Optimal Parallelization of Boosting



Arthur C.W. da Cunha* Mikael Møller Høgsgaard* Kasper Green Larsen

Aarhus University - Denmark

NeurIPS
2024

*Looking for jobs

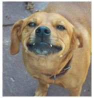
Binary classification

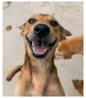
- Domain: $\mathcal{X}$ (and a distribution $\mathcal{D}$ over it)

Binary classification

- Domain: $\mathcal{X}$ (and a distribution $\mathcal{D}$ over it)

$\mathcal{X}$

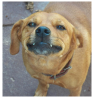


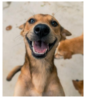




Binary classification

- Domain: $\mathcal{X}$ (and a distribution $\mathcal{D}$ over it)
- Label space: $\mathcal{Y}$, with $|\mathcal{Y}| = 2$

$\mathcal{X}$

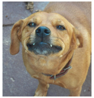


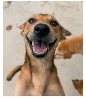




Binary classification

- Domain: $\mathcal{X}$ (and a distribution $\mathcal{D}$ over it)
- Label space: $\mathcal{Y}$, with $|\mathcal{Y}| = 2$

$\mathcal{X}$

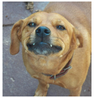


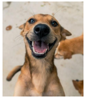




$\mathcal{Y}$ "CAT" "DOG"

- Domain: $\mathcal{X}$ (and a distribution $\mathcal{D}$ over it)
- Label space: $\mathcal{Y}$, with $|\mathcal{Y}| = 2$
- Target concept: $c\colon \mathcal{X} \to \mathcal{Y}$

$\mathcal{X}$

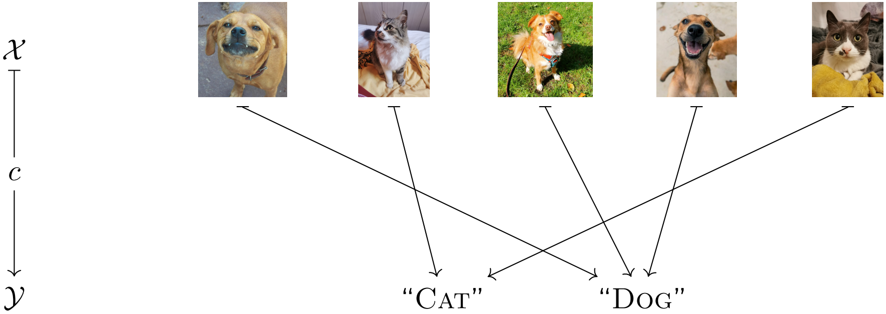



$\mathcal{Y}$ “Cat” “Dog”

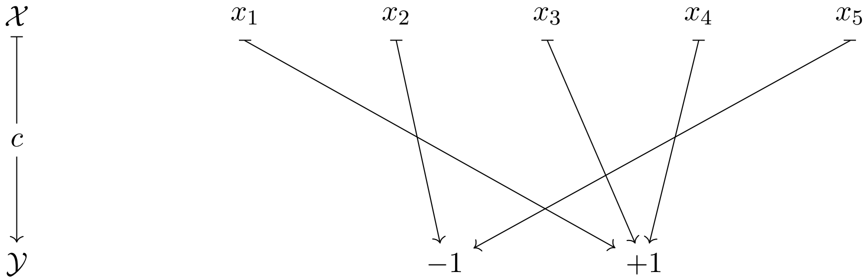
Binary classification

- Domain: $\mathcal{X}$ (and a distribution $\mathcal{D}$ over it)
- Label space: $\mathcal{Y}$, with $|\mathcal{Y}| = 2$
- Target concept: $c\colon \mathcal{X} \to \mathcal{Y}$



Binary classification

- Domain: $\mathcal{X}$ (and a distribution $\mathcal{D}$ over it)
- Label space: $\mathcal{Y}$, with $|\mathcal{Y}| = 2$
- Target concept: $c\colon \mathcal{X} \to \mathcal{Y}$

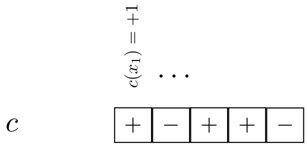


- Domain: $\mathcal{X}$ (and a distribution $\mathcal{D}$ over it)
- Label space: $\mathcal{Y}$, with $|\mathcal{Y}| = 2$
- Target concept: $c\colon \mathcal{X} \to \mathcal{Y}$

c	$+$	$-$	$+$	$+$	$-$

Binary classification

- Domain: $\mathcal{X}$ (and a distribution $\mathcal{D}$ over it)
- Label space: $\mathcal{Y}$, with $|\mathcal{Y}| = 2$
- Target concept: $c\colon \mathcal{X} \to \mathcal{Y}$



Goal: Strong Learner

c																																											
	−	−	+	−	+	+	−	−	+	+	+	+	−	+	−	+	−	+	−	−	−	+	−	+	−	+	−	+	−	+	−	+	+	−	+	+	+	+	−	−	−	−	⋯

Goal: Strong Learner

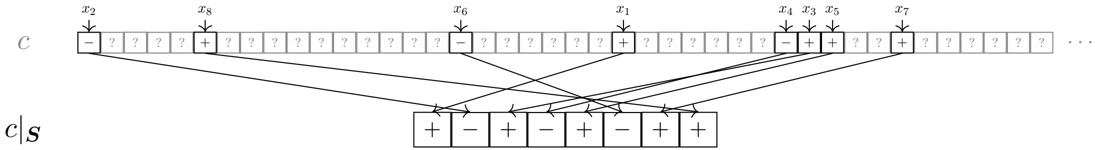
c	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	$\cdots$

Goal: Strong Learner

- We only see a sample $\boldsymbol{S} \sim \mathcal{D}^m$ $(m = 8)$

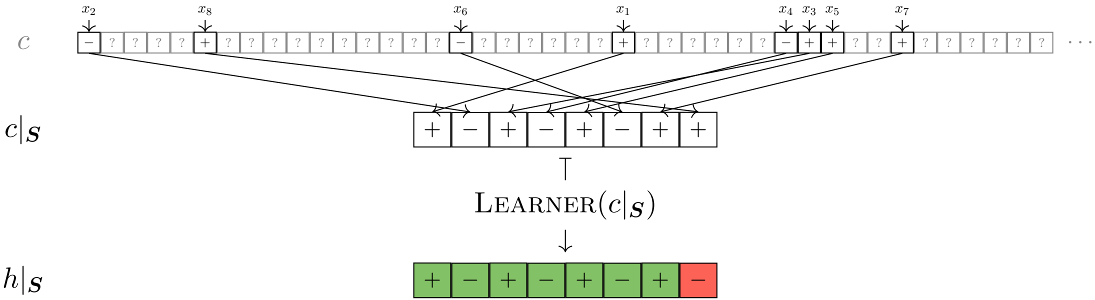
Goal: Strong Learner

- We only see a sample $\boldsymbol{S} \sim \mathcal{D}^m$ $(m = 8)$



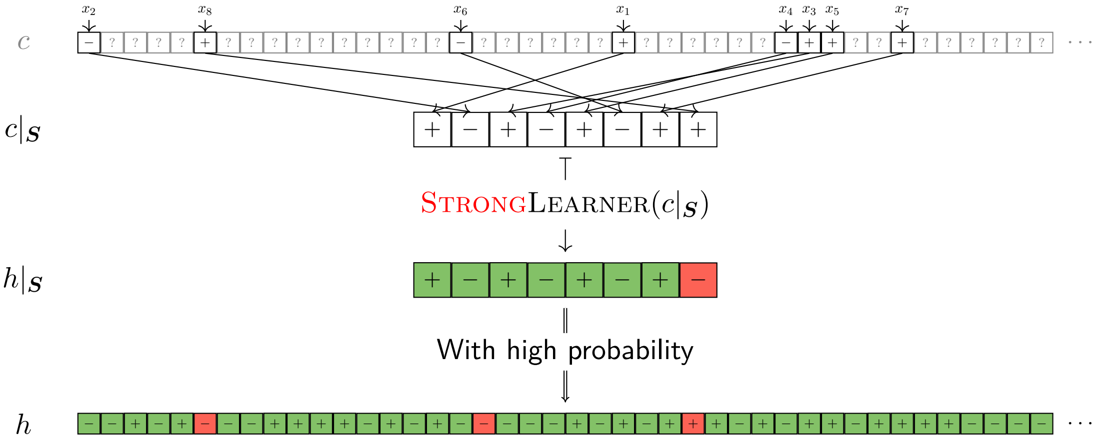
Goal: Strong Learner

- We only see a sample $S \sim \mathcal{D}^m$ $(m = 8)$



Goal: Strong Learner

- We only see a sample $S \sim \mathcal{D}^m$ $(m = 8)$



Algorithm StrongLearner such that

Algorithm StrongLearner such that

For all Precision $\varepsilon \in (0, 1)$.

Algorithm $\textsc{StrongLearner}$ such that

For all Precision $\varepsilon \in (0, 1)$.

Given Sufficiently large $m = m(\varepsilon)$ sample $\boldsymbol{S} \sim \mathcal{D}^m$.

Algorithm $\textsc{StrongLearner}$ such that

For all Precision $\varepsilon \in (0,1)$.

Given Sufficiently large $m = m(\varepsilon)$ sample $\boldsymbol{S} \sim \mathcal{D}^m$.

Given Examples of a target concept on sample $c|_{\boldsymbol{S}}$.

Algorithm $\textsc{StrongLearner}$ such that

For all Precision $\varepsilon \in (0, 1)$.

Given Sufficiently large $m = m(\varepsilon)$ sample $\boldsymbol{S} \sim \mathcal{D}^m$.

Given Examples of a target concept on sample $c|_{\boldsymbol{S}}$.

Satisfies With high probability returns a classifier $h = \textsc{StrongLearner}(c|_{\boldsymbol{S}})$ that approximates c well on the entire domain.

Goal: Strong Learner (more formally)

Algorithm $\textsc{StrongLearner}$ such that

For all Precision $\varepsilon \in (0, 1)$.

Given Sufficiently large $m = m(\varepsilon)$ sample $\boldsymbol{S} \sim \mathcal{D}^m$.

Given Examples of a target concept on sample $c|_{\boldsymbol{S}}$.

