Introduction to Machine Learning
TTIC 31020

Prof. Nati Srebro

Lecture 9:

Part I: Gaussian Process and Bayesian Machine Learning

Part II: Stochastic Gradient Descent

Generative Mode:
Assumptions on $P(x|y)$
$P(x,y) \in \{p(x,y;\theta) \mid \theta\}$

Discriminative Probability Model:
Assumption on $P(y|x)$
$P(y|x) = \frac{1}{1+e^{h(x)}} \qquad h \in \{h_w(x)|w\}$
$P(y|x) \in \left\{p(y|x;w) = Ber\left(\frac{1}{1+e^{h_w(x)}}\right) \middle| w\right\}$

Predictive Mode:
Assumption on h^*
$L_S(h)$ small for some
$h \in \{h_w(x)|w\}$

More knowledge
Less data

Less knowledge
More data

Task loss, e.g. L_S^{01}, or conv/tractable surrogate

ERM: $\arg\min_w L_S(h_w)$
Regularized ERM, e.g.: $\arg\min_w L_S(h_w) + \lambda\|w\|_2^2$

Scale sensitive, e.g. $L_S^{marg}, L_S^{hinge}, L_S^{lgstc}$

Fit Generative Model, e.g. via
Max Likelihood: $\arg\max_\theta \log P(\{(x_i,y_i)\}|\theta)$
or MAP: $\arg\max_\theta \log P(\{(x_i,y_i)\}|\theta) + \log P(\theta)$

Fit Discriminative Model, e.g. via
Max *conditional* Likelihood $\arg\max_w \log P(\{y_i\} \mid \{x_i\}; w) = \arg\min_w L_S^{lgst}(h_w)$
or conditional MAP: $\arg\max_w \log P(\{y_i\} \mid \{x_i\}; w) + \log P(w) = \arg\min_w L_S^{lgst}(h_w) + \frac{1}{2m\sigma^2}\|w\|_2^2$

For Gaussian prior, $w \sim N(0, \sigma^2 I)$

$$P(\{y_i\}|\{x_i\},w) = \prod \frac{1}{1+e^{-h_w(x_i)}} \qquad h_w(x_i) = \langle w, \phi(x_i)\rangle \qquad w \sim \mathcal{N}(0, I_D)$$

Can we talk about $P(\{y_i\}|\{x_i\})$, or $h(x)$, and without talking about w or ϕ, allowing $D = \infty$?

$h_w(\cdot)$ is a random function. How is it distributed?

How is $h_w(x_1), h_w(x_2), \ldots, h_w(x_m)$ distributed for some $x_1, \ldots, x_m$?

$$[h_w(x_1), h_w(x_2), \ldots, h_w(x_m)] = [\langle w, \phi(x_1)\rangle, \langle w, \phi(x_2)\rangle, \ldots, \langle w, \phi(x_m)\rangle] = w\Phi \sim \mathcal{N}(0, \Phi^\top \Phi)$$

Or in other words, $h_w(\cdot)$ is a **Gaussian Process**

(a random function where every set of values if jointly Gaussian)

A GP is characterized by it Covariance function (and mean, which we take to be $\mathbb{E}[r(x_i)] = 0$):

$$K(x_i, x_j) = Cov\left(h(x_i), h(x_j)\right) = Cov(\langle w, \phi(x_1)\rangle, \langle w, \phi(x_2)\rangle) = \phi(x_1)^\top Cov(w)\phi(x_2) = \langle \phi(x_1), \phi(x_2)\rangle$$

Gaussian Processes ≈ Kernalized ℓ_2-regularized linear learning

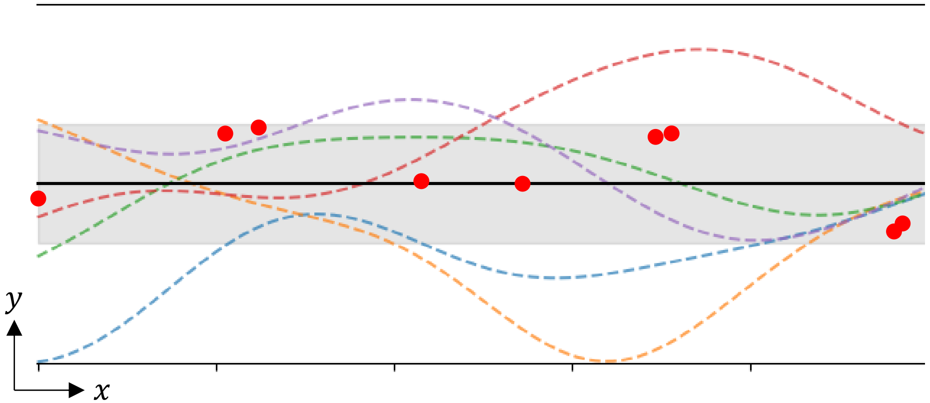
In this case $\mathbb{E}_{h|S}[h] = \arg\max_h P(h|S)$, since mode of Gaussian is its mean

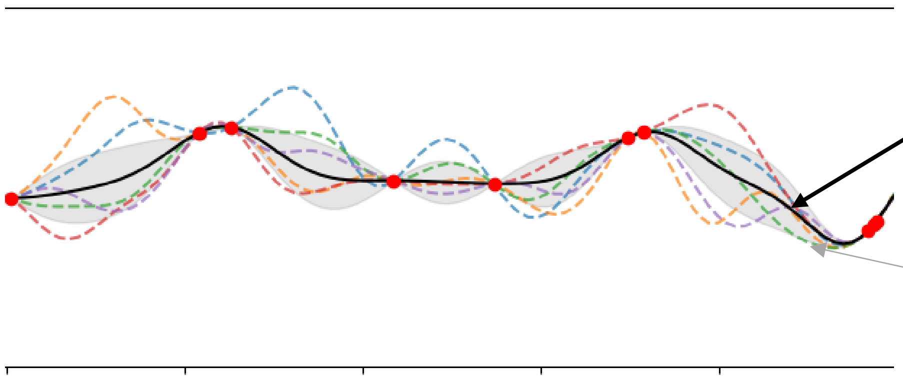
Another view: prior over functions given by Hilbert norm in a Reproducing Kernel Hilbert Space (RKHS)

