Modern methods in gradient boosting: theory and practice

Gleb Gusev, Sber AI Lab

June 25, 2022

Outline

1. Minimal Variance Sampling in stochastic gradient boosting

Outline

Outline

Outline

Outline

Outline

Outline

Background on stochastic gradient boosting

ML setting

- Dataset $\mathcal{D} = \{(\mathrm{x}_k, y_k)\}_{k=1..n}$, $\mathrm{x}_k \in \mathbb{R}^m$, $y_k \in \mathbb{R}$
- (x_k, y_k) i.i.d. according to unknown $P(\cdot, \cdot)$
- $L(\hat{y}, y)$ is a given loss function

Background on stochastic gradient boosting

ML setting

- Dataset $\mathcal{D} = \{(\mathrm{x}_k, y_k)\}_{k=1..n}$, $\mathrm{x}_k \in \mathbb{R}^m$, $y_k \in \mathbb{R}$
- (x_k, y_k) i.i.d. according to unknown $P(\cdot, \cdot)$
- $L(\hat{y}, y)$ is a given loss function

Problem:
Find $F^*\colon \mathbb{R}^m \to \mathbb{R}$, a good predictor of y: $\quad F^* = \arg\min_F \mathbb{E}_P(L(F(\mathrm{x}), y))$

Background on stochastic gradient boosting

Gradient boosting (GB)

- Choose a set of "weak" hypotheses $\mathcal{F} \subset \{f : \mathbb{R}^m \to \mathbb{R}\}$

Background on stochastic gradient boosting

Gradient boosting (GB)

- Choose a set of "weak" hypotheses $\mathcal{F} \subset \{f : \mathbb{R}^m \to \mathbb{R}\}$
- At each step:
  1. Compute derivatives:
     $g^t(x_k, y_k) = \frac{\partial L(s, y_k)}{\partial s}\big|_{s=F^t(x_k)}$, $h^t(x_k, y_k) = \frac{\partial^2 L(s, y_k)}{\partial s^2}\big|_{s=F^t(x_k)}$
  2. Approximate negative gradient / Newton step by $f^t \in \mathcal{F}$:

$$f^t = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{k=1}^{n} h^t(x_k, y_k) \left(-\frac{g^t(x_k, y_k)}{h^t(x_k, y_k)} - f(x_k) \right)^2$$

  3. Make a step: $F^{t+1} = F^t + \delta \cdot f^t$

Background on stochastic gradient boosting

Gradient boosting (GB)

- Choose a set of "weak" hypotheses $\mathcal{F} \subset \{f : \mathbb{R}^m \to \mathbb{R}\}$
- At each step:
  1. Compute derivatives:
     $g^t(\mathrm{x}_k, y_k) = \frac{\partial L(s, y_k)}{\partial s}\big|_{s=F^t(\mathrm{x}_k)}$, $h^t(\mathrm{x}_k, y_k) = \frac{\partial^2 L(s, y_k)}{\partial s^2}\big|_{s=F^t(\mathrm{x}_k)}$
  2. Approximate negative gradient / Newton step by $f^t \in \mathcal{F}$:

$$f^t = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{k=1}^{n} h^t(\mathrm{x}_k, y_k) \left(-\frac{g^t(\mathrm{x}_k, y_k)}{h^t(\mathrm{x}_k, y_k)} - f(\mathrm{x}_k) \right)^2$$

  3. Make a step: $F^{t+1} = F^t + \delta \cdot f^t$
- After T steps, obtain a "strong" model F^T

Background on stochastic gradient boosting

A key problem

- weak models f^t are highly correlated, since they are trained on the same dataset
- This leads to high variance wrt. data randomness
- what mean a limited generalization ability of GB

Background on stochastic gradient boosting

A key problem

- weak models f^t are highly correlated, since they are trained on the same dataset
- This leads to high variance wrt. data randomness
- what mean a limited generalization ability of GB

From GB to Stochastic GB

A common approach is to use random subsampling of the data at each gradient step

Background on stochastic gradient boosting

Stochastic Gradient Boosting (SGB)[1]

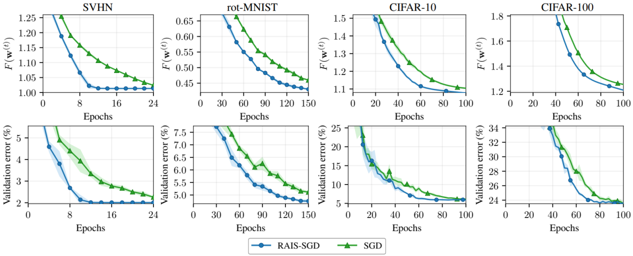
- A randomized version of gradient boosting algorithm proposed by Friedman
- At each iteration t, random fraction s of the dataset is used to fit the model f^t.
- SGB selects random $s \cdot n$ observations $\mathcal{D}^t \subset \mathcal{D}$ uniformly and without replacement
- SGB improves the quality of the learned model and reduces training complexity

[1] J. H. Friedman, Stochastic gradient boosting. Computational Statistics & Data Analysis 38(4), 2002

