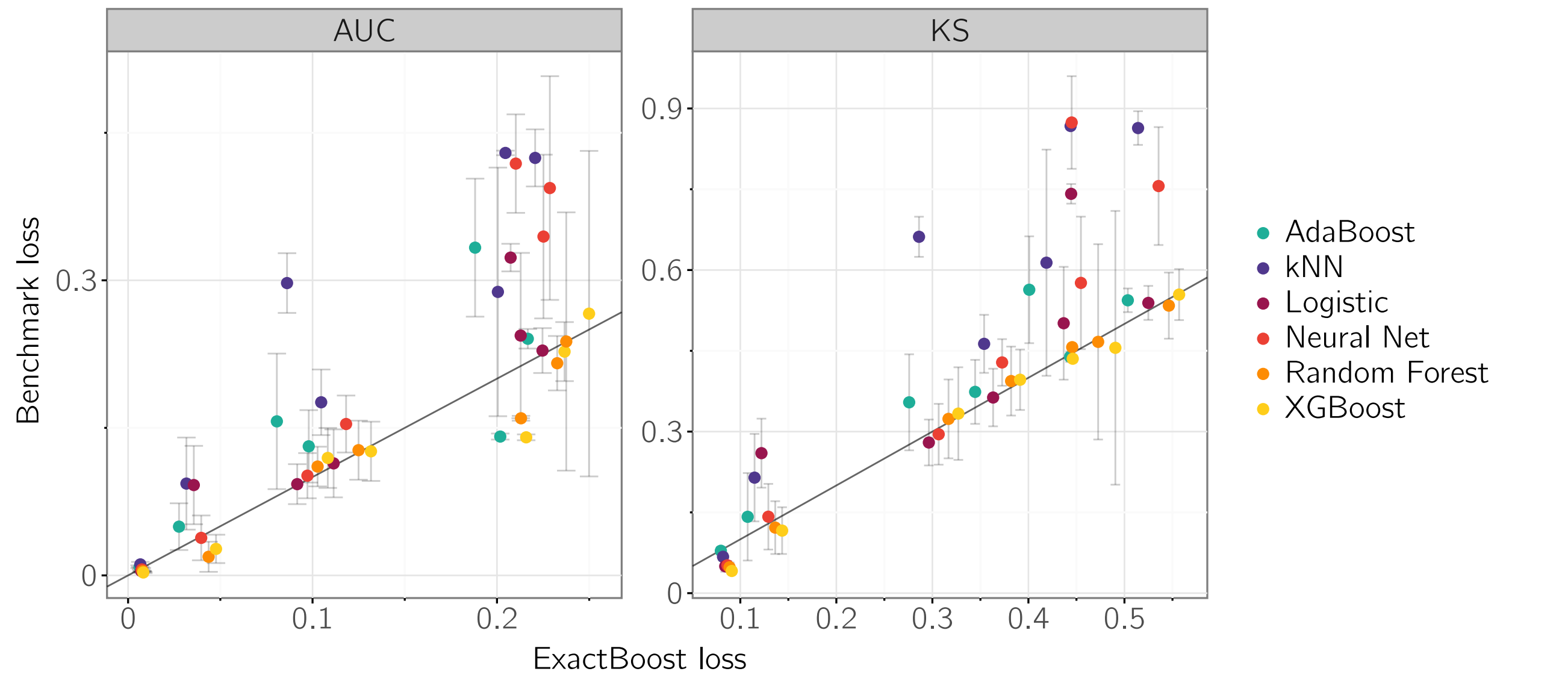

Fig 2: Test error for ExactBoost vs surrogate methods as estimators. Alternatives are generally worse than ExactBoost or statistically indistinguishable.

Ensembling ExactBoost is also a great ensembler. Six base models were used: AdaBoost, k-nearest neighbors, logistic regression, neural network, random forest and XGBoost. These models were trained on training folds, and their predictions on test folds were used as features for the ensemble models. Tables 2 and 3 show the results of using different surrogate and exact models as ensemblers.