$$p(h) \propto \exp(-\|h\|_K^2)$$

Beyond scope of course

$$K(x, x') = e^{-|x-x'|^2}$$





MAP: $\max_h P(h | L_S(h) = 0)$

$= \arg\max \|w\| \;\; s.t. \, L_S(h) = 0$

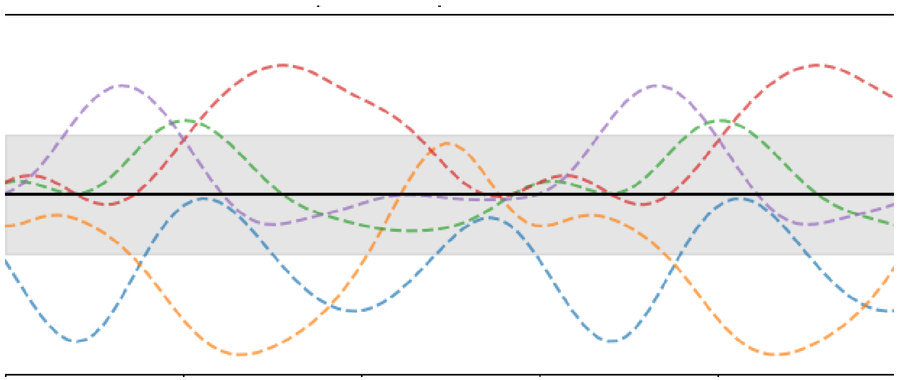
$= \mathbb{E}[h(x) | L_S(h) = 0]$

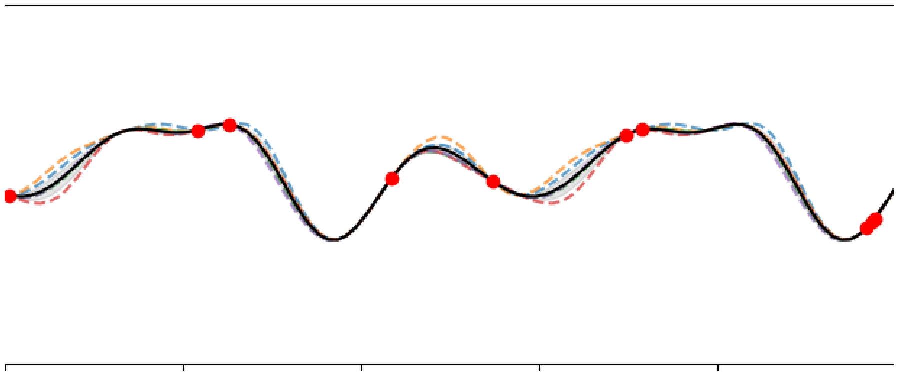
$\sqrt{\mathrm{Var}[h(x) | S]}$

Prior $P(h)$

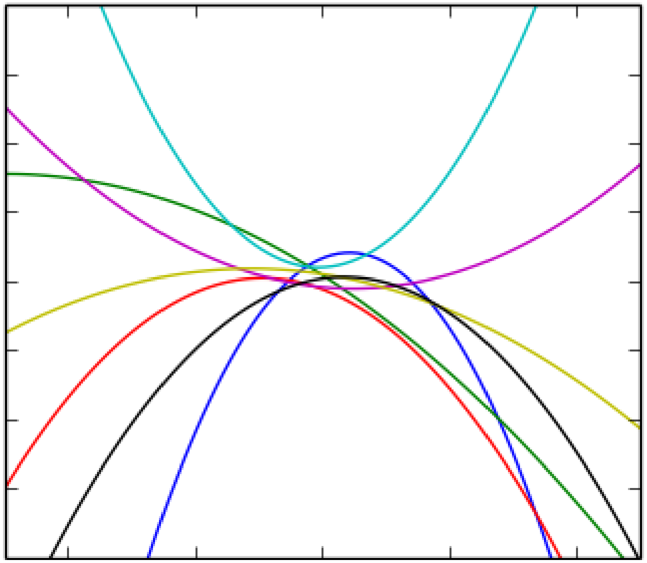
Posterior $P(h|S)$

$$K(x, x') = e^{-\sin^2(x-x')}$$





$$K(x, x') = (xx' + 1)^2$$

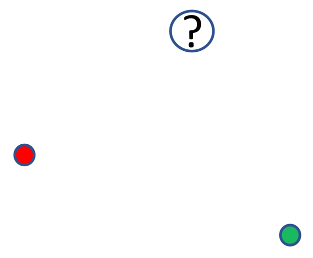


Beyond scope of course

Advantages of Probabilistic Approach

- **Probabilistic language** (conditional independence, hierarchical probabilistic models) often convenient and natural for expressing assumptions about reality
- Allows leveraging other assumptions, and integrating multiple data sources, in a principled way
  - E.g., knowledge about $P(x|w)$ can be useful if we have lots of *unlabeled* data (x_i without y_i)

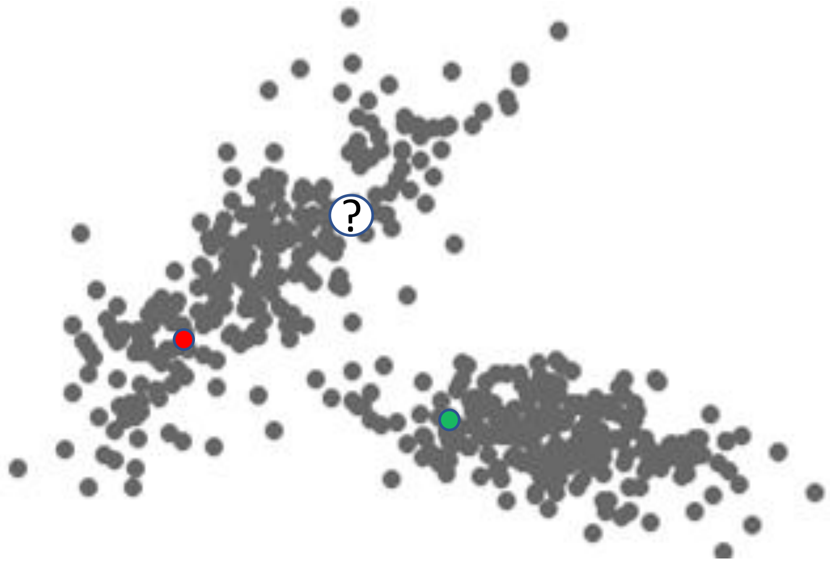
    (there are other ways to leverage unlabeled data, but prob approach directly implies recipe)



Advantages of Probabilistic Approach

- **Probabilistic language** (conditional independence, hierarchical probabilistic models) often convenient and natural for expressing assumptions about reality
- Allows leveraging other assumptions, and integrating multiple data sources, in a principled way
  - E.g., knowledge about $P(x|w)$ can be useful if we have lots of *unlabeled* data (x_i without y_i)

    (there are other ways to leverage unlabeled data, but prob approach directly implies recipe)



Advantages of Probabilistic Approach

- **Probabilistic language** (conditional independence, hierarchical probabilistic models) often convenient and natural for expressing assumptions about reality
- Allows leveraging other assumptions, and integrating multiple data sources, in a principled way
  - E.g., knowledge about $P(x|w)$ can be useful if we have lots of *unlabeled* data (x_i without y_i)

    (there are other ways to leverage unlabeled data, but prob approach directly implies recipe)
  - Language models: $\boldsymbol{P(text\,;\theta)} = \prod_{\boldsymbol{i}} \boldsymbol{p(\,text[i] \mid text[:i-1]\,;\theta)}$

    training: $\boldsymbol{\arg\max_{\theta} P(\{doc_1, doc_2, \ldots, doc_m\}\,;\theta)} \equiv \boldsymbol{\arg\min_{\theta} \sum_t \sum_i -\log p(doc_t[i] \mid doc_t[:i-1];\theta)}$
  - Leveraging other tasks, other observations of reality, …
- Yields meaningful ``certainties'' $P(y|x)$, not only $h(x)$, taking into account $P(y|x,w)$, but more importantly also uncertainty in estimating w (variance in the posterior $P(w|S)$)
- Actual right thing to do: integrate over posterior $p(w|S)$