Background on stochastic gradient boosting

Non-uniform sampling

Importance sampling shows its superiority over uniform sampling[2]:



Some non-uniform sampling methods were proposed for AdaBoost algorithm, but they are not applicable to SGB with decision trees

[2] Tyler B. Johnson, Carlos Guestrin, Training Deep Models Faster with Robust, Approximate Importance Sampling, NeurIPS, 2018

Background on stochastic gradient boosting

Gradient-based one-side sampling (GOSS)[3]

- GOSS samples:
  - αn objects with largest absolute gradients with probability 1
  - $(s - \alpha)n$ other objects at random

[3] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu, LightGBM: A highly efficient gradient boosting decision tree. In Advances in Neural Information Processing Systems, 2017

Background on stochastic gradient boosting

Gradient-based one-side sampling (GOSS)[3]

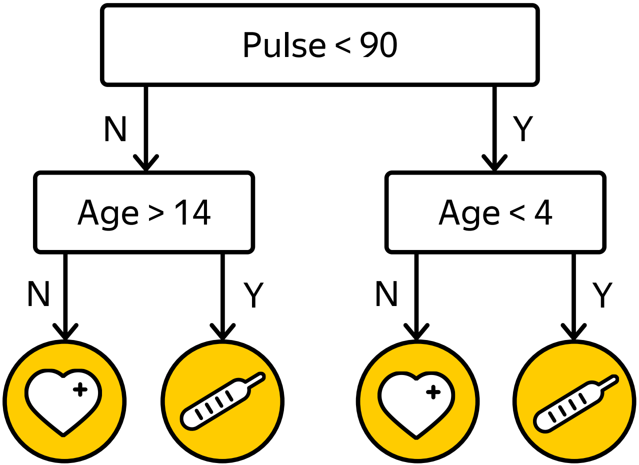
- GOSS samples:
  - αn objects with largest absolute gradients with probability 1
  - $(s - \alpha)n$ other objects at random
- For unbiased estimation, GOSS uses weights:
  - αn samples with largest gradients are used with weight 1
  - other samples are used with weight $\frac{1-\alpha}{s-\alpha}$.

[3] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu, LightGBM: A highly efficient gradient boosting decision tree. In Advances in Neural Information Processing Systems, 2017

Outline

A formalisation of sampling problem & theoretical analysis

Example of a decision tree model



A formalisation of sampling problem & theoretical analysis

Choosing a split

- At each step of building a tree, we take some candidate splits and choose the best one among them
- $S(f, v)$ is a score of a split based on feature f and threshold value v
- $S(f, v) = \sum_{l \in L} \left(\min_{c_l} \sum_{i \in l} h^t(\mathrm{x}_i, y_i) \left(-\frac{g^t(\mathrm{x}_i, y_i)}{h^t(\mathrm{x}_i, y_i)} - c_l \right)^2 \right) = \sum\limits_{l \in L} \frac{\left(\sum_{i \in l} g_i\right)^2}{\sum\limits_{i \in l} h_i} + Const.$

  Hint: $c_l = \frac{\sum_{i \in l} g_i}{\sum\limits_{i \in l} h_i}$.

A formalisation of sampling problem & theoretical analysis

Minimal Variance Sampling in Stochastic Gradient Boosting[4]

- Let $\xi_i := Id((x_i, y_i) \in \mathcal{D}^t)$ be independent Bernoulli variables, $\xi_i \sim \text{Bernoulli}(p_i)$.
- Sampling ratio is $s = \frac{1}{n}\mathbb{E}\sum_{i=1}^{n} \xi_i = \frac{1}{n}\sum_{i=1}^{n} p_i$.
- Inverse probability weighting estimation: $w_i = \frac{1}{p_i}$ for instance i

[4] B. Ibragimov, G. Gusev, NeurIPS, 2019.

A formalisation of sampling problem & theoretical analysis

Minimal Variance Sampling in Stochastic Gradient Boosting[4]

- Let $\xi_i := Id((x_i, y_i) \in \mathcal{D}^t)$ be independent Bernoulli variables, $\xi_i \sim \text{Bernoulli}(p_i)$.
- Sampling ratio is $s = \frac{1}{n}\mathbb{E}\sum_{i=1}^{n} \xi_i = \frac{1}{n}\sum_{i=1}^{n} p_i$.
- Inverse probability weighting estimation: $w_i = \frac{1}{p_i}$ for instance i
- Score is approximated by $\hat{S}(f, v) := \sum_{l \in L} \frac{\left(\sum_{i \in l} \frac{1}{p_i}\xi_i g_i\right)^2}{\sum_{i \in l} \frac{1}{p_i}\xi_i h_i}$
- Goal: choose p_i that minimize $\mathbb{E}\Delta^2 = \mathbb{E}\left(\hat{S}(f, v) - S(f, v)\right)^2$

[4] B. Ibragimov, G. Gusev, NeurIPS, 2019.