Satisfies With high probability returns a classifier $h = \textsc{StrongLearner}(c|_{\boldsymbol{S}})$ that approximates c well on the entire domain. More precisely,

$$\mathrm{err}_{\mathcal{D}}(h) := \Pr_{\mathbf{x} \sim \mathcal{D}}[h(\mathbf{x}) \neq c(\mathbf{x})] < \varepsilon.$$

Starting from: Weak Learner

$c \mid \boldsymbol{s}$	$-$	$-$	$+$	$-$	$+$	$+$	$-$	$-$	$+$	$+$	$+$	$+$	$-$

Starting from: Weak Learner

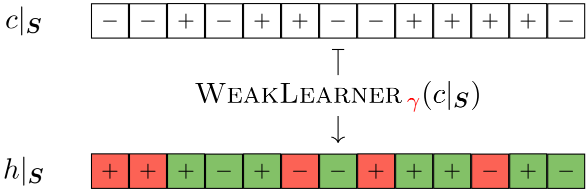
$c\|_{\boldsymbol{S}}$	−	−	+	−	+	+	−	−	+	+	+	+	−

$\uparrow$

$\textsc{WeakLearner}(c|_{\boldsymbol{S}})$

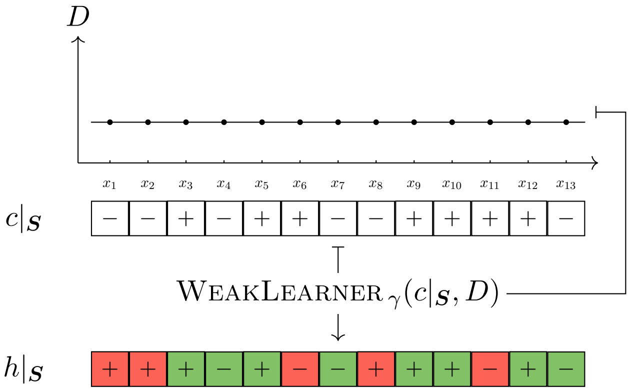
$\downarrow$

$h\|_{\boldsymbol{S}}$	+	+	+	−	+	−	−	+	+	+	−	+	−



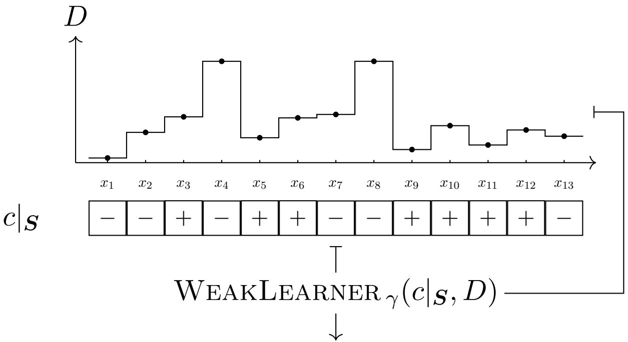
✓ at least $50\% + \gamma$

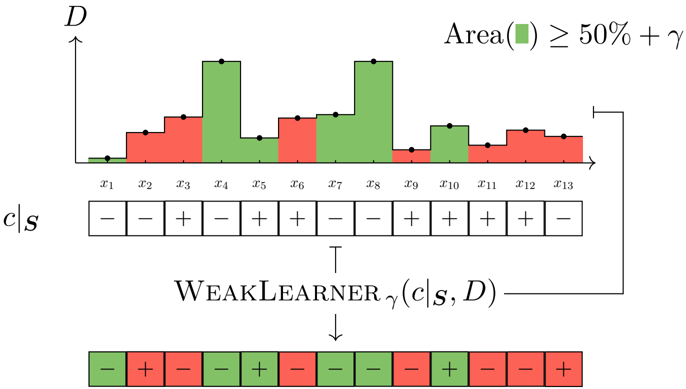
Starting from: Weak Learner



✓ at least $50\% + \gamma$

Starting from: Weak Learner





D
Area(■) $\geq 50\% + \gamma$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13}
$c|_S$
− − + − + + − − + + + + −
$\textsc{WeakLearner}_\gamma(c|_S, D)$
− + − − + − − − − + − − +

Algorithm $\textsc{WeakLearner}_\gamma$ such that

Algorithm $\mathcal{W}_\gamma$ such that

Algorithm $\mathcal{W}_\gamma$ such that

Given Set $S \subseteq \mathcal{X}$.

Algorithm $\mathcal{W}_\gamma$ such that

Given Set $S \subseteq \mathcal{X}$.

Given Examples of the target concept $c|_S$.

Algorithm $\mathcal{W}_\gamma$ such that

Given Set $S \subseteq \mathcal{X}$.

Given Examples of the target concept $c|_S$.

Given Any distribution (weighing) D over S.

Algorithm $\mathcal{W}_\gamma$ such that

- **Given** Set $S \subseteq \mathcal{X}$.
- **Given** Examples of the target concept $c|_S$.
- **Given** Any distribution (weighing) D over S.
- **Satisfies** Returns a classifier $h = \mathcal{W}_\gamma(c|_S, D)$ that approximates c a bit better than chance on the training data.

Algorithm $\mathcal{W}_\gamma$ such that

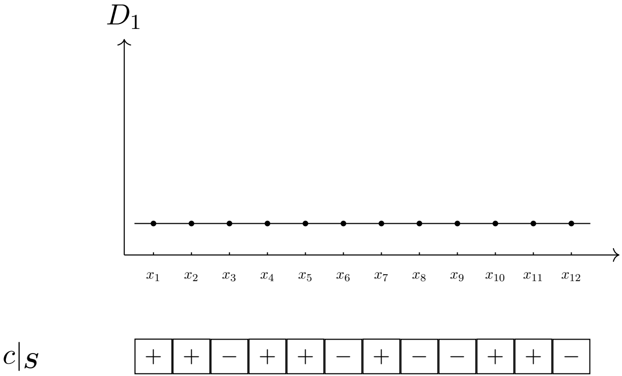
Given Set $S \subseteq \mathcal{X}$.

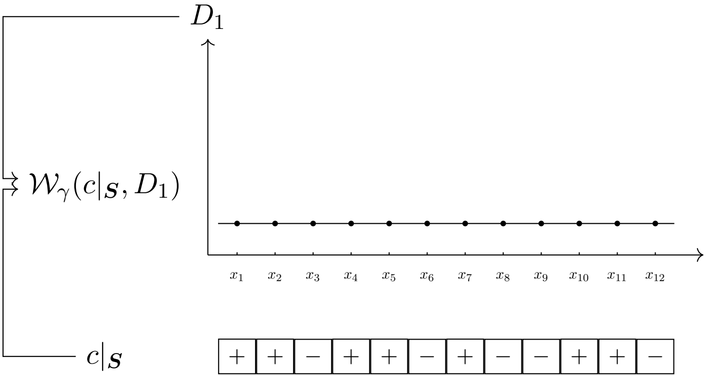
Given Examples of the target concept $c|_S$.

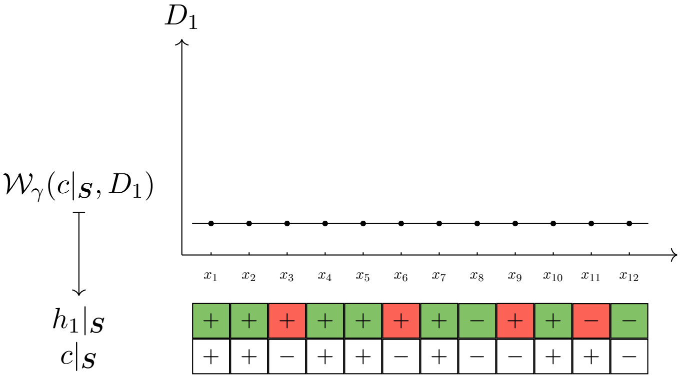
Given Any distribution (weighing) D over S.

Satisfies Returns a classifier $h = \mathcal{W}_\gamma(c|_S, D)$ that approximates c a bit better than chance on the training data. More precisely,

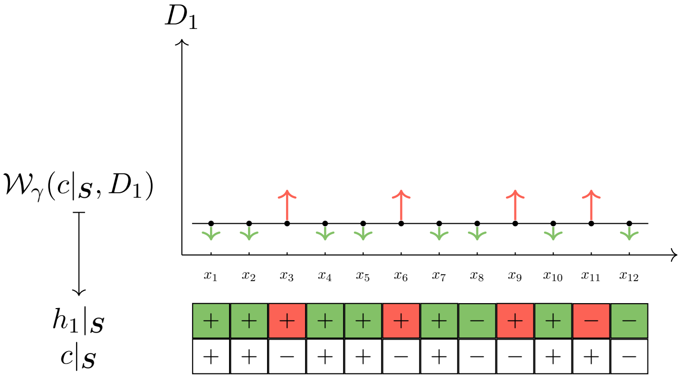
$$\mathrm{err}_D(h) \coloneqq \Pr_{\mathbf{x}\sim D}[h(\mathbf{x}) \neq c(\mathbf{x})] < \frac{1}{2} - \gamma.$$


D_1
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
$c|_{\boldsymbol{S}}$
+ + − + + − + − − + + −

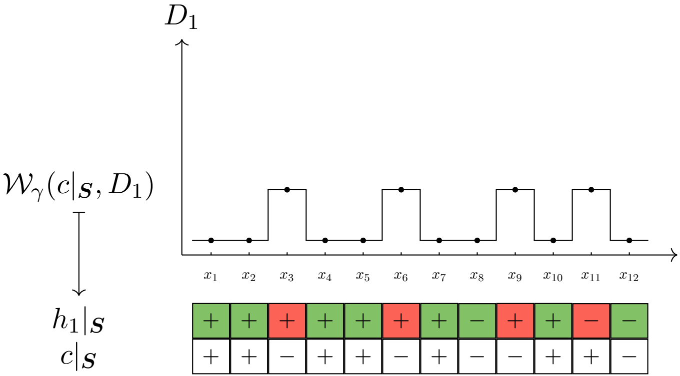

D_1
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_1)$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
$c|_{\boldsymbol{S}}$
+ + − + + − + − − + + −



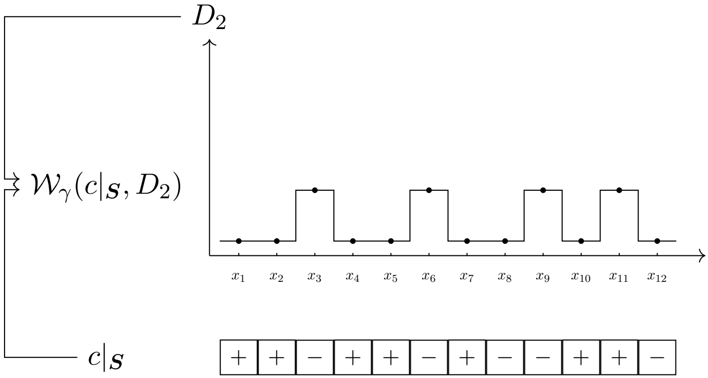
D_1
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_1)$
$h_1|_{\boldsymbol{S}}$
$c|_{\boldsymbol{S}}$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
+ + + + + + + − + + − −
+ + − + + − + − − + + −