- MAP approach:
  - $\hat{w} = \arg\max_w p(w|\{y_i\};\{x_i\})$ or $\hat{w} = \arg\max_w p(w|\{y_i\},\{x_i\})$ e.g. $\arg\min_w L^{lgst}(r_w) + \lambda\|w\|^2$
  - Build predictor $h(x_{query}) = h_{\hat{w}}(x_{query})$ based on $P(y_{query}|x_{query};\hat{w})$ e.g. $h_{\hat{w}}(w) = sign(r_{\hat{w}})$

- This depends not only on model class, but also choice of parametrization
  - Consider parametrizing using $u = e^w$ (ie $w = \log u$), with the same prior *distribution* over models, i.e. $P(u \in [a,b]) = P(w \in [\log a, \log b])$

    ➔ The *density:* $p_u(u) = \frac{dw}{du} p_w(w) = \frac{1}{u} p_w(\log u)$
  - The posterior *distribution* over models is the same, i.e. $P(u \in [a,b]|S) = P(w \in [\log a, \log b]|S)$ $S = \{x_i\},\{y_i\}$

    ➔ The posterior *density*: $p_{u|S}(u|S) = \frac{\mathbf{1}}{\boldsymbol{u}} p_{w|S}(w|S)$
  - Maximizing $p_{u|S}$ and $p_{w|S}$ is different!

- True Probabilistic (Bayesian) approach:
  - Use $\boldsymbol{P}(\boldsymbol{y_{query}}|\boldsymbol{S},\boldsymbol{x_{query}}) = \int_w P(y_{query}|x_{query},w)\, P(w|S) = \mathbb{E}_{w\sim P(w|S)}[P(y_{query}|x_{query},w)]$
  - E.g. $h(x_{query}) = \operatorname{argmin}_y \mathbb{E}_{y\sim \boldsymbol{P}(\boldsymbol{y_{query}}|\boldsymbol{S})}[loss(y, y_{query})] = [[\boldsymbol{P}(\boldsymbol{y_{query}}|\boldsymbol{S}) > 0.5]]$
- Much of the challenge of Bayesian Inference/Bayesian ML: calculating the integral

Beyond scope of course

Probabilistic **Bayesian** Approach

MAP: $\hat{w} = \arg\max_{w} p(w|\{y_i\};\{x_i\})$ or $\hat{w} = \arg\max_{w} p(w|\{y_i\},\{x_i\})$

then predict using $r_{\hat{w}}(x)$, pretending, e.g., $P(y|x) = P(y|x,\hat{w}) = \frac{1}{1+\exp(-r_{\hat{w}}(x))}$

Actual "right thing to do"™: integrate over posterior $p(w|S)$

Use $P(y_{tst}|x_{tst},\{y_i\},\{x_i\}) = \int P(y_{tst}|x_{tst},w)p(w|\{y_i\},\{x_i\})dw$

- Sometimes (significantly) improve prediction
- More "correct" since unlike MAP, doesn't depend on parametrization/base measure

  → can fix "broken" situations with MAP
- More importantly: more correct uncertainty estimate
- But: integral generally extremely intractable (though worth approximating?)
- And…… it's the "right thing to do" only if you truly believe $P(y,x|w), P(w)$

Beyond scope of course

Conditional likelihood
$-\log P(\{y_i\}|\{x_i\}; w)$

predictor class
E.g. $h_w = \langle w, \phi(x) \rangle$

"prior"
$R(w) = -\log P(w)$

$$\min_w \; L_S(h_w) \; + \; \lambda R(w)$$

$L_S(h_w) = \frac{1}{m} \sum \ell(h_w(x_i); y_i)$

$\ell(h_w(x_i); y_i) = -\log P(x|y)$

Captures, or is surrogate, for task-loss (e.g. ℓ^{01})

If $R(w) \propto scale$: Scale sensitive (e.g. Lipschitz)

For tractability: convex

$\ell(z; y)$ convex in z AND $h_w = \langle w, \phi(x) \rangle$ linear in w

➔ $\ell(h_w(x); y)$ and hence $L_S(h_w)$ convex in w

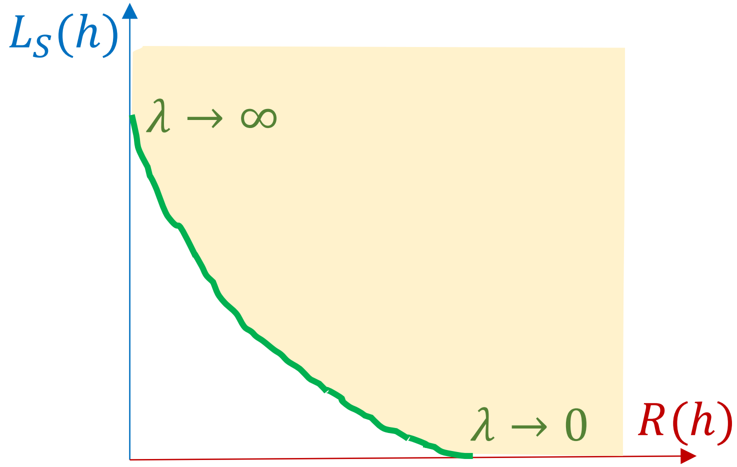
Inductive bias, simplicity among predictors $h_w \in \mathcal{H}$ (≈all functions?)

e.g. $R(w) = \|w\|_2$

Or other norms, e.g. $\|w\|_1$

Or non-norm, e.g. $\|w\|_0 = |\{i | w[i] \neq 0\}|$

For tractability: $R(w)$ convex in w



$h_w = \boldsymbol{w} \cdot \boldsymbol{x} + \boldsymbol{w}^2$

$$\min_{w} \quad L_S(h_w) \quad + \quad \lambda\, R(w)$$

$$L_S(h_w) = \frac{1}{m} \sum \ell(h_w(x_i); y_i)$$

$$\ell(h_w(x_i); y_i) = -\log P(x|y)$$

Captures, or is surrogate, for task-loss (e.g. ℓ^{01})

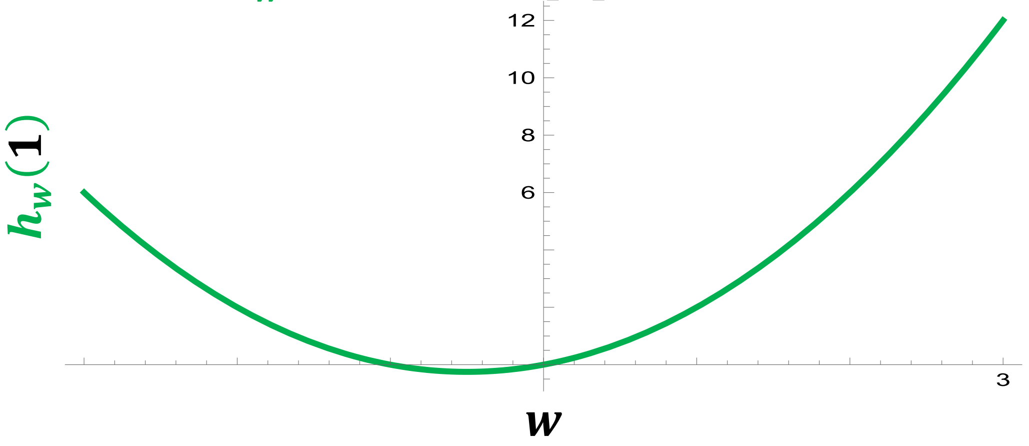
If $R(w) \propto scale$: Scale sensitive (e.g. Lipschitz)