A formalisation of sampling problem & theoretical analysis

Theorem

$$\text{Denote } A_l := \sum_{i \in l} \frac{1}{p_i} \xi_i g_i,\ B_l := \sum_{i \in l} \frac{1}{p_i} \xi_i h_i,\ c_l := \frac{\sum_{i \in l} g_i}{\sum_{i \in l} h_i}$$

$$\text{We have } \mathbb{E}\Delta^2 \approx \sum_{l \in L} c_l^2 (4Var(A_l) - 4c_l Cov(A_l, B_l) + c_l^2 Var(B_l))$$

A formalisation of sampling problem & theoretical analysis

Sketch of proof

- Estimate the expectation by representing $\hat{S}(f, v)$ as the value of function $F(a_1, b_1, \ldots, a_{|L|}, b_{|L|}) := \sum_{l=1}^{|L|} \frac{a_l^2}{b_l}$ at point $(\mathrm{A}_1, B_1, \ldots, A_{|L|}, B_{|L|})$.

A formalisation of sampling problem & theoretical analysis

Sketch of proof

- Estimate the expectation by representing $\hat{S}(f, v)$ as the value of function $F(a_1, b_1, \ldots, a_{|L|}, b_{|L|}) := \sum_{l=1}^{|L|} \frac{a_l^2}{b_l}$ at point $(\mathrm{A}_1, B_1, \ldots, A_{|L|}, B_{|L|})$.

- Use the first-order Taylor series expansion of F at point $(\mu_{a_1}, \mu_{b_1}, \ldots, \mu_{a_{|L|}}, \mu_{b_{|L|}})$, where $\mu_{a_l} = \mathbb{E}A_l = \sum_{i \in l} g_i$ and $\mu_{b_l} = \mathbb{E}B_l = \sum_{i \in l} h_i$.

A formalisation of sampling problem & theoretical analysis

Sketch of proof

- Estimate the expectation by representing $\hat{S}(f, v)$ as the value of function $F(a_1, b_1, \ldots, a_{|L|}, b_{|L|}) := \sum_{l=1}^{|L|} \frac{a_l^2}{b_l}$ at point $(\mathrm{A}_1, B_1, \ldots, A_{|L|}, B_{|L|})$.

- Use the first-order Taylor series expansion of F at point $(\mu_{a_1}, \mu_{b_1}, \ldots, \mu_{a_{|L|}}, \mu_{b_{|L|}})$, where $\mu_{a_l} = \mathbb{E}A_l = \sum_{i \in l} g_i$ and $\mu_{b_l} = \mathbb{E}B_l = \sum_{i \in l} h_i$.
- Without loss of generality, we further provide calculations for the case $|L| = 1$.

A formalisation of sampling problem & theoretical analysis

Sketch of proof

- Estimate the expectation by representing $\hat{S}(f, v)$ as the value of function $F(a_1, b_1, \ldots, a_{|L|}, b_{|L|}) := \sum_{l=1}^{|L|} \frac{a_l^2}{b_l}$ at point $(A_1, B_1, \ldots, A_{|L|}, B_{|L|})$.

- Use the first-order Taylor series expansion of F at point $(\mu_{a_1}, \mu_{b_1}, \ldots, \mu_{a_{|L|}}, \mu_{b_{|L|}})$, where $\mu_{a_l} = \mathbb{E}A_l = \sum_{i \in l} g_i$ and $\mu_{b_l} = \mathbb{E}B_l = \sum_{i \in l} h_i$.
- Without loss of generality, we further provide calculations for the case $|L| = 1$.
- We have $F(a_1, b_1) \approx F(\mu_{a_1}, \mu_{b_1}) + 2\frac{\mu_{a_1}}{\mu_{b_1}}(a_1 - \mu_{a_1}) - \frac{\mu_{a_1}^2}{\mu_{b_1}^2}(b_1 - \mu_{b_1})$, and, therefore,

$$\Delta = F(a_1, b_1) - F(\mu_{a_1}, \mu_{b_1}) \approx 2\frac{\mu_{a_1}}{\mu_{b_1}}(a_1 - \mu_{a_1}) - \frac{\mu_{a_1}^2}{\mu_{b_1}^2}(b_1 - \mu_{b_1}).$$

A formalisation of sampling problem & theoretical analysis

Sketch of proof

- Estimate the expectation by representing $\hat{S}(f, v)$ as the value of function $F(a_1, b_1, \dots, a_{|L|}, b_{|L|}) := \sum_{l=1}^{|L|} \frac{a_l^2}{b_l}$ at point $(A_1, B_1, \dots, A_{|L|}, B_{|L|})$.