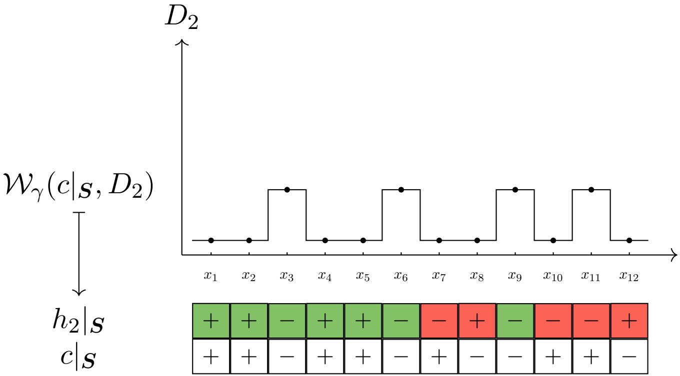


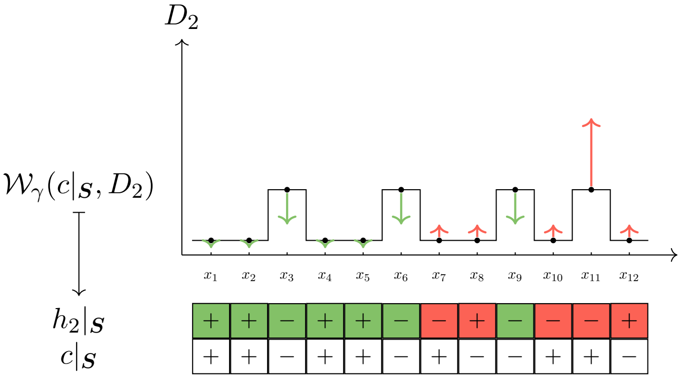
D_1
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_1)$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
$h_1|_{\boldsymbol{S}}$
+ + + + + + + − + + − −
$c|_{\boldsymbol{S}}$
+ + − + + − + − − + + −

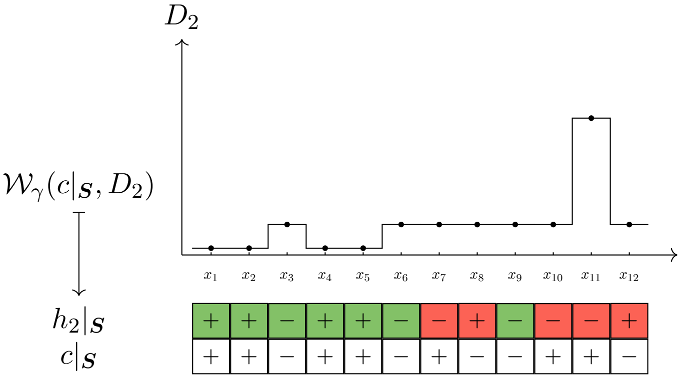


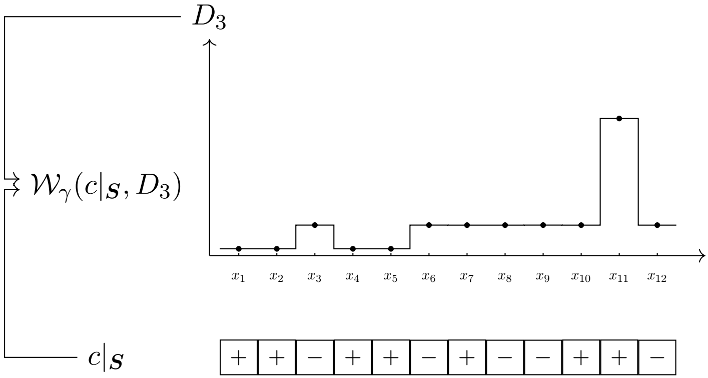
D_1
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_1)$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
$h_1|_{\boldsymbol{S}}$
$c|_{\boldsymbol{S}}$
+ + + + + + + − + + − −
+ + − + + − + − − + + −


D_2
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_2)$
$c|_{\boldsymbol{S}}$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
+ + − + + − + − − + + −

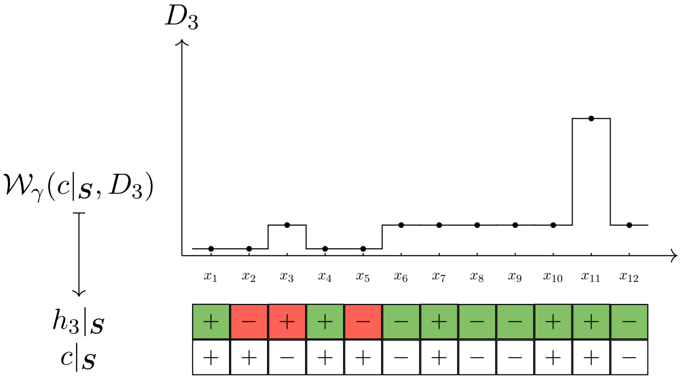

D_2
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_2)$
$h_2|_{\boldsymbol{S}}$
$c|_{\boldsymbol{S}}$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
+ + − + + − − + − − − +
+ + − + + − + − − + + −

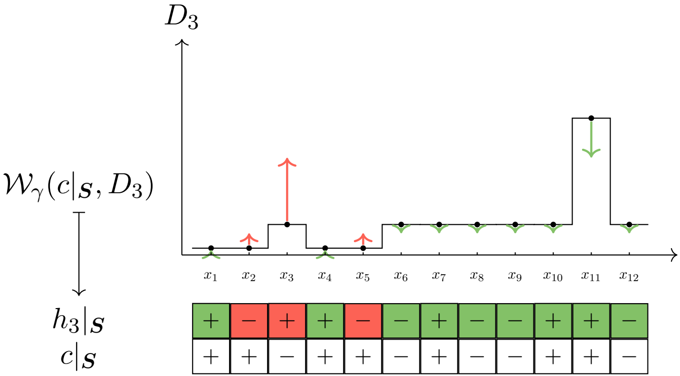

D_2
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_2)$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
$h_2|_{\boldsymbol{S}}$
+ + − + + − − + − − − +
$c|_{\boldsymbol{S}}$
+ + − + + − + − − + + −

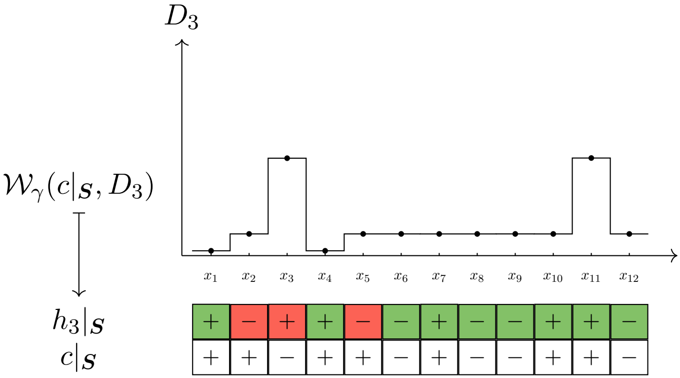

D_2
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_2)$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
$h_2|_{\boldsymbol{S}}$: + + − + + − − + − − − +
$c|_{\boldsymbol{S}}$: + + − + + − + − − + + −



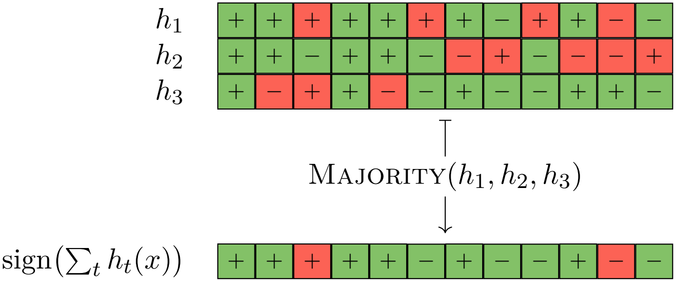
D_3
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_3)$
$c|_{\boldsymbol{S}}$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
+ + − + + − + − − + + −


D_3
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_3)$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
$h_3|_{\boldsymbol{S}}$
+ − + + − − + − − + + −
$c|_{\boldsymbol{S}}$
+ + − + + − + − − + + −


D_3
$\mathcal{W}_\gamma(c|_S, D_3)$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
$h_3|_S$
$c|_S$
+ − + + − − + − − + + −
+ + − + + − + − − + + −


D_3
$\mathcal{W}_\gamma(c|_{\boldsymbol{S}}, D_3)$
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}
$h_3|_{\boldsymbol{S}}$
+ − + + − − + − − + + −
$c|_{\boldsymbol{S}}$
+ + − + + − + − − + + −

AdaBoost: Aggregation



- State-of-the-art in practice:

Boosting in practice

- State-of-the-art in practice:
  - Gradient boosters: e.g., XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017);

- State-of-the-art in practice:
  - Gradient boosters: e.g., XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017);
  - Base learners: low to medium depth decision trees;

- State-of-the-art in practice:
  - Gradient boosters: e.g., XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017);
  - Base learners: low to medium depth decision trees;
  - Often win kaggle™ competitions with small datasets and/or tabular data.

- State-of-the-art in practice:
  - Gradient boosters: e.g., XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017);
  - Base learners: low to medium depth decision trees;
  - Often win kaggle™ competitions with small datasets and/or tabular data.
- Drawbacks:

- State-of-the-art in practice:
  - Gradient boosters: e.g., XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017);
  - Base learners: low to medium depth decision trees;
  - Often win kaggle™ competitions with small datasets and/or tabular data.
- Drawbacks:
  - Achieving the best performance often takes 1000s of iterations.

- State-of-the-art in practice:
  - Gradient boosters: e.g., XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017);
  - Base learners: low to medium depth decision trees;
  - Often win kaggle™ competitions with small datasets and/or tabular data.
- Drawbacks:
  - Achieving the best performance often takes 1000s of iterations.
  - Sequential nature: even with many computers available, it's not obvious how to speed it up.