For tractability: convex

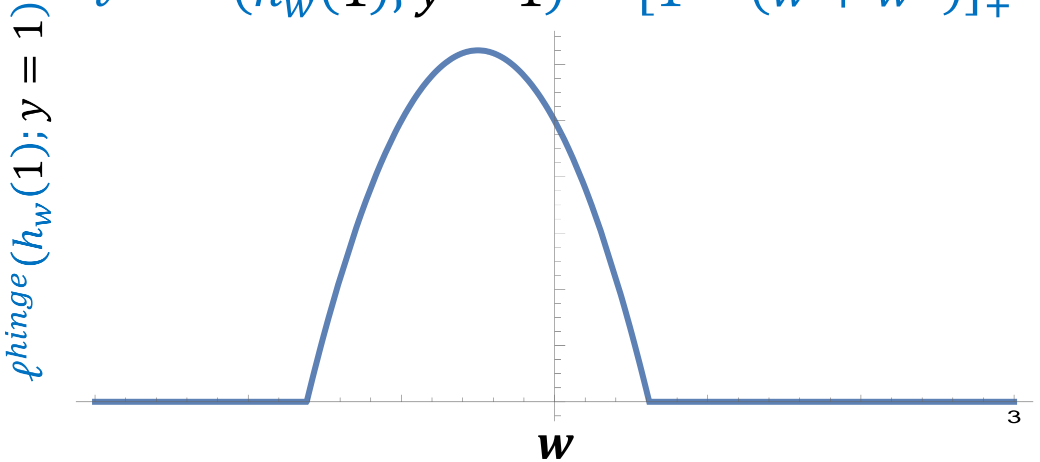
$\ell(z; y)$ convex in z

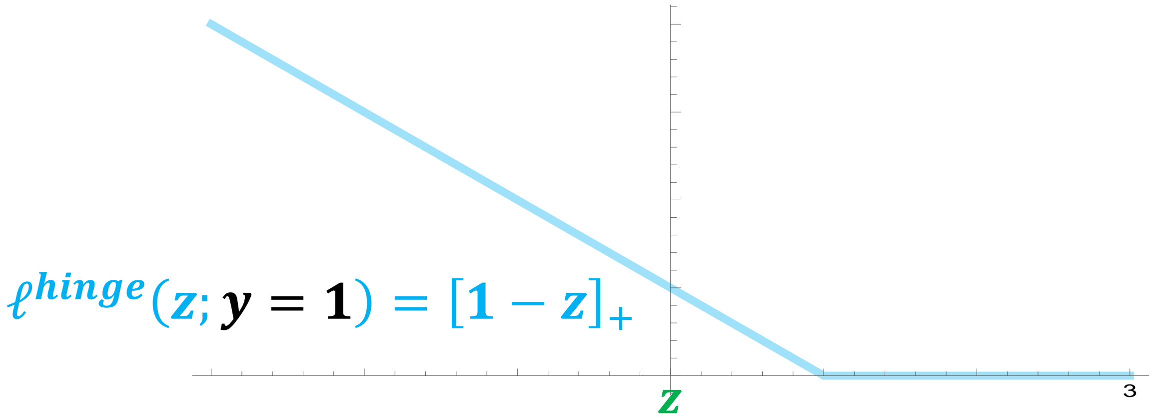
➔ $\ell(h_w(x); y)$ and hence $L_S(h_w)$ convex in w

$\boldsymbol{h_w(x) = w \cdot x[1] + w^2}$



$\ell^{hinge}(h_w(1); y = 1) = [1 - (w + w^2)]_+$





$$h_w = \langle w, \phi(x) \rangle$$

convex(affine) is convex

$\sum$convex is convex

$$\min_{w} \quad L_S(h_w) \quad + \quad \lambda\, R(w)$$

$$L_S(h_w) = \frac{1}{m} \sum \ell(h_w(x_i); y_i)$$

$$\ell(h_w(x_i); y_i) = -\log P(x|y)$$

Captures, or is surrogate, for task-loss (e.g. ℓ^{01})

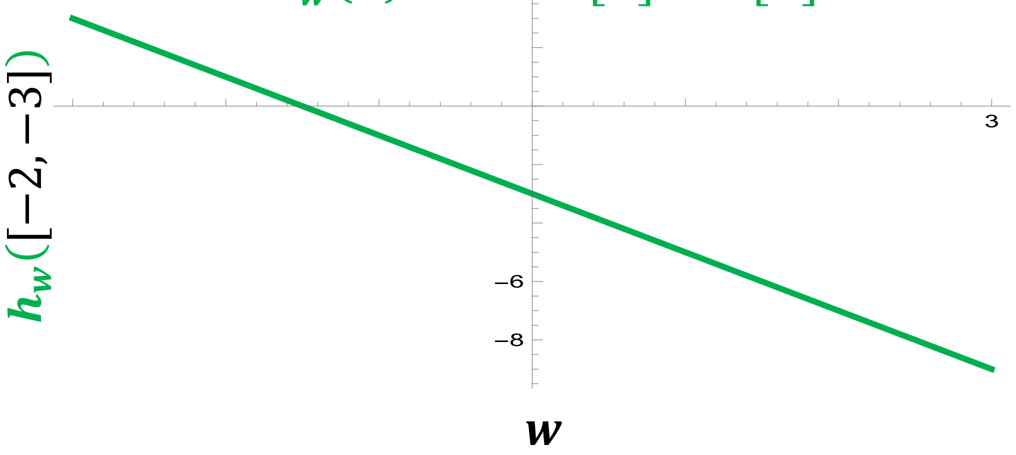
If $R(w) \propto scale$: Scale sensitive (e.g. Lipschitz)

For tractability: convex

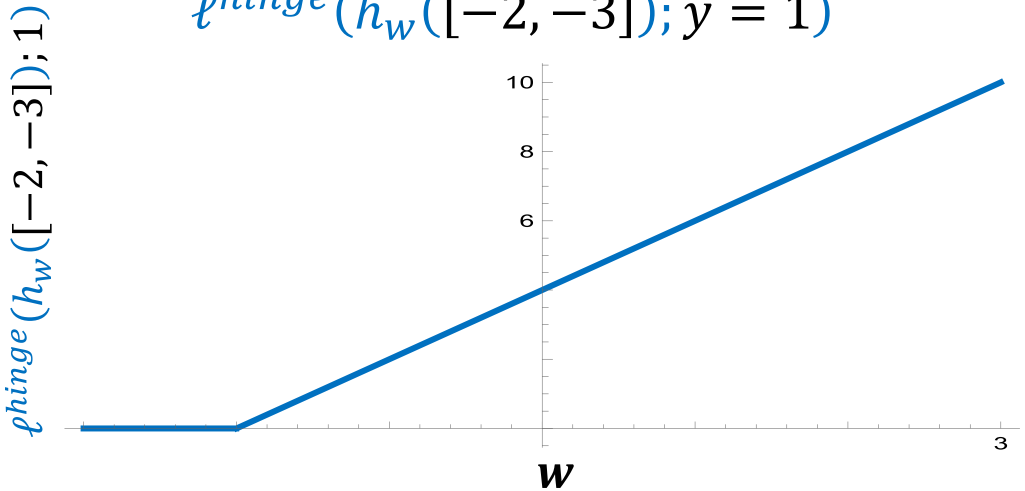
$\ell(z; y)$ convex in z AND $h_w = \langle w, \phi(x) \rangle$ linear in w