- Use the first-order Taylor series expansion of F at point $(\mu_{a_1}, \mu_{b_1}, \dots, \mu_{a_{|L|}}, \mu_{b_{|L|}})$, where $\mu_{a_l} = \mathbb{E}A_l = \sum_{i \in l} g_i$ and $\mu_{b_l} = \mathbb{E}B_l = \sum_{i \in l} h_i$.
- Without loss of generality, we further provide calculations for the case $|L| = 1$.
- We have $F(a_1, b_1) \approx F(\mu_{a_1}, \mu_{b_1}) + 2\frac{\mu_{a_1}}{\mu_{b_1}}(a_1 - \mu_{a_1}) - \frac{\mu_{a_1}^2}{\mu_{b_1}^2}(b_1 - \mu_{b_1})$, and, therefore,

  $\Delta = F(a_1, b_1) - F(\mu_{a_1}, \mu_{b_1}) \approx 2\frac{\mu_{a_1}}{\mu_{b_1}}(a_1 - \mu_{a_1}) - \frac{\mu_{a_1}^2}{\mu_{b_1}^2}(b_1 - \mu_{b_1})$.
- Further, we have

$$\mathbb{E}\Delta^2 \approx \mathbb{E}(2\frac{\mu_{a_1}}{\mu_{b_1}}(a_1 - \mu_{a_1}) - \frac{\mu_{a_1}^2}{\mu_{b_1}^2}(b_1 - \mu_{b_1}))^2 = c_1^2(4Var(a_1) - 4c_1 Cov(a_1, b_1) + c_1^2 Var(b_1)).$$

Outline

Minimal Variance Sampling (MVS)

Further simplifications towards an optimization problem

- Note that $-4c_l Cov(A_l, B_l) \leq 4Var(A_l) + c_l^2 Var(B_l)$
- so $\sum_{l \in L} c_l^2 \left(4Var(A_l) + c_l^2 Var(B_l)\right)$ is an upper bound for $\mathrm{E}\Delta^2$

Minimal Variance Sampling (MVS)

Further simplifications towards an optimization problem

- Note that $-4c_l Cov(A_l, B_l) \leq 4Var(A_l) + c_l^2 Var(B_l)$
- so $\sum_{l \in L} c_l^2 \left(4Var(A_l) + c_l^2 Var(B_l)\right)$ is an upper bound for $E\Delta^2$
- Replacing c_l by a constant upper bound and bounding $(1 - p_i)$ by 1:

$$\sum_{i=1}^{n} \frac{1}{p_i} g_i^2 + \lambda \sum_{i=1}^{n} \frac{1}{p_i} h_i^2 \to \min_{p_i} \quad \text{w.r.t.} \quad \sum_{i=1}^{n} p_i = n \cdot s \quad \text{and} \quad p_i \in [0,1],\ i = 1, \ldots, n.$$

Minimal Variance Sampling (MVS)

Theorem

There exists a value μ *such that* $p_i = \min\left(1, \frac{\sqrt{g_i^2+\lambda h_i^2}}{\mu}\right)$ *is a solution for the above problem*

Minimal Variance Sampling (MVS)

Theorem

There exists a value μ *such that* $p_i = \min\left(1, \frac{\sqrt{g_i^2+\lambda h_i^2}}{\mu}\right)$ *is a solution for the above problem*

Remark

- for $\lambda = 0$ we have importance sampling
- for $\lambda \to \infty$ and $h_i = 1$ we have SGB.

Minimal Variance Sampling (MVS)

MVS algorithm

- Sort the data by ascending $g_i^2 + \lambda h_i^2$
- Compute $cumsum[k] = \sum_{i=1}^{k} \sqrt{g_i^2 + \lambda h_i^2}$
- Sample rate $s[i] = \frac{n - i + \frac{cumsum[i]}{\sqrt{g_i^2 + \lambda h_i^2}}}{n}$
- Use binary search to find the threshold
- It is possible to reduce complexity from $O(n \log n)$ to $O(n)$