Dataset	ExactBoost	AdaBoost	Logistic	XGBoost	RankBoost
a1a	**0.13** ± **0.0**	0.17 ± 0.0	0.14 ± 0.0	0.28 ± 0.1	0.16 ± 0.0
german	**0.23** ± **0.0**	0.32 ± 0.0	0.24 ± 0.0	0.35 ± 0.0	0.30 ± 0.1
gisette	**0.00** ± **0.0**	0.01 ± 0.0	0.01 ± 0.0	0.02 ± 0.0	0.01 ± 0.0
gmsc	0.15 ± 0.0	**0.14** ± **0.0**	0.31 ± 0.0	0.41 ± 0.0	0.15 ± 0.0
heart	**0.12** ± **0.0**	0.18 ± 0.1	**0.12** ± **0.0**	0.23 ± 0.1	0.15 ± 0.0
iono	**0.04** ± **0.0**	0.05 ± 0.0	0.07 ± 0.0	0.09 ± 0.0	0.05 ± 0.0
liver	**0.30** ± **0.1**	0.34 ± 0.1	0.34 ± 0.1	0.38 ± 0.0	0.38 ± 0.1

Table 2: Evaluation of AUC ensemblers. ExactBoost is generally the top performer.

Dataset	ExactBoost	AdaBoost	Logistic	XGBoost	DMKS
a1a	**0.37** ± **0.1**	0.44 ± 0.1	0.40 ± 0.1	0.57 ± 0.1	0.49 ± 0.1
german	**0.50** ± **0.1**	0.68 ± 0.1	0.53 ± 0.1	0.69 ± 0.1	0.53 ± 0.1
gisette	**0.04** ± **0.0**	**0.04** ± **0.0**	0.07 ± 0.0	**0.04** ± **0.0**	0.10 ± 0.0
gmsc	**0.43** ± **0.0**	0.44 ± 0.0	0.73 ± 0.0	0.83 ± 0.0	0.46 ± 0.0
heart	**0.34** ± **0.1**	0.38 ± 0.1	0.37 ± 0.1	0.46 ± 0.1	0.40 ± 0.0
iono	**0.13** ± **0.1**	0.18 ± 0.1	0.18 ± 0.1	0.19 ± 0.1	0.27 ± 0.1
liver	**0.53** ± **0.1**	0.60 ± 0.2	0.59 ± 0.2	0.76 ± 0.0	0.60 ± 0.2

Table 3: Evaluation of KS ensemblers. ExactBoost is always the top performer.

Takeaways

The main takeaways from the proposed boosting algorithm are:

- first-order surrogate methods are widely used in classification tasks, but there is value to be gained in working with the intended loss function;
- ExactBoost's theoretical guarantees translate to compelling empirical performance;
- computational implementations can be made reasonably fast through interval arithmetic;
- we expect the margin extension and theoretical results presented to hold over more general classes of combinatorial and non-decomposable losses beyond AUC and KS (see the main paper for the precision at k loss).

Main References

Bartlett, Peter et al. (1998). "Boosting the margin: a new explanation for the effectiveness of voting methods". In: *Annals of Statistics* 26.5, pp. 1651–1686.

Fang, Fang and Yuanyuan Chen (2019). "A new approach for credit scoring by directly maximizing the Kolmogorov–Smirnov statistic". In: *Computational Statistics & Data Analysis* 133, pp. 180–194. ISSN: 0167-9473.

Freund, Yoav et al. (Dec. 2003). "An Efficient Boosting Algorithm for Combining Preferences". In: *J. Mach. Learn. Res.* 4, pp. 933–969. ISSN: 1532-4435.

Koltchinskii, V. and D. Panchenko (Feb. 2002). "Empirical Margin Distributions and Bounding the Generalization Error of Combined Classifiers". In: *Annals of Statistics* 30.1, pp. 1–50. DOI: 10.1214/aos/1015362183.

Schapire, Robert E and Yoav Freund (2013). "Boosting: Foundations and algorithms". In: *Kybernetes*. DOI: 10.1108/03684921311295547.