- State-of-the-art in practice:
  - Gradient boosters: e.g., XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017);
  - Base learners: low to medium depth decision trees;
  - Often win kaggle™ competitions with small datasets and/or tabular data.
- Drawbacks:
  - Achieving the best performance often takes 1000s of iterations.
  - Sequential nature: even with many computers available, it's not obvious how to speed it up.
  - Infeasible for large datasets or "expensive" base learners.

Class of parallel Boosting algorithms considered

1
2
3
4
5

Class of parallel Boosting algorithms considered

Input : training examples $c|\mathbf{s}$, γ-weak learner $\mathcal{W}_\gamma$

Parallelizing Boosting: Formal framework and previous works

Class of parallel Boosting algorithms considered

```
Input   : training examples c|S, γ-weak learner W_γ
for p ← 1 to P do
```

Input : training examples $c|\mathbf{S}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**

Class of parallel Boosting algorithms considered

Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**

  parallel for $t \leftarrow 1$ **to** T **do**

    $h_{p,t} \leftarrow$ Query $\mathcal{W}_\gamma$ with some distribution $D_{p,t}$

Class of parallel Boosting algorithms considered

Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

```
for p ← 1 to P do
    parallel for t ← 1 to T do
        h_{p,t} ← Query W_γ with some distribution D_{p,t}
    ... // Do something with hypotheses found so far
```

Class of parallel Boosting algorithms considered

Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**

  parallel for $t \leftarrow 1$ **to** T **do**

    $h_{p,t} \leftarrow$ Query $\mathcal{W}_\gamma$ with some distribution $D_{p,t}$

  ... `// Do something with hypotheses found so far`

return Classifier H with generalization error not much worse than AdaBoost's ($\tilde{O}\left(\frac{d}{\gamma^2 \cdot |\boldsymbol{S}|}\right)$, with $d = \mathrm{VC}($“base classifiers”$))$

Class of parallel Boosting algorithms considered

Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**

  parallel for $t \leftarrow 1$ **to** T **do**

    $h_{p,t} \leftarrow$ Query $\mathcal{W}_\gamma$ with some distribution $D_{p,t}$

  ... `// Do something with hypotheses found so far`

return Classifier H with generalization error not much worse than AdaBoost's ($\tilde{\mathrm{O}}\left(\frac{d}{\gamma^2 \cdot |\boldsymbol{S}|}\right)$, with $d = \mathrm{VC}(\text{“base classifiers”})$)

- E.g., AdaBoost: $P = \Theta(\frac{\ln|\boldsymbol{S}|}{\gamma^2})$ and $T = 1$ (no parallelism).

Class of parallel Boosting algorithms considered

Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**
  parallel for $t \leftarrow 1$ **to** T **do**
    $h_{p,t} \leftarrow$ Query $\mathcal{W}_\gamma$ with some distribution $D_{p,t}$
  ... // Do something with hypotheses found so far

return Classifier H with generalization error not much worse than AdaBoost's ($\tilde{\mathrm{O}}\left(\frac{d}{\gamma^2 \cdot |\boldsymbol{S}|}\right)$, with $d = \mathrm{VC}(\text{"base classifiers"})$)

- E.g., AdaBoost: $P = \Theta(\frac{\ln|\boldsymbol{S}|}{\gamma^2})$ and $T = 1$ (no parallelism).
- Karbasi and Larsen (2024):

Class of parallel Boosting algorithms considered

Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**

  parallel for $t \leftarrow 1$ **to** T **do**

    $h_{p,t} \leftarrow$ Query $\mathcal{W}_\gamma$ with some distribution $D_{p,t}$

  … `// Do something with hypotheses found so far`

return Classifier H with generalization error not much worse than AdaBoost's ($\tilde{\mathrm{O}}\left(\frac{d}{\gamma^2 \cdot |\boldsymbol{S}|}\right)$, with $d = \mathrm{VC}($"base classifiers"$))$

- E.g., AdaBoost: $P = \Theta(\frac{\ln|\boldsymbol{S}|}{\gamma^2})$ and $T = 1$ (no parallelism).
- Karbasi and Larsen (2024):
  - Boosting algorithm with $P = 1$ and $T = \exp\left(\mathrm{O}\left(\frac{d \ln m}{\gamma^2}\right)\right)$.

Class of parallel Boosting algorithms considered

Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**

  parallel for $t \leftarrow 1$ **to** T **do**

    $h_{p,t} \leftarrow$ Query $\mathcal{W}_\gamma$ with some distribution $D_{p,t}$

  $\ldots$ `// Do something with hypotheses found so far`

return Classifier H with generalization error not much worse than AdaBoost's ($\tilde{\mathrm{O}}\left(\frac{d}{\gamma^2 \cdot |\boldsymbol{S}|}\right)$, with $d = \mathrm{VC}(\text{“base classifiers”})$)

- E.g., AdaBoost: $P = \Theta(\frac{\ln |\boldsymbol{S}|}{\gamma^2})$ and $T = 1$ (no parallelism).
- Karbasi and Larsen (2024):
  - Boosting algorithm with $P = 1$ and $T = \exp\left(\mathrm{O}\left(\frac{d \ln m}{\gamma^2}\right)\right)$.
  - Lower bound: any significant parallelization of Boosting requires exponential total work.

Class of parallel Boosting algorithms considered

```
Input  : training examples c|S, γ-weak learner W_γ
for p ← 1 to P do
    parallel for t ← 1 to T do
        h_{p,t} ← Query W_γ with some distribution D_{p,t}
    ... // Do something with hypotheses found so far
return Classifier H with generalization error not much worse than AdaBoost's (Õ(d/(γ^2·|S|)), with
    d = VC("base classifiers"))
```

- E.g., AdaBoost: $P = \Theta(\frac{\ln|\boldsymbol{S}|}{\gamma^2})$ and $T = 1$ (no parallelism).
- Karbasi and Larsen (2024):
    - Boosting algorithm with $P = 1$ and $T = \exp\big(\mathrm{O}(\frac{d \ln m}{\gamma^2})\big)$.
    - Lower bound: any significant parallelization of Boosting requires exponential total work.
- Lyu, Wu and Yang (2024):

Parallelizing Boosting: Formal framework and previous works

Class of parallel Boosting algorithms considered

Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**

  parallel for $t \leftarrow 1$ **to** T **do**

    $h_{p,t} \leftarrow$ Query $\mathcal{W}_\gamma$ with some distribution $D_{p,t}$

  ... `// Do something with hypotheses found so far`

return Classifier H with generalization error not much worse than AdaBoost's ($\tilde{\mathrm{O}}\left(\frac{d}{\gamma^2 \cdot |\boldsymbol{S}|}\right)$, with $d = \mathrm{VC}(\text{“base classifiers”})$)

- E.g., AdaBoost: $P = \Theta(\frac{\ln|\boldsymbol{S}|}{\gamma^2})$ and $T = 1$ (no parallelism).
- Karbasi and Larsen (2024):
  - Boosting algorithm with $P = 1$ and $T = \exp\left(\mathrm{O}\left(\frac{d \ln m}{\gamma^2}\right)\right)$.
  - Lower bound: any significant parallelization of Boosting requires exponential total work.
- Lyu, Wu and Yang (2024):
  - $P = \mathrm{O}\left(\frac{\ln m}{\gamma^2 R}\right)$ and $T = \exp\left(\mathrm{O}\left(dR^2\right)\right) \cdot \ln \frac{1}{\gamma}$.

Class of parallel Boosting algorithms considered

Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**

  parallel for $t \leftarrow 1$ **to** T **do**

    $h_{p,t} \leftarrow$ Query $\mathcal{W}_\gamma$ with some distribution $D_{p,t}$

  ... `// Do something with hypotheses found so far`

return Classifier H with generalization error not much worse than AdaBoost's ($\tilde{\mathrm{O}}\left(\frac{d}{\gamma^2 \cdot |\boldsymbol{S}|}\right)$, with $d = \mathrm{VC}(\text{“base classifiers”})$)

- E.g., AdaBoost: $P = \Theta(\frac{\ln|\boldsymbol{S}|}{\gamma^2})$ and $T = 1$ (no parallelism).
- Karbasi and Larsen (2024):
  - Boosting algorithm with $P = 1$ and $T = \exp\left(\mathrm{O}\left(\frac{d \ln m}{\gamma^2}\right)\right)$.
  - Lower bound: any significant parallelization of Boosting requires exponential total work.
- Lyu, Wu and Yang (2024):
  - $P = \mathrm{O}\left(\frac{\ln m}{\gamma^2 R}\right)$ and $T = \exp\left(\mathrm{O}\left(dR^2\right)\right) \cdot \ln \frac{1}{\gamma}$.
  - Improved lower bound.

Parallelizing Boosting: Formal framework and previous works

Class of parallel Boosting algorithms considered

```
Input : training examples c|S, γ-weak learner W_γ
for p ← 1 to P do
    parallel for t ← 1 to T do
        h_{p,t} ← Query W_γ with some distribution D_{p,t}
    ... // Do something with hypotheses found so far
return Classifier H with generalization error not much worse than AdaBoost's (Õ(d/(γ²·|S|)), with
  d = VC("base classifiers"))
```

Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**

- **parallel for** $t \leftarrow 1$ **to** T **do**
  - $h_{p,t} \leftarrow$ Query $\mathcal{W}_\gamma$ with some distribution $D_{p,t}$
- ... // Do something with hypotheses found so far

return Classifier H with generalization error not much worse than AdaBoost's ($\tilde{\mathrm{O}}\left(\frac{d}{\gamma^2 \cdot |\boldsymbol{S}|}\right)$, with $d = \mathrm{VC}(\text{"base classifiers"})$)

- E.g., AdaBoost: $P = \Theta(\frac{\ln|\boldsymbol{S}|}{\gamma^2})$ and $T = 1$ (no parallelism).
- Karbasi and Larsen (2024):
  - Boosting algorithm with $P = 1$ and $T = \exp\left(\mathrm{O}\left(\frac{d \ln m}{\gamma^2}\right)\right)$.
  - Lower bound: any significant parallelization of Boosting requires exponential total work.
- Lyu, Wu and Yang (2024):
  - $P = \mathrm{O}\left(\frac{\ln m}{\gamma^2 R}\right)$ and $T = \exp\left(\mathrm{O}\left(dR^2\right)\right) \cdot \ln \frac{1}{\gamma}$.
  - Improved lower bound.
- A gap remains