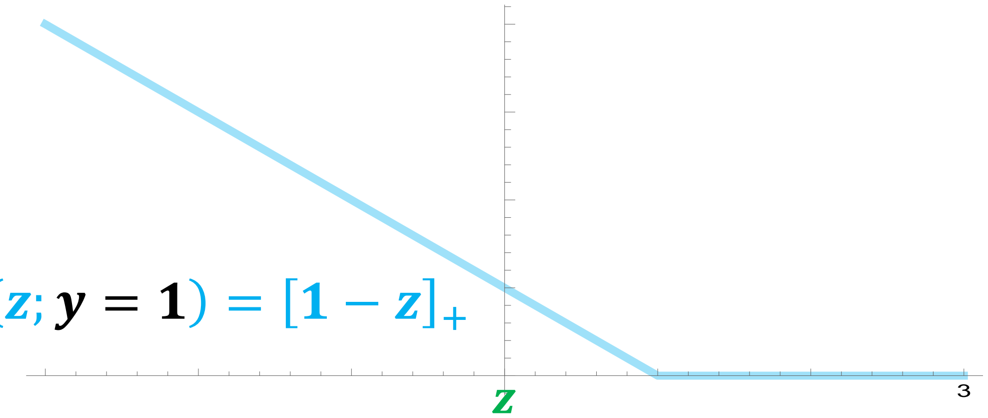
➔ $\ell(h_w(x); y)$ and hence $L_S(h_w)$ convex in w

$$\boldsymbol{h_w(x) = w \cdot x[1] + x[2]}$$



$$\ell^{hinge}(h_w([-2,-3]); y = 1)$$





$$\ell^{hinge}(\boldsymbol{z}; \boldsymbol{y} = \boldsymbol{1}) = [\boldsymbol{1} - \boldsymbol{z}]_+$$

Sub-Gradient Descent

$$\min F(w)$$

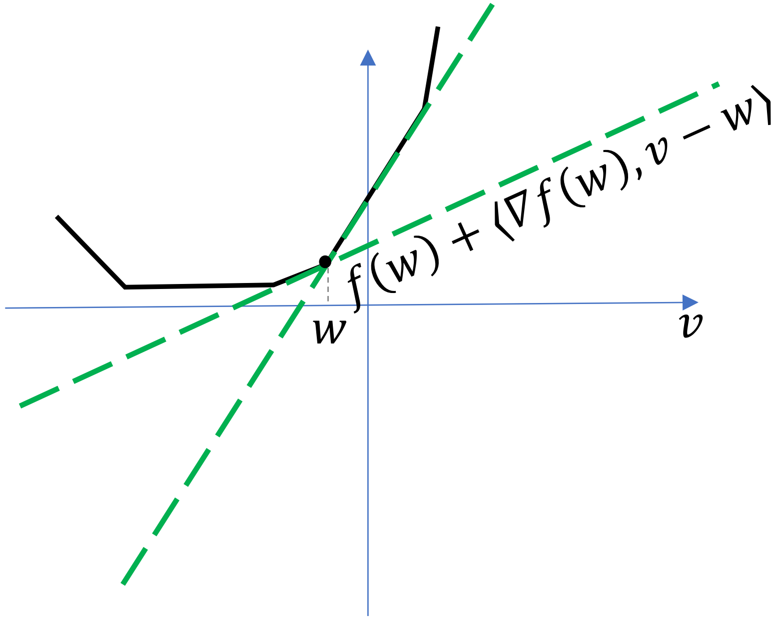
Initialize $w^{(0)} = 0$

At iteration t:

- $w^{(t+1)} \leftarrow w^{(t)} - \eta_t \nabla F\left(w^{(t)}\right)$

Sub-Gradients

- Definition: $\nabla f(w) \in \mathbb{R}^d$ is a **sub-gradient** of $f: \mathbb{R}^d \to \mathbb{R}$ at w if for all $v \in \mathbb{R}^d$:
$$f(v) \geq \underbrace{f(w) + \langle \nabla f(w), v - w \rangle}_{\text{linear lower bound passing through } f(w)}$$
- Claim: f is convex if and only if f has a sub-gradient at each point.
- Claim: If f is convex and differentiable, its only subgradient at each point is the gradient.
  - At non-diff points, there may be many sub-gradients. The set of all sub-gradients is the "differential set" $\partial f(w)$
- Sub-Gradient Descent: $w^{(t+1)} \leftarrow w^{(t)} - \eta_t \cdot \nabla f\left(w^{(t)}\right)$



(Sub)Gradient Descent

$$\min F(w) \quad \text{s.t. } w \in \mathcal{W}$$

Initialize $w^{(0)} = 0$
At iteration t:
- Obtain $g^{(t)} = \nabla F\left(w^{(t)}\right) \in \partial \mathrm{F}\left(\mathrm{w}^{(\mathrm{t})}\right)$
- $w^{(t+1)} \leftarrow \Pi_{\mathcal{W}}\left(w^{(t)} - \eta_t g^{(t)} \right)$

Return $w^{(T)}$

$\Pi_{\mathcal{W}}(w) = \arg\min_{v \in \mathcal{W}} \|v - w\|_2$
e.g., $\Pi_{\|w\|_2 \leq B}(w) = B \frac{w}{\|w\|_2} \; if \; \|w\|_2 \geq B$

Gradients for (R)ERM

$$F(w) = \frac{1}{m} \sum_i loss(\langle w, \phi(x_i) \rangle, y_i) + \lambda R(w)$$

$$\nabla F(w) = \frac{1}{m} \sum_i loss'(\langle w, \phi(x_i) \rangle, y_i) \phi(x_i) + \lambda \nabla R(w)$$

(Sub)Gradient Descent

$$\min F(w) \quad \text{s.t. } w \in \mathcal{W}$$

Initialize $w^{(0)} = 0$
At iteration t:
- Obtain $g^{(t)} = \nabla F\left(w^{(t)}\right) \in \partial \mathrm{F}\left(\mathrm{w}^{(t)}\right)$
- $w^{(t+1)} \leftarrow \Pi_{\mathcal{W}}\left(w^{(t)} - \eta_t g^{(t)} \right)$