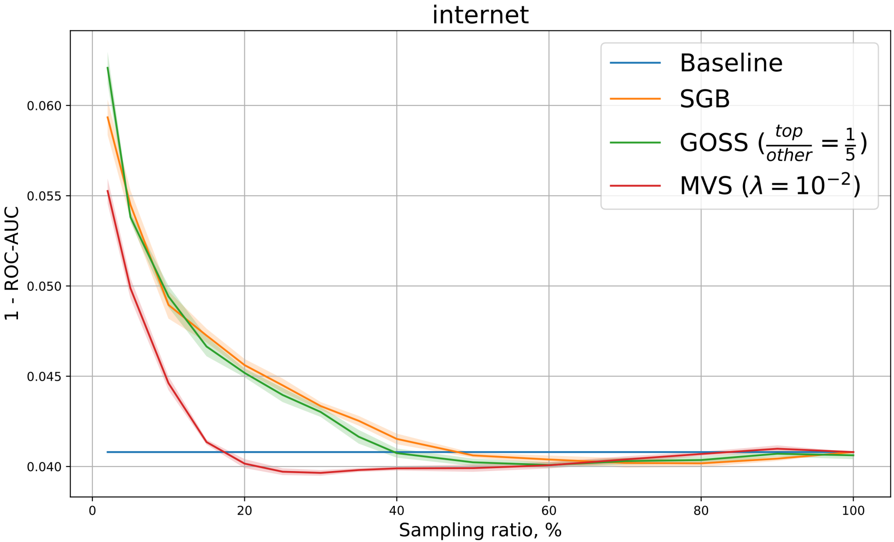
Outline

Experimental results

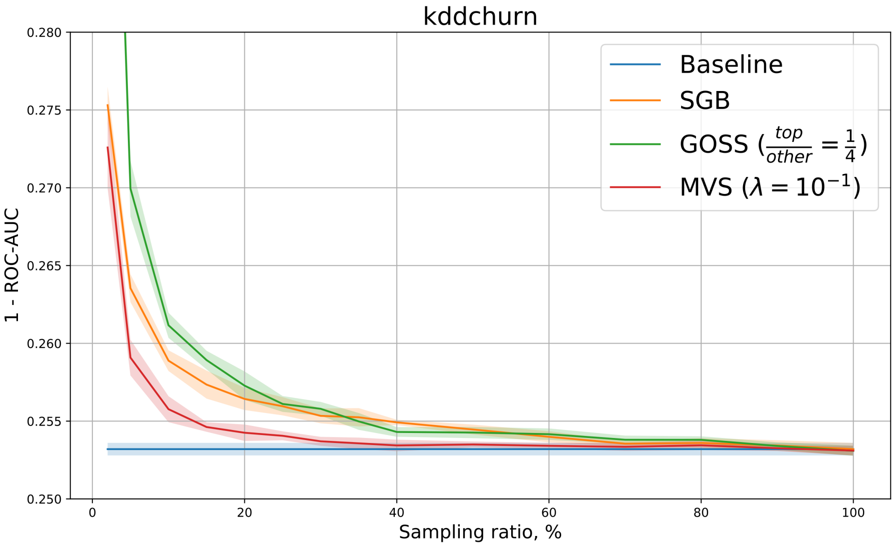
Datasets

Dataset	# Examples	# Features
KDD Internet	10108	69
Adult	48842	15
Amazon	32769	10
KDD Upselling	50000	231
Kick prediction	72983	36
KDD Churn	50000	231
Click prediction	399482	12

Experimental results



Experimental results



Experimental results

Performance comparison

	KDD Internet	Adult	Amazon	KDD Upselling	Kick	KDD Churn	Click	Average
Baseline	0.0408	0.0688	0.1517	0.1345	0.2265	**0.2532**	0.2655	-0.0%
SGB	-1.13%	+0.81%	-1.14%	+0.03%	-0.14%	+0.14%	**-0.14%**	-0.22%
GOSS	-0.64%	-0.11%	-1.23%	+0.07%	-0.10%	+0.16%	-0.09%	-0.28%
MVS	**-3.03%**	**-0.24%**	**-1.78%**	-0.07%	**-0.19%**	+0.17%	-0.04%	**-0.74%**
MVS Adaptive	-2.79%	-0.13%	-1.57%	**-0.28**%	**-0.19**%	+0.07%	-0.03%	-0.70%

Table: Baseline scores / relative error change

Experimental results

Different sample rates

Sample rate	0.02	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.5
SGB	+19.92%	+11.35%	+6.83%	+4.99%	+3.84%	+3.03%	+2.17%	+1.57%	+1.10%	+0.42%
GOSS	+22.37%	+12.75%	+8.00%	+5.32%	+3.39%	+2.25%	+1.41%	+0.75%	+0.23%	-0.16%
MVS	+13.93%	+7.76%	**+3.69%**	+1.91%	+0.74%	+0.14%	**-0.21%**	**-0.43%**	**-0.41%**	-0.45%
MVS Adaptive	**+13.72%**	**+7.47%**	+3.71%	**+1.70%**	**+0.55%**	**-0.03%**	-0.07%	-0.28%	-0.32%	**-0.51%**

Table: Relative error change, average over datasets

Conclusion

1. MVS: a theoretically grounded sampling method for SGB
2. Improves generalization ability / training time
3. Used as a dafault setting in Catboost at Yandex
4. Replaced ordered boosting[5], a highly complex and expensive option

[5] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, Andrey Gulin, "CatBoost: unbiased boosting with categorical features", NeurIPS, 2018.

Outline

Outline

Outline

Outline

Outline

Outline

Outline

1. Minimal Variance Sampling in stochastic gradient boosting
   - 1.1 Background on stochastic gradient boosting
   - 1.2 A formalisation of sampling problem & theoretical analysis
   - 1.3 Minimal Variance Sampling
   - 1.4 Experimental results
2. Instance-wise early stopping
   - 2.1 Cross-validation scheme
   - 2.2 Naíve approaches
   - 2.3 Main idea
   - 2.4 Two-level cross-validation algorithm
   - 2.5 Experiments

Background on stochastic gradient boosting

Gradient boosting (GB)

- Choose a set of "weak" hypotheses $\mathcal{F} \subset \{f : \mathbb{R}^m \to \mathbb{R}\}$
- At each step:
  1. Compute derivatives: $g^t(\mathrm{x}_k, y_k) = \frac{\partial L(s, y_k)}{\partial s}\big|_{s=F^t(\mathrm{x}_k)}$, $h^t(\mathrm{x}_k, y_k) = \frac{\partial^2 L(s, y_k)}{\partial s^2}\big|_{s=F^t(\mathrm{x}_k)}$
  2. Approximate negative gradient / Newton step by $f^t \in \mathcal{F}$:

$$f^t = \operatorname*{arg\,min}_{f \in \mathcal{F}} \frac{1}{n} \sum_{k=1}^{n} h^t(\mathrm{x}_k, y_k) \left(-\frac{g^t(\mathrm{x}_k, y_k)}{h^t(\mathrm{x}_k, y_k)} - f(\mathrm{x}_k) \right)^2$$

  3. Make a step: $F^{t+1} = F^t + \delta \cdot f^t$
- After T steps, obtain a "strong" model F^T

Background on stochastic gradient boosting

Key problem

- How to select the number of steps T?

$$F^T = \sum_{t=1}^{T} \delta \cdot f^t$$

Outline

Cross-validation scheme

It is standard to use cross-validation protocol to determine an optimal number of steps:

- Randomly split $\mathcal{D} = \bigsqcup_{j=1}^{k} \mathcal{S}_j$ into k disjoint subsets.