Parallelizing Boosting: Formal framework and previous works

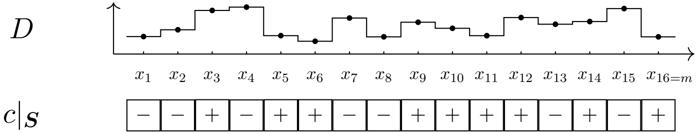
Class of parallel Boosting algorithms considered

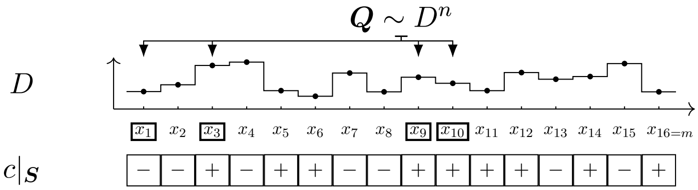
Input : training examples $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

for $p \leftarrow 1$ **to** P **do**
  parallel for $t \leftarrow 1$ **to** T **do**
    $h_{p,t} \leftarrow$ Query $\mathcal{W}_\gamma$ with some distribution $D_{p,t}$
  ... `// Do something with hypotheses found so far`

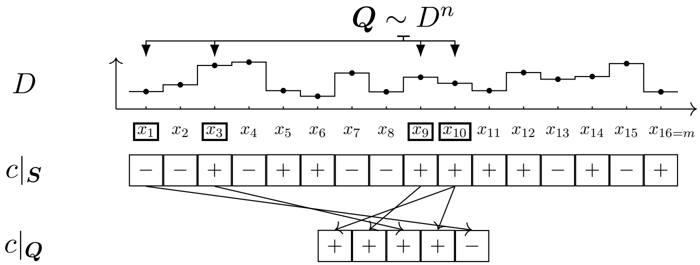
return Classifier H with generalization error not much worse than AdaBoost's ($\tilde{\mathrm{O}}\left(\frac{d}{\gamma^2 \cdot |\boldsymbol{S}|}\right)$, with $d = \mathrm{VC}(\text{"base classifiers"})$)

- E.g., AdaBoost: $P = \Theta(\frac{\ln|\boldsymbol{S}|}{\gamma^2})$ and $T = 1$ (no parallelism).
- Karbasi and Larsen (2024):
  - Boosting algorithm with $P = 1$ and $T = \exp\left(\mathrm{O}\left(\frac{d \ln m}{\gamma^2}\right)\right)$.
  - Lower bound: any significant parallelization of Boosting requires exponential total work.
- Lyu, Wu and Yang (2024):
  - $P = \mathrm{O}\left(\frac{\ln m}{\gamma^2 R}\right)$ and $T = \exp\left(\mathrm{O}\left(dR^2\right)\right) \cdot \ln \frac{1}{\gamma}$.
  - Improved lower bound.
- A gap remains: **this work closes it.**

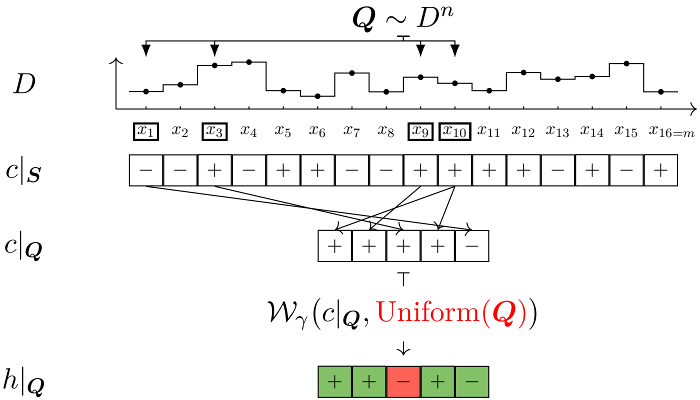

D
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x11 x12 x13 x14 x15 x16=m
c|S
− − + − + + − − + + + + − + − +

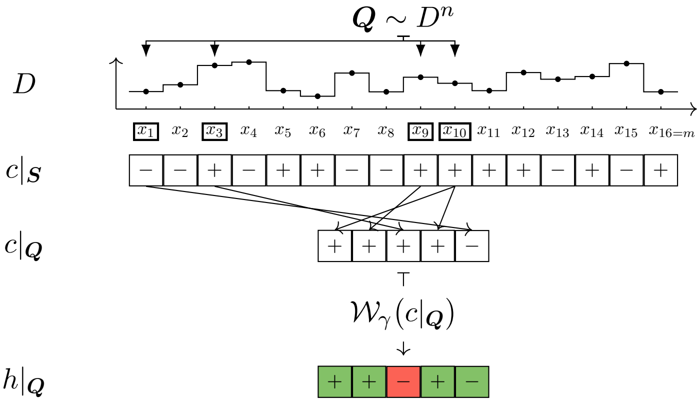

$\boldsymbol{Q} \sim D^n$
D
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13} x_{14} x_{15} $x_{16=m}$
$c|\boldsymbol{S}$
− − + − + + − − + + + + − + − +

Parallelizing Boosting: subsampling step

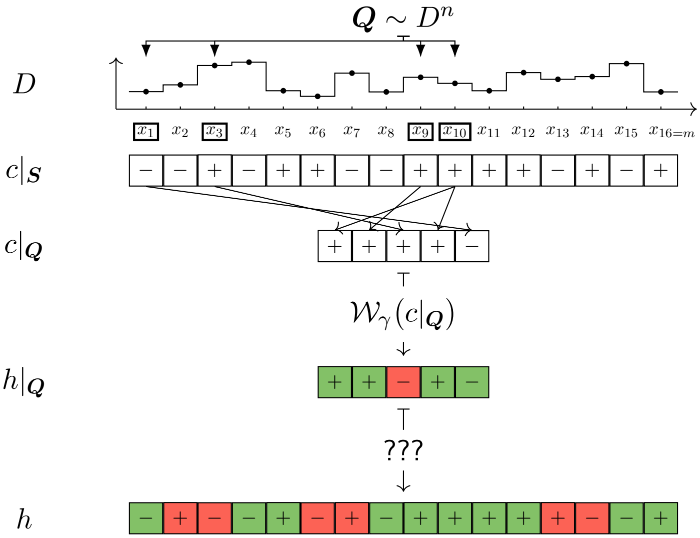


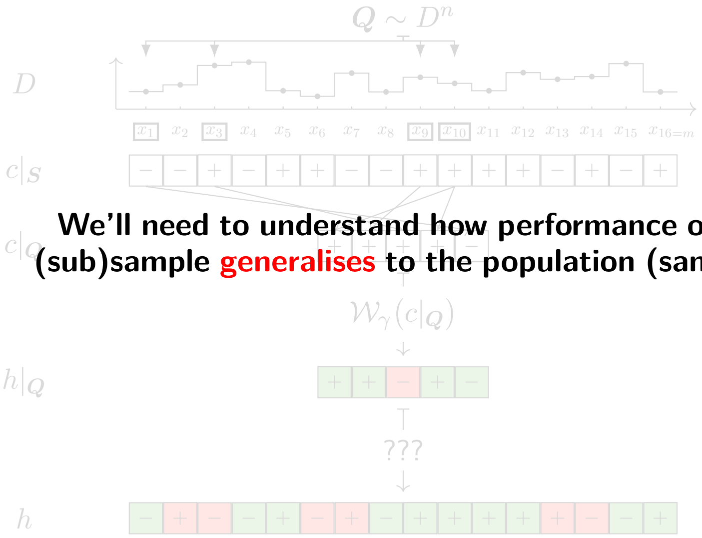
Parallelizing Boosting: subsampling step




$\boldsymbol{Q} \sim D^n$
D
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13} x_{14} x_{15} $x_{16=m}$
$c|_{\boldsymbol{S}}$
− − + − + + − − + + + + − + − +
$c|_{\boldsymbol{Q}}$
+ + + + −
$\mathcal{W}_\gamma(c|_{\boldsymbol{Q}})$
$h|_{\boldsymbol{Q}}$
+ + − + −

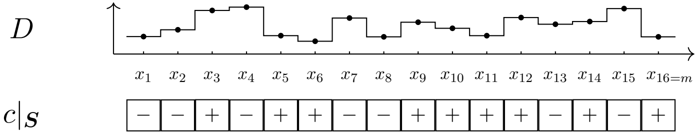
Parallelizing Boosting: subsampling step

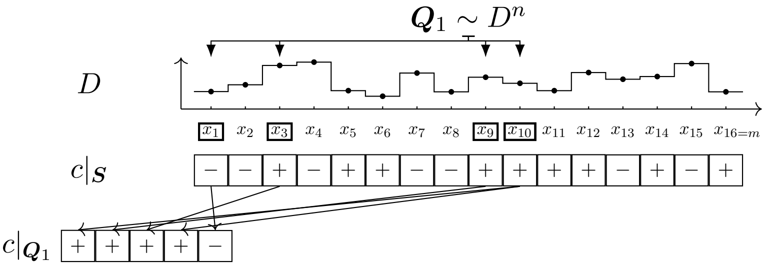




We'll need to understand how performance on a (sub)sample generalises to the population (sample)

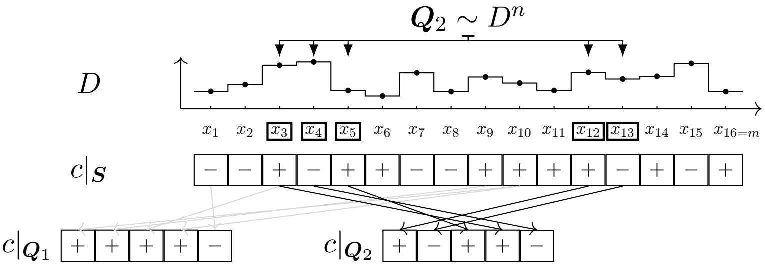
Parallelizing Boosting: bagging step



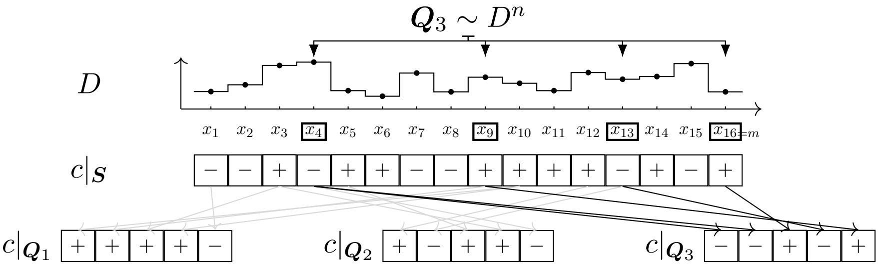


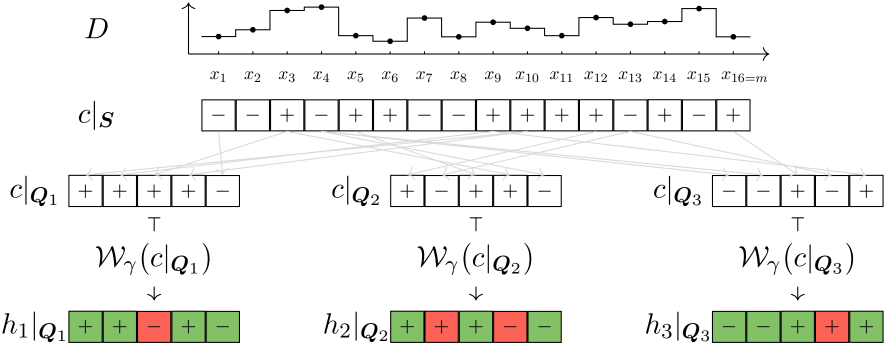
$Q_1 \sim D^n$
D
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13} x_{14} x_{15} $x_{16=m}$
$c|_S$
− − + − + + − − + + + + − + − +
$c|_{Q_1}$
+ + + + −