Return $w^{(T)}$

$$\Pi_{\mathcal{W}}(w) = \arg\min_{v \in \mathcal{W}} \|v - w\|_2$$

e.g., $\Pi_{\|w\|_2 \le B}(w) = B \frac{w}{\|w\|_2} \; if \; \|w\|_2 \ge B$

Analysis: If F is convex and **ρ-Lipschitz (i.e. $\|\nabla F(w)\|_2 \le \rho$)** then

using appropriate step-size (line search or $\eta_t = \sqrt{\frac{B^2}{\rho^2 T}}$ or $\eta_t = \sqrt{\frac{B^2}{\rho^2 t}}$):

$$F\left(w^{(T)}\right) \le \inf_{w \in \mathcal{W}, \|w\|_2 \le B} F(w) + \sqrt{\frac{B^2 \rho^2}{T}}$$

i.e., to ensure $F(w) \le F(w^*) + \epsilon$, need:

$$T \ge \frac{B^2 \rho^2}{\epsilon^2}$$

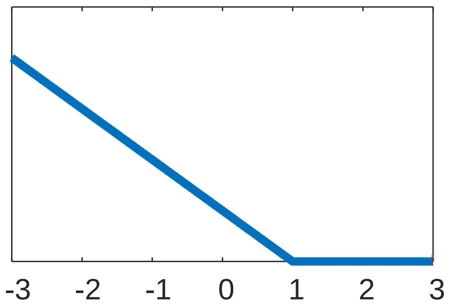
Analysis of Projected Sub-Gradient Descent for SVM

$$\min \frac{1}{m}\sum_i hinge(y_i\langle w, \phi(x_i)\rangle) \quad \text{s.t.} \quad \|w\|_2 \le B$$

$$\|\nabla F(w)\|_2 =$$

$$\le \frac{1}{m}\sum_i \underset{\le 1}{|hinge'(\cdot)|} \cdot \underset{= 1}{|y_i|} \cdot \|\phi(x_i)\|_2$$

Recall we need $\|\phi(x)\| \le R$



- Conclusion: to ensure $L_S\left(w^{(T)}\right) - L_S(\hat{w}) \le \epsilon_{opt}$:

$$T = \frac{R^2 B^2}{\epsilon_{opt}^2} \text{ iterations}$$

- Runtime for each iteration:

$O(m)$ vector operations

(Sub)Gradient Descent

$$\min F(w) \quad \text{s.t. } w \in \mathcal{W}$$

Initialize $w^{(0)} = 0$

At iteration t:

- Obtain $g^{(t)} = \nabla F\left(w^{(t)}\right) \in \partial \mathrm{F}\left(\mathrm{w}^{(t)}\right)$
- $w^{(t+1)} \leftarrow \Pi_{\mathcal{W}}\left(w^{(t)} - \eta_t g^{(t)} \right)$

Return $w^{(T)}$

$$\Pi_{\mathcal{W}}(w) = \arg\min_{v \in \mathcal{W}} \|v - w\|_2$$

e.g., $\Pi_{\|w\|_2 \leq B}(w) = B \frac{w}{\|w\|_2}$ if $\|w\|_2 \geq B$

Analysis: If F is convex and **ρ-Lipschitz** (**i.e.** $\|\nabla F(w)\|_2 \leq \rho$) then

using appropriate step-size (line search or $\eta_t = \sqrt{\frac{B^2}{\rho^2 T}}$ or $\eta_t = \sqrt{\frac{B^2}{\rho^2 t}}$):

$$F\left(w^{(T)}\right) \leq \inf_{w \in \mathcal{W}, \|w\|_2 \leq B} F(w) + \sqrt{\frac{B^2 \rho^2}{T}}$$

i.e., to ensure $F(w) \leq F(w^*) + \epsilon$, need:

$$T = \frac{B^2 \rho^2}{\epsilon^2} = \frac{B^2 R^2}{\epsilon^2} \text{ iterations}$$

$$F(w) = \frac{1}{m} \sum_i hinge(y_i \langle w, \phi(x_i) \rangle)$$

$$\|\phi(c_i)\| \leq R$$

To calculate $g^{(t)} = \nabla F\left(w^{(t)}\right) \in \partial F\left(w^{(t)}\right)$:

$O(m)$ vector operations $= O(md)$ runtime per iteration

Stochastic (Sub)Gradient Descent (SGD)

$$\min F(w) \quad \text{s.t. } w \in \mathcal{W}$$

Initialize $w^{(0)} = 0$
At iteration t:
- Obtain $g^{(t)}$ s.t. $\mathbb{E}[g^{(t)}] \in \partial F(w^{(t)})$
- $w^{(t+1)} \leftarrow \Pi_{\mathcal{W}}(\, w^{(t)} - \eta_t g^{(t)} \,)$

Return $\bar{w}^{(T)} = \frac{1}{T}\sum_{t=1}^{T} w^{(t)}$

$g^{(1)}, g^{(2)}, \dots$ independent
Formally:
$\mathbb{E}[g^{(t)} | g^{(1)} \dots g^{(t-1)}] \in \partial F(w^{(t)})$

$\Pi_{\mathcal{W}}(w) = \arg\min_{v \in \mathcal{W}} \|v - w\|_2$
e.g., $\Pi_{\|w\|_2 \le B}(w) = B \frac{w}{\|w\|_2}$ $if\ \|w\|_2 \ge B$

Analysis: If F is convex and **ρ-Lipschitz (i.e. $\|\nabla F(w)\|_2 \le \rho$)** then using appropriate step-size (line search or $\eta_t = \sqrt{\frac{B^2}{\rho^2 T}}$ or $\eta_t = \sqrt{\frac{B^2}{\rho^2 t}}$):

$$F(w^{(T)}) \le \inf_{w \in \mathcal{W}, \|w\|_2 \le B} F(w) + \sqrt{\frac{B^2 \rho^2}{T}}$$

i.e., to ensure $F(w) \le F(w^*) + \epsilon$, need:

$$T = \frac{B^2 \rho^2}{\epsilon^2} = \frac{B^2 R^2}{\epsilon^2} \text{ iterations}$$

Only need **independent** unbiased estimates $g^{(t)}$ of (sub)gradient, i.e. s.t. $\mathbb{E}[g^{(t)}] = \nabla F(w^{(t)}) \in \partial F(w^{(t)})$

Obtaining Independent Sub-Gradients Estimates

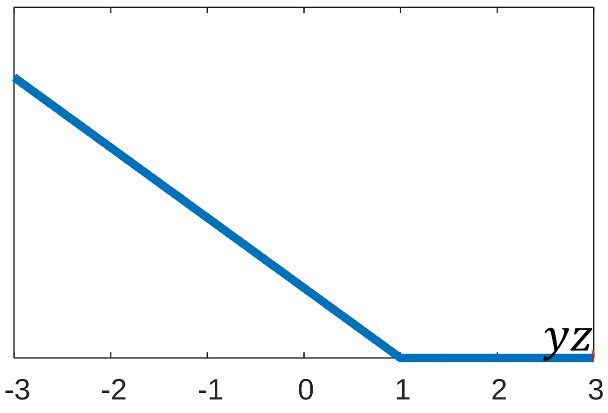
$$F(w) = \frac{1}{m}\sum_i loss(\langle w, \phi(x_i)\rangle, y_i)$$

$$\nabla F(w) = \frac{1}{m}\sum_i loss'(\langle w, \phi(x_i)\rangle, y_i)\ \phi(x_i)$$

Pick $i \sim Unif(1..m)$ at random and use: $g = loss'(\langle w, \phi(x_i)\rangle, y_i)\ \phi(x_i)$