Cross-validation scheme

It is standard to use cross-validation protocol to determine an optimal number of steps:

- Randomly split $\mathcal{D} = \bigsqcup_{j=1}^{k} \mathcal{S}_j$ into k disjoint subsets.
- For each j, train an ensemble F_j on S_{-j} and obtain a learning curve on S_j:

$$l_j^{(t)} = \frac{1}{|\mathcal{S}_j|} \sum_{(x,y)\in\mathcal{S}_j} L\left(F_j^t(x), y\right), \forall t \leq T$$

Cross-validation scheme

It is standard to use cross-validation protocol to determine an optimal number of steps:

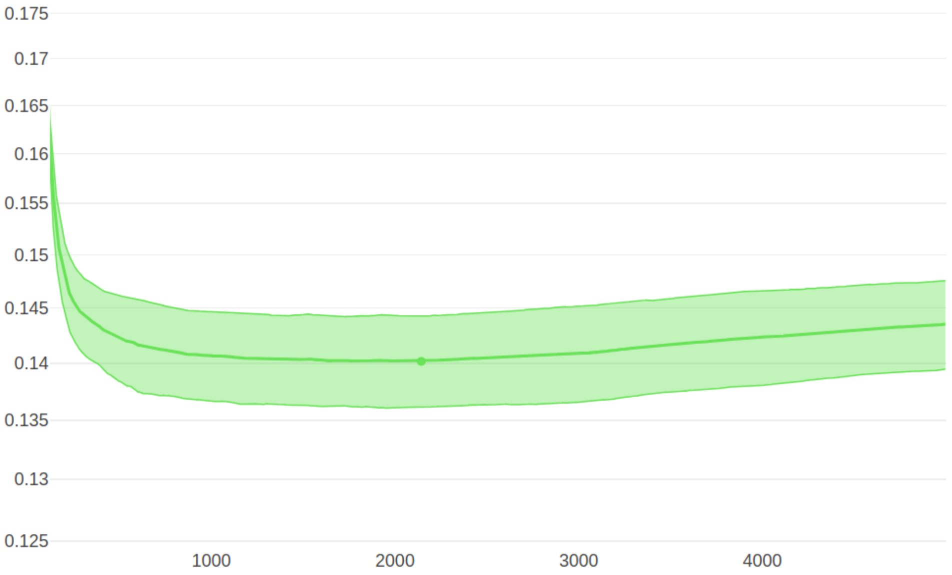
- Randomly split $\mathcal{D} = \bigsqcup_{j=1}^{k} \mathcal{S}_j$ into k disjoint subsets.
- For each j, train an ensemble F_j on S_{-j} and obtain a learning curve on S_j:

$$l_j^{(t)} = \frac{1}{|\mathcal{S}_j|} \sum_{(x,y)\in\mathcal{S}_j} L\left(F_j^t(x), y\right), \forall t \leq T$$

- Average learning curves over all j and select the moment $\hat{t}$ with the least value:

$$\hat{t} := \arg\min_t l^{(t)}, \quad l^{(t)} = \frac{1}{k} \sum_j l_j^{(t)}.$$

Cross-validation scheme



Cross-validation scheme

- This scheme aims at the optimization problem

$$\min_t \mathbb{E}_{(x,y)\sim P}[L(F^t(x), y)]$$

.

Cross-validation scheme

- This scheme aims at the optimization problem

$$\min_t \mathbb{E}_{(x,y)\sim P}[L(F^t(x), y)]$$

.

- However, we could set another problem instead:

$$\mathbb{E}_{(x,y)\sim P} \min_{t(x)}[L(F^{t(x)}(x), y)]$$

due to an obvious inequality:

$$\mathbb{E}_{x\sim P} \min_{t(x)} \mathbb{E}_{(y|x)\sim P}[L(F^{t(x)}, y)] \leq \min_t \mathbb{E}_{(x,y)\sim P}[L(F^t(x), y)]$$

.

Cross-validation scheme

- This scheme aims at the optimization problem

$$\min_t \mathbb{E}_{(x,y)\sim P}[L(F^t(x), y)]$$

.