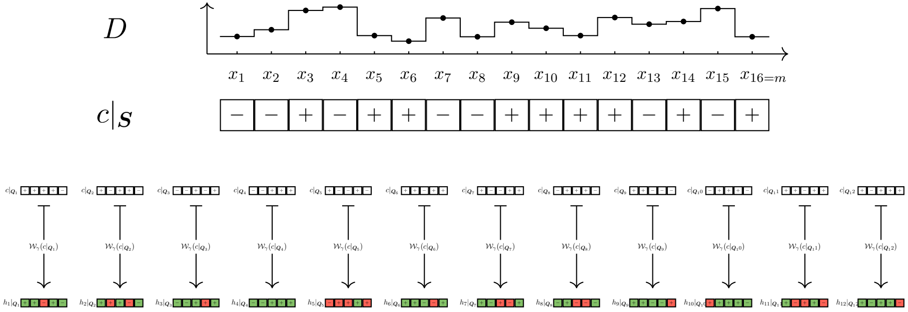
Parallelizing Boosting: bagging step

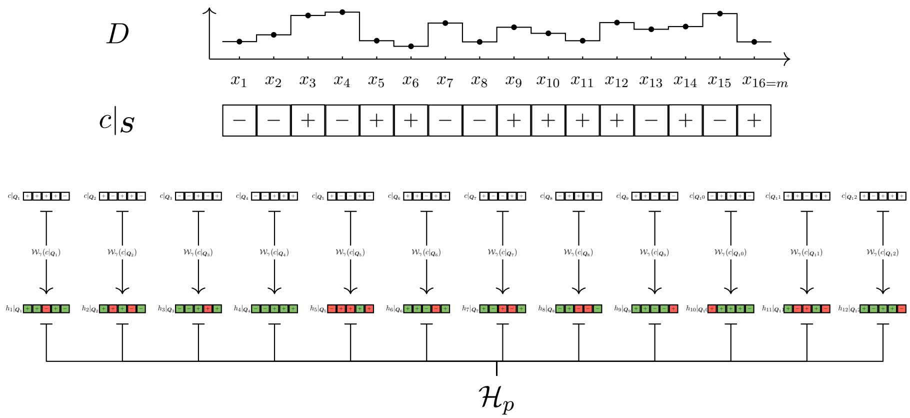


Parallelizing Boosting: bagging step




D
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x11 x12 x13 x14 x15 x16=m
c|S
− − + − + + − − + + + + − + − +
c|Q1 + + + + −
c|Q2 + − + + −
c|Q3 − − + − +
Wγ(c|Q1)
Wγ(c|Q2)
Wγ(c|Q3)
h1|Q1 + + − + −
h2|Q2 + + + − −
h3|Q3 − − + + +


D
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13} x_{14} x_{15} $x_{16=m}$
$c|_S$
− − + − + + − − + + + + − + − +



Bag of hypotheses used to perform multiple boosting steps

Input : Training data $c|_{\mathbf{S}}$,γ-weak learner $\mathcal{W}_\gamma$

Parallel Boosting Algorithm: Sketch

Input : Training data $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

$\boldsymbol{D}_1 \leftarrow$ Uniform distribution over the m examples

Input : Training data $c|_S$, γ-weak learner $\mathcal{W}_\gamma$

$D_1 \leftarrow$ Uniform distribution over the m examples

for $p \leftarrow 1$ **to** P **do**

Parallel Boosting Algorithm: Sketch

Input : Training data $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

$D_1 \leftarrow$ Uniform distribution over the m examples

for $p \leftarrow 1$ **to** P **do**

    // Bagging step

    $\mathcal{H}_p \leftarrow \emptyset$ // Bag of weak hypotheses

Input : Training data $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

$D_1 \leftarrow$ Uniform distribution over the m examples

for $p \leftarrow 1$ **to** P **do**

  // Bagging step

  $\boldsymbol{\mathcal{H}}_p \leftarrow \emptyset$ // Bag of weak hypotheses

  parallel for $t \leftarrow 1$ **to** T **do**

Parallel Boosting Algorithm: Sketch

Input : Training data $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

$D_1 \leftarrow$ Uniform distribution over the m examples

for $p \leftarrow 1$ **to** P **do**

    // Bagging step

    $\boldsymbol{\mathcal{H}}_p \leftarrow \emptyset$ // Bag of weak hypotheses

    parallel for $t \leftarrow 1$ **to** T **do**

        $\mathbf{h} \leftarrow$ Query $\mathcal{W}_\gamma$ on subsample following the current distribution $(\boldsymbol{D}_{(p-1)R+1})$

        Add $\mathbf{h}$ to $\boldsymbol{\mathcal{H}}_p$

```
Input   : Training data c|S, γ-weak learner W_γ
D_1 ← Uniform distribution over the m examples
for p ← 1 to P do
    // Bagging step
    H_p ← ∅                                            // Bag of weak hypotheses
    parallel for t ← 1 to T do
        h ← Query W_γ on subsample following the current distribution (D_{(p−1)R+1})
        Add h to H_p
    // Boosting steps
```

Input : Training data $c|_S$, γ-weak learner $\mathcal{W}_\gamma$

$D_1 \leftarrow$ Uniform distribution over the m examples

for $p \leftarrow 1$ **to** P **do**

  // Bagging step

  $\mathcal{H}_p \leftarrow \emptyset$ // Bag of weak hypotheses

  parallel **for** $t \leftarrow 1$ **to** T **do**

    $\mathbf{h} \leftarrow$ Query $\mathcal{W}_\gamma$ on subsample following the current distribution $(D_{(p-1)R+1})$

    Add $\mathbf{h}$ to $\mathcal{H}_p$

  // Boosting steps

  for $r \leftarrow 1$ **to** R **do**

    $\mathbf{h}_{(p-1)R+r} \leftarrow$ Simulate $\frac{\gamma}{2}$-weak learner: search $\boldsymbol{\mathcal{H}}_p$ for h s.t. $\mathrm{err}_{\boldsymbol{D}_{(p-1)R+r}}(h) \leq \frac{1}{2} - \frac{\gamma}{2}$

Parallel Boosting Algorithm: Sketch

Input : Training data $c|_S$, γ-weak learner $\mathcal{W}_\gamma$

$D_1 \leftarrow$ Uniform distribution over the m examples

for $p \leftarrow 1$ **to** P **do**

  // Bagging step

  $\mathcal{H}_p \leftarrow \emptyset$ // Bag of weak hypotheses

  parallel for $t \leftarrow 1$ **to** T **do**

    $\mathbf{h} \leftarrow$ Query $\mathcal{W}_\gamma$ on subsample following the current distribution $(D_{(p-1)R+1})$

    Add $\mathbf{h}$ to $\mathcal{H}_p$

  // Boosting steps

  for $r \leftarrow 1$ **to** R **do**

    $\mathbf{h}_{(p-1)R+r} \leftarrow$ Simulate $\frac{\gamma}{2}$-weak learner: search $\boldsymbol{\mathcal{H}}_p$ for h s.t. $\mathrm{err}_{\boldsymbol{D}_{(p-1)R+r}}(h) \leq \frac{1}{2} - \frac{\gamma}{2}$

    ... // Omitted details

Input : Training data $c|_S$, γ-weak learner $\mathcal{W}_\gamma$

$D_1 \leftarrow$ Uniform distribution over the m examples

for $p \leftarrow 1$ **to** P **do**

  // Bagging step

  $\mathcal{H}_p \leftarrow \emptyset$ // Bag of weak hypotheses

  parallel for $t \leftarrow 1$ **to** T **do**

    $\mathrm{h} \leftarrow$ Query $\mathcal{W}_\gamma$ on subsample following the current distribution $(\boldsymbol{D}_{(p-1)R+1})$

    Add h to $\mathcal{H}_p$

  // Boosting steps

  for $r \leftarrow 1$ **to** R **do**

    $\mathbf{h}_{(p-1)R+r} \leftarrow$ Simulate $\frac{\gamma}{2}$-weak learner: search $\boldsymbol{\mathcal{H}}_p$ for h s.t. $\mathrm{err}_{\boldsymbol{D}_{(p-1)R+r}}(h) \leq \frac{1}{2} - \frac{\gamma}{2}$

    ... // Omitted details

    $\boldsymbol{D}_{(p-1)R+r+1} \leftarrow$ Usual "AdaBoost update" of $\boldsymbol{D}_{(p-1)R+r}$

```
Input   : Training data c|_S, γ-weak learner W_γ
D_1 ← Uniform distribution over the m examples
for p ← 1 to P do
  | // Bagging step
  | H_p ← ∅                                          // Bag of weak hypotheses
  | parallel for t ← 1 to T do
  |  | h ← Query W_γ on subsample following the current distribution (D_{(p−1)R+1})
  |  | Add h to H_p
  | // Boosting steps
  | for r ← 1 to R do
  |  | h_{(p−1)R+r} ← Simulate γ/2-weak learner: search H_p for h s.t. err_{D_{(p−1)R+r}}(h) ≤ 1/2 − γ/2
  |  | . . . // Omitted details
  |  | D_{(p−1)R+r+1} ← Usual "AdaBoost update" of D_{(p−1)R+r}
return Majority aggregation of h_1, h_2, . . . , h_{PR}
```

Parallel Boosting Algorithm: Sketch

Input : Training data $c|_{\mathbf{S}}$, γ-weak learner $\mathcal{W}_\gamma$

$\boldsymbol{D}_1 \leftarrow$ Uniform distribution over the m examples

for $p \leftarrow 1$ **to** P **do**

    // Bagging step

    $\boldsymbol{\mathcal{H}}_p \leftarrow \emptyset$ // Bag of weak hypotheses

    parallel for $t \leftarrow 1$ **to** T **do**

        $\mathbf{h} \leftarrow$ Query $\mathcal{W}_\gamma$ on subsample following the current distribution $(\boldsymbol{D}_{(p-1)R+1})$

        Add $\mathbf{h}$ to $\boldsymbol{\mathcal{H}}_p$

    // Boosting steps

    for $r \leftarrow 1$ **to** R **do**

        $\mathbf{h}_{(p-1)R+r} \leftarrow$ Simulate $\frac{\gamma}{2}$-weak learner: search $\boldsymbol{\mathcal{H}}_p$ for h s.t. $\mathrm{err}_{\boldsymbol{D}_{(p-1)R+r}}(h) \leq \frac{1}{2} - \frac{\gamma}{2}$

        ... // Omitted details

        $\boldsymbol{D}_{(p-1)R+r+1} \leftarrow$ Usual "AdaBoost update" of $\boldsymbol{D}_{(p-1)R+r}$

return Majority aggregation of $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{PR}$

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_p$: hypotheses trained on samples from $\boldsymbol{D}_{(p-1)R+1}$.

Proof strategy (assuming $p = 1$ for simplicity)

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_1$: hypotheses trained on samples from $\boldsymbol{D}_1$.

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_1$: hypotheses trained on samples from $\boldsymbol{D}_1$.
- Easy to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ for the first step.

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_1$: hypotheses trained on samples from $\boldsymbol{D}_1$.
- Easy to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ for the first step.
- Then we change the distribution but not the bag.

Proof strategy (assuming $p = 1$ for simplicity)

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_1$: hypotheses trained on samples from $\boldsymbol{D}_1$.
- Easy to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ for the first step.
- Then we change the distribution but not the bag.

Solution

- Track the divergence from starting point: $\mathrm{KL}(D_r \parallel D_1)$.

Proof strategy (assuming $p = 1$ for simplicity)

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_1$: hypotheses trained on samples from $\boldsymbol{D}_1$.
- Easy to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ for the first step.
- Then we change the distribution but not the bag.

Solution

- Track the divergence from starting point: $\mathrm{KL}(D_r \parallel D_1)$.
- Low divergence:

Proof strategy (assuming $p = 1$ for simplicity)

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_1$: hypotheses trained on samples from $\boldsymbol{D}_1$.
- Easy to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ for the first step.
- Then we change the distribution but not the bag.

Solution

- Track the divergence from starting point: $\mathrm{KL}(D_r \parallel D_1)$.
- Low divergence:
  - Lemma: likely to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$.

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_1$: hypotheses trained on samples from $\boldsymbol{D}_1$.
- Easy to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ for the first step.
- Then we change the distribution but not the bag.

Solution

- Track the divergence from starting point: $\mathrm{KL}(D_r \parallel D_1)$.
- Low divergence:
  - Lemma: likely to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$.
- Large divergence:

Proof strategy (assuming $p = 1$ for simplicity)

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_1$: hypotheses trained on samples from $\boldsymbol{D}_1$.
- Easy to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ for the first step.
- Then we change the distribution but not the bag.

Solution

- Track the divergence from starting point: $\mathrm{KL}(D_r \parallel D_1)$.
- Low divergence:
  - Lemma: likely to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$.
- Large divergence:
  - Hard to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$.

Proof strategy (assuming $p = 1$ for simplicity)

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_1$: hypotheses trained on samples from $\boldsymbol{D}_1$.
- Easy to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ for the first step.
- Then we change the distribution but not the bag.

Solution

- Track the divergence from starting point: $\mathrm{KL}(D_r \parallel D_1)$.
- Low divergence:
  - Lemma: likely to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$.
- Large divergence:
  - Hard to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$.
  - Perhaps some $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ is so bad that $-\mathbf{h}$ is good.

Proof strategy (assuming $p = 1$ for simplicity)

Main challenge

- Bag $\boldsymbol{\mathcal{H}}_1$: hypotheses trained on samples from $\boldsymbol{D}_1$.
- Easy to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ for the first step.
- Then we change the distribution but not the bag.

Solution

- Track the divergence from starting point: $\mathrm{KL}(D_r \parallel D_1)$.
- Low divergence:
  - Lemma: likely to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$.
- Large divergence:
  - Hard to find good $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$.
  - Perhaps some $\mathbf{h} \in \boldsymbol{\mathcal{H}}_1$ is so bad that $-\mathbf{h}$ is good.
  - Otherwise already made enough progress: early stopping.

Recall:

- $T :=$ Number of parallel calls to $\mathcal{W}_\gamma$.
- $R :=$ Number of steps of "simulated" $\gamma/2$-weak learner.
- $P :=$ Number of iterations of those.
- $d := \mathrm{VC}($"base classifiers"$)$.

Results

Recall:

- $T :=$ Number of parallel calls to $\mathcal{W}_\gamma$.
- $R :=$ Number of steps of "simulated" $\gamma/2$-weak learner.
- $P :=$ Number of iterations of those.
- $d := \mathrm{VC}($"base classifiers"$)$.

Results

- Given $R \in \mathbb{N}$,

Recall:

- $T :=$ Number of parallel calls to $\mathcal{W}_\gamma$.
- $R :=$ Number of steps of "simulated" $\gamma/2$-weak learner.
- $P :=$ Number of iterations of those.
- $d := \mathrm{VC}($"base classifiers"$)$.

Results

- Given $R \in \mathbb{N}$,
  - With high probability, the algorithm described performs well (generalization error no worse than AdaBoost's).

Contributions

Recall:

- $T :=$ Number of parallel calls to $\mathcal{W}_\gamma$.
- $R :=$ Number of steps of "simulated" $\gamma/2$-weak learner.
- $P :=$ Number of iterations of those.
- $d := \mathrm{VC}(\text{"base classifiers"})$.

Results

- Given $R \in \mathbb{N}$,
  - With high probability, the algorithm described performs well (generalization error no worse than AdaBoost's).
  - Satisfies
    - $P = \mathrm{O}\left(\frac{\ln|\boldsymbol{S}|}{\gamma^2 \cdot R}\right)$,
    - $T = e^{\mathrm{O}(d \cdot R)}$.

Recall:

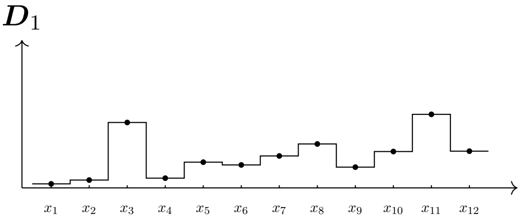
- $T :=$ Number of parallel calls to $\mathcal{W}_\gamma$.
- $R :=$ Number of steps of "simulated" $\gamma/2$-weak learner.
- $P :=$ Number of iterations of those.
- $d := \mathrm{VC}($"base classifiers"$)$.

Results

- Given $R \in \mathbb{N}$,
  - With high probability, the algorithm described performs well (generalization error no worse than AdaBoost's).
  - Satisfies
    - $P = \mathrm{O}\left(\frac{\ln|\boldsymbol{S}|}{\gamma^2 \cdot R}\right)$,
    - $T = e^{\mathrm{O}(d \cdot R)}$.
- Matching lower bounds (up to logarithmic factors) for all values of R.

Thank you!

We'll be presenting this work's poster in 20 minutes from now (at West Ballroom A-D).
Come chat with us!

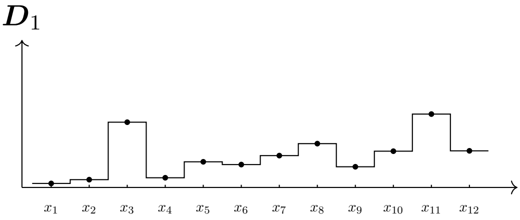


$c|_{\boldsymbol{S}}$

+	−	+	+	−	+	−	−	−	+	+	−

Goal Simulate a $\gamma/2$-weak learner.

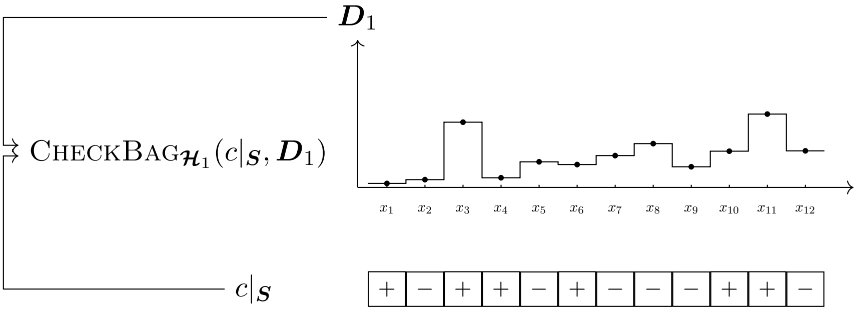
$\textsc{CheckBag}_{\mathcal{H}_1}$



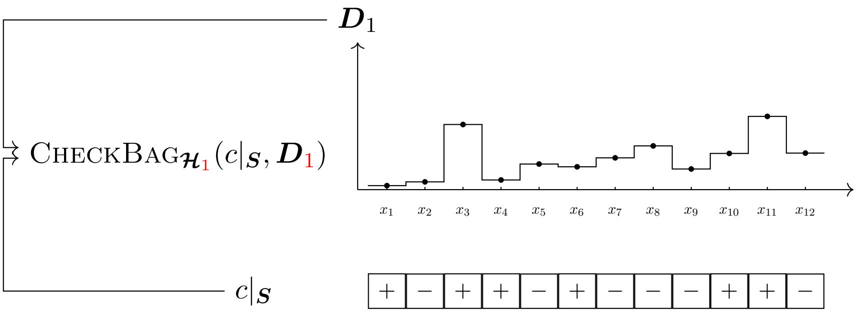
$c|_S$

+	−	+	+	−	+	−	−	−	+	+	−

Goal Simulate a $\gamma/2$-weak learner.