- However, we could set another problem instead:

$$\mathbb{E}_{(x,y)\sim P} \min_{t(x)} [L(F^{t(x)}(x), y)]$$

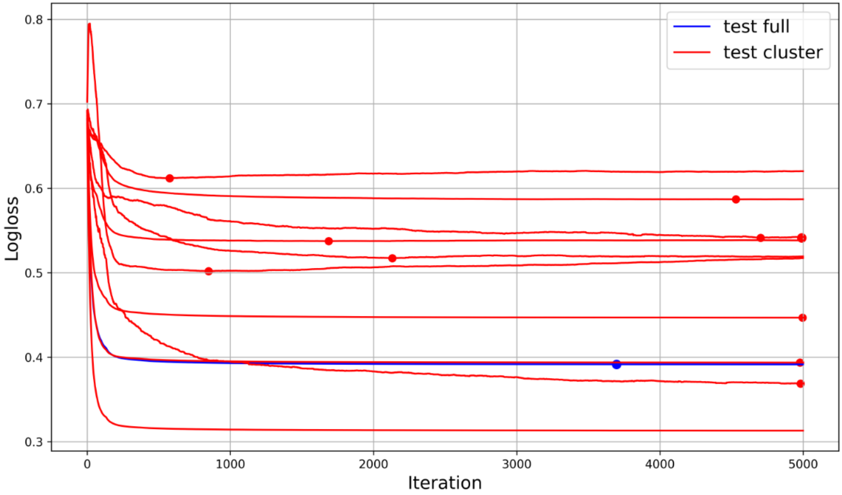
due to an obvious inequality:

$$\mathbb{E}_{x\sim P} \min_{t(x)} \mathbb{E}_{(y|x)\sim P}[L(F^{t(x)}, y)] \le \min_t \mathbb{E}_{(x,y)\sim P}[L(F^t(x), y)]$$

.

- The standard cross-validation scheme ignores heterogeneity of the sample space.

Cross-validation scheme



Outline

Naíve approaches

Straightforward regression idea

- calculate an individual optimal moment for each objects by cross–validation

Naíve approaches

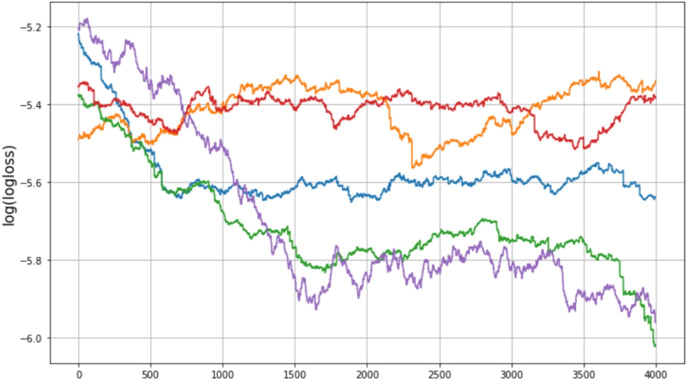
Straightforward regression idea

- calculate an individual optimal moment for each objects by cross–validation
- approximate optimal moment for each object by a separate regression model

Naíve approaches

Straightforward regression idea

- calculate an individual optimal moment for each objects by cross–validation
- approximate optimal moment for each object by a separate regression model
- Fails due to large noise: target is clearly an overestimate.



Outline

Main idea

- Suppose the input space $\mathcal{D}$ is divided into C disjoint regions $(\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_C)$ in such a way that all samples in $\mathcal{D}_i$ are close to each other in some sense.
- Ensemble size selection based on partition $\{\mathcal{D}_i\}$, where the number of estimators is chosen individually for each cluster $\mathcal{D}_i$, can have better quality compared to one "universal" common size:

$$\mathbb{E}_P \min_t [L(F^t(x), y)] \leq \mathbb{E}_{\mathcal{D}_i \sim \mathcal{D}} \min_t \mathbb{E}[L(F^t(x), y) | \mathcal{D}_i] \leq \min_t \mathbb{E}_P [L(F^t(x), y)].$$

Main idea

Algorithm 1 Adaptive stopping procedure

Input: $\mathcal{S} = (\boldsymbol{X}, \boldsymbol{y})$

$folds \leftarrow (\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_k) \leftarrow CvSplit(k, \mathcal{S})$

$cvPredictions \leftarrow CvPredict(folds)$

$partition \leftarrow (\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_C) \leftarrow GetPartition(\mathcal{S})$

$bestIterations \leftarrow EstimateBestIterations(folds, cvPredictions, partition)$

$finalModel \leftarrow Train(\boldsymbol{X}, \boldsymbol{y}, partition, bestIterations)$

return $finalModel$

Outline

Two-level cross-validation algorithm

Problems

- It is still unclear how to estimate the possible effect of cluster-based pruning for a particular learning task.
- Moreover, the proposed method incorporates some extra hyperparameters which can be tuned (e.g., clusterization method and number of clusters)
- Obviously, since the validation sets are used to estimate stopping moments for clusters, we can not use them for tuning. In particular, the error estimated in this way monotonically decreases with growth of cluster count.