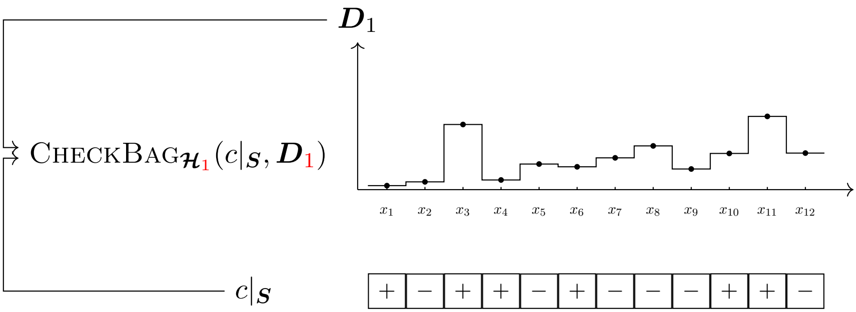


Goal Simulate a $\gamma/2$-weak learner.



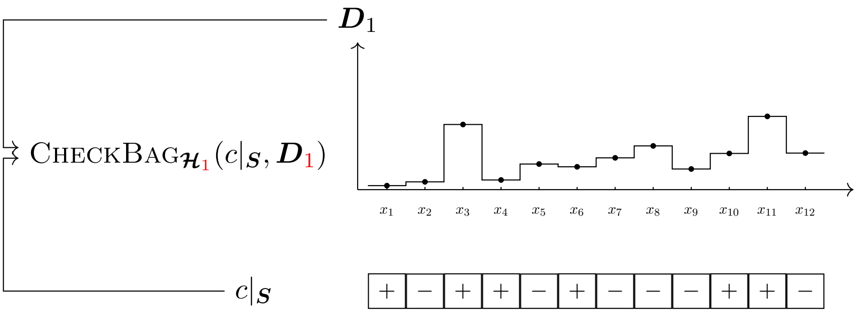
- $\mathcal{H}_1$ contains weak-hypotheses for subsamples following $\boldsymbol{D}_1$.

Goal Simulate a $\gamma/2$-weak learner.



- $\boldsymbol{\mathcal{H}}_1$ contains weak-hypotheses for subsamples following $\boldsymbol{D}_1$.
- Classical LT: large enough ($\mathrm{O}(d/\gamma^2)$) samples are likely to be (γ-)representative

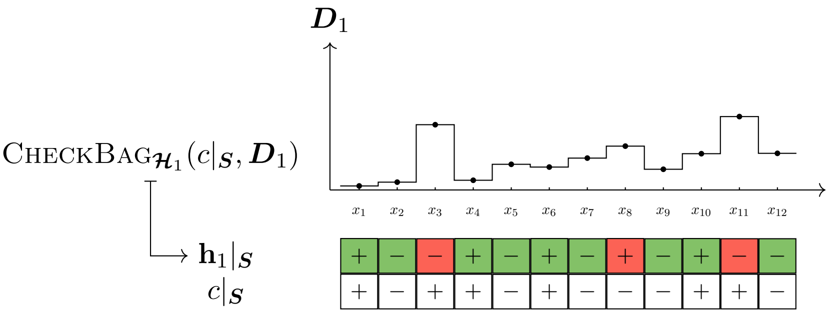
Goal Simulate a $\gamma/2$-weak learner.



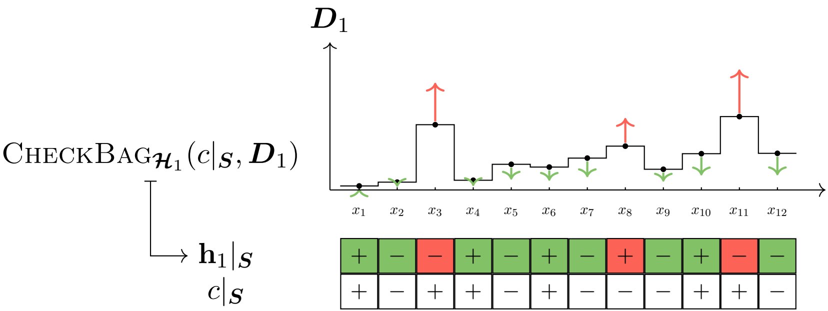
- $\mathcal{H}_1$ contains weak-hypotheses for subsamples following $\boldsymbol{D}_1$.
- Classical LT: large enough $(\mathrm{O}(d/\gamma^2))$ samples are likely to be (γ-)representative:

$$|\mathrm{err}_{\boldsymbol{Q}\sim\boldsymbol{D}_1^n}(h) - \mathrm{err}_{\boldsymbol{D}_1}(h)| < \gamma/2 \quad \text{(with high probability)}$$

Goal Simulate a $\gamma/2$-weak learner.

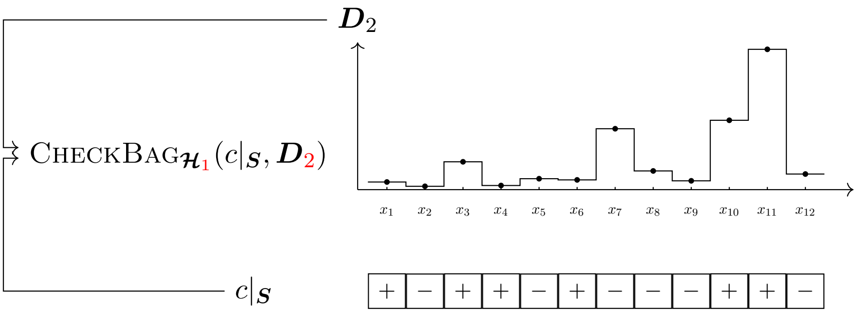


Goal Simulate a $\gamma/2$-weak learner.

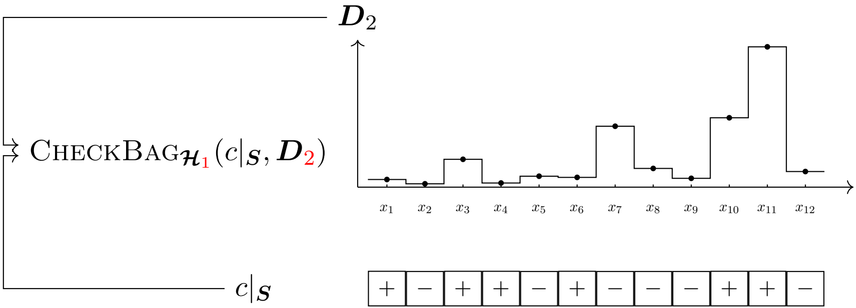


Parallelizing Boosting: Boosting steps

Goal Simulate a $\gamma/2$-weak learner.

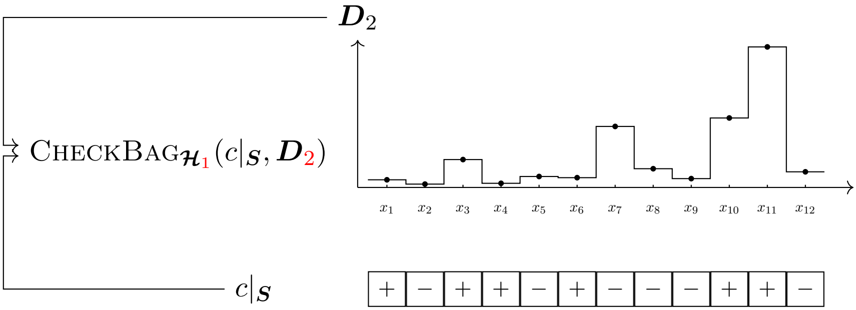


Goal Simulate a $\gamma/2$-weak learner.



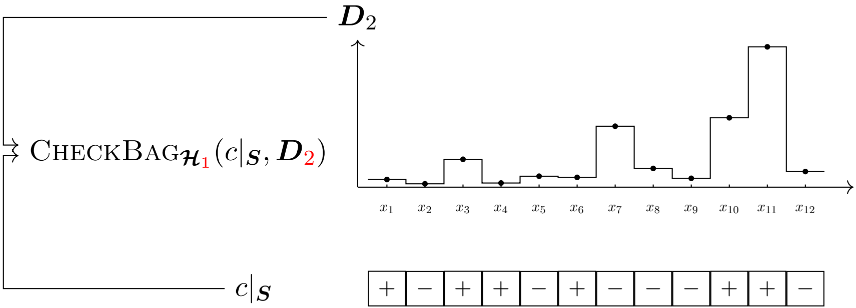
- $\mathcal{H}_1$ contains weak-hypotheses for subsamples following $\boldsymbol{D}_1$.

Goal Simulate a $\gamma/2$-weak learner.



- $\mathcal{H}_1$ contains weak-hypotheses for subsamples following $\boldsymbol{D}_1$.
- Does performance on subsamples from $\boldsymbol{D}_1$ generalise to performance under $\boldsymbol{D}_2$?

Goal Simulate a $\gamma/2$-weak learner.



- $\boldsymbol{\mathcal{H}}_1$ contains weak-hypotheses for subsamples following $\boldsymbol{D}_1$.
- Does performance on subsamples from $\boldsymbol{D}_1$ generalise to performance under $\boldsymbol{D}_2, \boldsymbol{D}_3, \ldots$?

$$|\mathrm{err}_{\boldsymbol{Q}\sim\boldsymbol{D}_1^n}(h) - \mathrm{err}_{\boldsymbol{D}_r}(h)| < ??? \quad \text{(with high probability)}$$

- Chen, Tianqi and Carlos Guestrin (2016). 'XGBoost: A Scalable Tree Boosting System.'. In: *KDD*. ACM, pp. 785–794. ISBN: 978-1-4503-4232-2.
- Karbasi, Amin and Kasper Green Larsen (2024). 'The Impossibility of Parallelizing Boosting'. In: *Proceedings of The 35th International Conference on Algorithmic Learning Theory*. Ed. by Claire Vernade and Daniel Hsu. Vol. 237. Proceedings of Machine Learning Research. PMLR, pp. 635–653. URL: https://proceedings.mlr.press/v237/karbasi24a.html.
- Ke, Guolin et al. (2017). 'LightGBM: A Highly Efficient Gradient Boosting Decision Tree'. In: *NIPS*.
- Lyu, Xin, Hongxun Wu and Junzhao Yang (2024). 'The Cost of Parallelizing Boosting'. In: *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*. Ed. by David P. Woodruff. SIAM, pp. 3140–3155. DOI: 10.1137/1.9781611977912.112. URL: https://doi.org/10.1137/1.9781611977912.112.