Two-level cross-validation algorithm

To avoid above problems we propose the following framework:

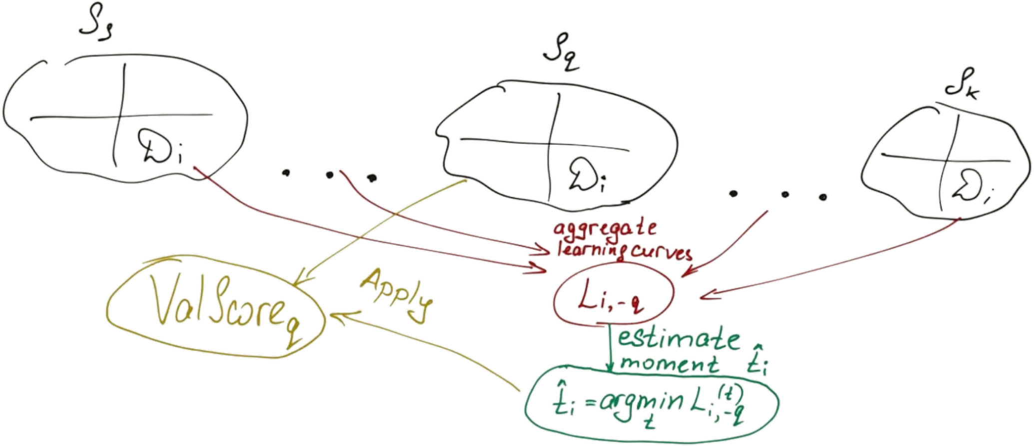
- Let $\mathcal{D}_{i,j} = \mathcal{D}_i \cap \mathcal{S}_j$ be the set of objects from the j-th fold belonging to the cluster $\mathcal{D}_i$ and $n_{i,j} = |\mathcal{D}_{i,j}|$.
- Calculate the learning curve for each $\mathcal{D}_{i,j}$

$$l_{i,j}^{(t)} = \frac{1}{n_{i,j}} \sum_{(x,y) \in \mathcal{D}_{i,j}} L\left(F_j^t(x), y\right).$$

- To obtain less biased estimator, for each fold q we shrink the model size to the number of steps calculated via learning curves for the remaining folds:

$$L_{i,-q}^{(t)} = \frac{\sum_{j \neq q} n_{i,j} \cdot l_{i,j}^{(t)}}{\sum_{j \neq q} n_{i,j}},$$

Two-level cross-validation algorithm



Two-level cross-validation algorithm

Algorithm 3 Evaluation Procedure

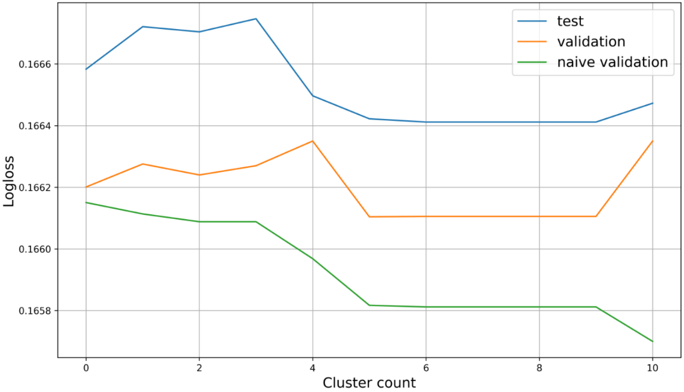
```
procedure EVALUATE(folds, cvPredictions, partition)
    for S_q ← folds do
        {M_i^q} ← EstimateBestIteration(folds \ S_q, cvPredictions, partition)
        predictions_q ← cvPredictions[S_q]
        for D_i ← partition do
            Shrink(predictions_q[S_q ∩ D_i], M_i^q)
        end for
        L_q = Eval(predictions_q)
    end for
    return Mean({L_q})
end procedure
```

The complexity of this step $O(C(T+k)+nT)$ is meager compared to the ensemble training complexity, which is at least $\Omega(mndT)$

Outline

Experiments

- We use one of the most popular implementation of Gradient Boosting – CatBoost, as it achieves SOTA results on many benchmarks.
- We use divisive clustering via decision tree to obtain data clusters.



Experiments

- We use one of the most popular implementation of Gradient Boosting – CatBoost, as it achieves SOTA results on many benchmarks.
- We use divisive clustering via decision tree to obtain data clusters.

Table: Quality estimation, 0-1 loss / logloss, relative error change

	Adult	**Amazon**	**KDD Upselling**	**Kick**	**KDD Internet**
Baseline	0.1264 / 0.2723	0.0447 / 0.1400	0.0494 / 0.1666	**0.0496** / 0.2857	0.1004 / 0.2202
Adaptive pruning	**-0.24% / -0.24%**	**-1.37% / -0.53%**	**-0.20% / -0.10%**	+0.11% / **-0.19%**	**-2.46% / -0.52%**
	Click	**Higgs**	**Marketing**	**Default**	**HEPMASS**
Baseline	**0.1564** / 0.3916	0.2364 / 0.4810	0.0926 / 0.1937	0.1865 / 0.4327	0.1258 / 0.2768
Adaptive pruning	+0.04% / **-0.03%**	**-0.14% / -0.14%**	**-2.27% / -0.71%**	**-2.50%** / **-0.07%**	**-0.17% / -0.16%**

Thank you!