➔ $\mathbb{E}[g] = \nabla F(w)$

For $\ell^{hinge}(z,y) = [1 - yz]_+$, $loss'(\langle w, \phi(x_i)\rangle, y_i) = \begin{cases} -y_i, & y_i\langle w, \phi(x_i)\rangle < 1 \\ 0, & y_i\langle w, \phi(x_i)\rangle > 1 \end{cases}$



➔ $g = loss'(\langle w, \phi(x_i)\rangle, y_i)\ \phi(x_i) = \begin{cases} -y_i\phi(x_i) & , y_i\langle w, \phi(x_i)\rangle < 1 \\ 0 & , y_i\langle w, \phi(x_i)\rangle > 1 \end{cases}$

Obtaining Independent Sub-Gradients Estimates

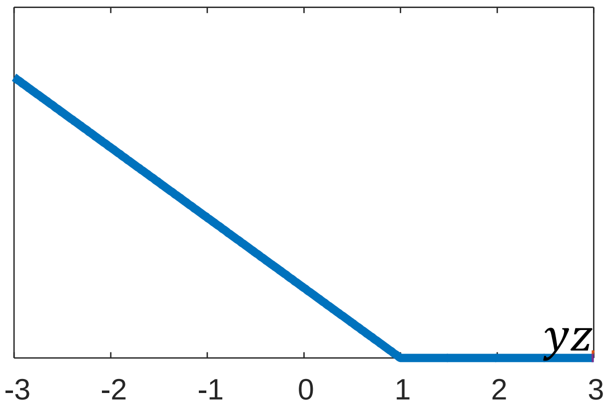
$$F(w) = \frac{1}{m}\sum_i loss(\langle w, \phi(x_i)\rangle, y_i) + \lambda R(w)$$

$$\nabla F(w) = \frac{1}{m}\sum_i loss'(\langle w, \phi(x_i)\rangle, y_i)\ \phi(x_i)\ + \lambda \nabla R(w)$$

Pick $i \sim Unif(1..m)$ at random and use: $g = loss'(\langle w, \phi(x_i)\rangle, y_i)\ \phi(x_i)\ + \lambda \nabla R(w)$

➔ $\mathbb{E}[g] = \nabla F(w)$

For $\ell^{hinge}(z, y) = [1 - yz]_+,\ loss'(\langle w, \phi(x_i)\rangle, y_i) = \begin{cases} -y_i, & y_i\langle w, \phi(x_i)\rangle < 1 \\ 0, & y_i\langle w, \phi(x_i)\rangle > 1 \end{cases}$



For $R(w) = \frac{1}{2}\|w\|_2^2,\ \nabla R(w) = w$

➔ $g = loss'(\langle w, \phi(x_i)\rangle, y_i)\ \phi(x_i)\ + \lambda w = \begin{cases} -y_i\phi(x_i) + \lambda w & ,\ y_i\langle w, \phi(x_i)\rangle < 1 \\ \lambda w & ,\ y_i\langle w, \phi(x_i)\rangle > 1 \end{cases}$

SGD for SVM

$\{g^{(t)}\}$ independent
And $\mathbb{E}\left[g^{(t)}\right] = \nabla L_S(w^{(t)})$

$$\min L_S(w) \;\; s.t. \;\; \|w\|_2 \leq B$$

Use $g^{(t)} = \nabla_w hinge(y_i\langle w, \phi_i(x)\rangle) = \begin{cases} -y_i\phi(x_i), & y_i\phi(x_i) < 1 \\ 0, & y_i\phi(x_i) > 1 \end{cases}$ for random i

Initialize $w^{(0)} = 0$
At iteration t:

- Obtain $g^{(t)}$ s.t. $\mathbb{E}\left[g^{(t)}\right] \in \partial F\left(w^{(t)}\right)$
- $w^{(t+1)} \leftarrow \Pi_{\mathcal{W}}(\, w^{(t)} - \eta_t g^{(t)} \,)$

Return $\overline{w}^{(T)} = \frac{1}{T}\sum_{t=1}^{T} w^{(t)}$

Initialize $w^{(0)} = 0$
At iteration t:

- Pick $i \in 1 \ldots m$ at random
- If $y_i\langle w^{(t)}, \phi(x_i)\rangle < 1$,
  $w^{(t+1)} \leftarrow w^{(t)} + \eta_t y_i \phi(x_i)$
  else: $w^{(t+1)} \leftarrow w^{(t)}$
- If $\left\|w^{(t+1)}\right\|_2 > B$, then $w^{(t+1)} \leftarrow B\frac{w^{(t+1)}}{\left\|w^{(t+1)}\right\|_2}$

Return $\overline{w}^{(T)} = \frac{1}{T}\sum_{t=1}^{T} w^{(t)}$

$g^{(t)} = \nabla_w\left[1 - y_i w^{(t)}\langle w, \phi_i(x)\rangle\right]_+$

$w^{(t+1)} \leftarrow \boldsymbol{\Pi}_{\|w\|\leq B}\left(\, w^{(t)} - \eta_t g^{(t)} \,\right)$

$$\left\|g^{(t)}\right\|_2 \leq R \Rightarrow L_S\left(\overline{w}^{(T)}\right) \leq L_S(\hat{w}) + \sqrt{\frac{B^2R^2}{T}}$$

(in expectation over randomness in algorithm)

SGD for SVM

$\{g^{(t)}\}$ independent
And $\mathbb{E}[g^{(t)}] = \nabla\left(L_S(w^{(t)}) + \frac{\lambda}{2}\|w\|_2^2\right)$

$$\min L_S(w) + \frac{\lambda}{2}\|w\|_2^2$$

Use $g^{(t)} = \nabla_w hinge(y_i\langle w, \phi_i(x)\rangle) = \begin{cases} -y_i\phi(x_i) + \lambda w, & y_i\phi(x_i) < 1 \\ \lambda w, & y_i\phi(x_i) > 1 \end{cases}$ for random i

Initialize $w^{(0)} = 0$
At iteration t:

- Obtain $g^{(t)}$ s.t. $\mathbb{E}[g^{(t)}] \in \partial F(w^{(t)})$
- $w^{(t+1)} \leftarrow \Pi_{\mathcal{W}}(w^{(t)} - \eta_t g^{(t)})$

Return $\bar{w}^{(T)} = \frac{1}{T}\sum_{t=1}^{T} w^{(t)}$

Initialize $w^{(0)} = 0$
At iteration t:

- Pick $i \in 1 \ldots m$ at random
- If $y_i\langle w^{(t)}, \phi(x_i)\rangle < 1$,
  $w^{(t+1)} \leftarrow w^{(t)} + \eta_t y_i \phi(x_i)$
  else: $w^{(t+1)} \leftarrow w^{(t)}$
- $w^{(t+1)} \leftarrow w^{(t+1)} - \lambda\eta_t w^{(t)}$

Return $\bar{w}^{(T)} = \frac{1}{T}\sum_{t=1}^{T} w^{(t)}$

$g^{(t)} = \nabla_w[1 - y_i w^{(t)}\phi(x_i)]_+ + \lambda w$

$w^{(t+1)} \leftarrow \quad w^{(t)} - \eta_t g^{(t)}$

Stochastic vs Batch

- Batch Gradient descent:

  $O\left(\frac{B^2R^2}{\epsilon^2}\right)$ iterations, $\boldsymbol{O(m)}$ **vector operations per iteration**, $O(d)$ per vector operation

  ➔ Runtime: $O\left(\frac{B^2R^2}{\epsilon^2}d \cdot m\right)$

- Stochastic Gradient Descent:

  $O\left(\frac{B^2R^2}{\epsilon^2}\right)$ iterations, $\boldsymbol{O(1)}$ **vector operations per iteration**, $O(d)$ per vector operation

  ➔ Runtime: $O\left(\frac{B^2R^2}{\epsilon^2}d\right)$

Intuitive argument: if only taking simple gradient steps, might as well be stochastic

Stochastic vs Batch

$$\min L_S(w) = \frac{1}{m}\sum_i loss\big(w \text{ on } (x_1, y_1)\big)$$

$w \leftarrow w - \eta \sum g_i$	$w \leftarrow w - \eta g_1$	$g_1 = \nabla loss\big(w \text{ on } (x_1, y_1)\big)$	$\mathbf{x}_1, y_1$	$g_1 = \nabla loss\big(w \text{ on } (x_1, y_1)\big)$
	$w \leftarrow w - \eta g_2$	$g_2 = \nabla loss\big(w \text{ on } (x_2, y_2)\big)$	$\mathbf{x}_2, y_2$	$g_2 = \nabla loss\big(w \text{ on } (x_2, y_2)\big)$
	$w \leftarrow w - \eta g_3$	$g_3 = \nabla loss\big(w \text{ on } (x_3, y_3)\big)$	$\mathbf{x}_3, y_3$	$g_3 = \nabla loss\big(w \text{ on } (x_3, y_3)\big)$
	$w \leftarrow w - \eta g_4$	$g_4 = \nabla loss\big(w \text{ on } (x_4, y_4)\big)$	$\mathbf{x}_4, y_4$	$g_4 = \nabla loss\big(w \text{ on } (x_4, y_4)\big)$
	$w \leftarrow w - \eta g_5$	$g_5 = \nabla loss\big(w \text{ on } (x_5, y_5)\big)$	$\mathbf{x}_5, y_5$	$g_5 = \nabla loss\big(w \text{ on } (x_5, y_5)\big)$
	$\vdots$		$\vdots$	
	$w \leftarrow w - \eta g_{m-1}$			
	$w \leftarrow w - \eta g_m$	$g_m = \nabla loss\big(w \text{ on } (x_m, y_m)\big)$	$\mathbf{x}_m, y_m$	$g_m = \nabla loss\big(w \text{ on } (x_m, y_m)\big)$

$$\nabla L_S(w) = \frac{1}{m}\sum g_i$$

$$w \leftarrow w - \eta \tfrac{1}{m}\sum g_i$$

Stochastic vs Batch

$$\min L_S(w) = \frac{1}{m}\sum_i loss\big(w \text{ on } (x_1, y_1)\big)$$

$$w \leftarrow w - \eta \sum g_i \left\{ \begin{array}{l} w \leftarrow w - \eta g_1 \\ w \leftarrow w - \eta g_2 \\ w \leftarrow w - \eta g_3 \end{array} \right.$$

$$w \leftarrow w - \eta \sum g_i \left\{ \begin{array}{l} w \leftarrow w - \eta g_4 \\ w \leftarrow w - \eta g_5 \end{array} \right.$$

$$\vdots \qquad \vdots$$

$$w \leftarrow w - \eta \sum g_i \left\{ \begin{array}{l} w \leftarrow w - \eta g_{m-1} \\ w \leftarrow w - \eta g_m \end{array} \right.$$

$g_1 = \nabla loss\big(w \text{ on } (x_1, y_1)\big)$	$\mathbf{x}_1, y_1$	$g_1 = \nabla loss\big(w \text{ on } (x_1, y_1)\big)$
$g_2 = \nabla loss\big(w \text{ on } (x_2, y_2)\big)$	$\mathbf{x}_2, y_2$	$g_2 = \nabla loss\big(w \text{ on } (x_2, y_2)\big)$
$g_3 = \nabla loss\big(w \text{ on } (x_3, y_3)\big)$	$\mathbf{x}_3, y_3$	$g_3 = \nabla loss\big(w \text{ on } (x_3, y_3)\big)$
$g_4 = \nabla loss\big(w \text{ on } (x_4, y_4)\big)$	$\mathbf{x}_4, y_4$	$g_4 = \nabla loss\big(w \text{ on } (x_4, y_4)\big)$
$g_5 = \nabla loss\big(w \text{ on } (x_5, y_5)\big)$	$\mathbf{x}_5, y_5$	$g_5 = \nabla loss\big(w \text{ on } (x_5, y_5)\big)$
	$\vdots$	
$g_m = \nabla loss\big(w \text{ on } (x_m, y_m)\big)$	$\mathbf{x}_m, y_m$	$g_m = \nabla loss\big(w \text{ on } (x_m, y_m)\big)$

$$\nabla L_S(w) = \frac{1}{m}\sum g_i$$

$$w \leftarrow w - \eta \tfrac{1}{m} \sum g_i$$

Mini-Batch SGD: use $g_t = \frac{1}{\boldsymbol{b}} \sum_{j \in \mathcal{B}_t} \nabla \ell\big(h_{w^{(t)}}(x_j), y_j\big)$

$\mathcal{B}_t$ =random subset of $\boldsymbol{b}$ training indices

$$F(w) = \frac{1}{m}\sum_i loss(\langle w, \phi(x_i)\rangle, y_i) \qquad \nabla F(w) = \frac{1}{m}\sum_i loss'(\langle w, \phi(x_i)\rangle, y_i)\ \phi(x_i)$$

At each iteration, independent (sub)gradient estimate g_t s.t. $\mathbb{E}[g_t] = \nabla F\left(w^{(t)}\right)$

$$g_t = \frac{1}{\boldsymbol{b}}\sum_{i \in \boldsymbol{\mathcal{B}_t}} loss'\left(\left\langle w^{(t)}, \phi(x_i)\right\rangle, y_i\right)\ \phi(x_i)$$

$\boldsymbol{\mathcal{B}_t}$ =random subset of $\boldsymbol{b}$ training indices

Initialize $w^{(0)} = 0$
At iteration t:
- Obtain $g^{(t)}$ s.t. $\mathbb{E}\left[g^{(t)}\right] \in \partial \mathrm{F}\left(\mathrm{w}^{(t)}\right)$
- $w^{(t+1)} \leftarrow \Pi_{\mathcal{W}}(\ w^{(t)} - \eta_t g^{(t)}\ )$

Initialize $w^{(0)} = 0$
At iteration t:
- Pick $\mathcal{B}_t \subseteq [1..m]$, $|\mathcal{B}_t| = \boldsymbol{b}$ at random
- $w^{(t+1)} = w^{(t)}$
- For $i \in \mathcal{B}_t$

  $$w^{(t+1)} \mathrel{+}= \frac{\eta_t}{b}\ loss'\left(\left\langle w^{(t)}, \phi(x_i)\right\rangle, y_i\right)\ \phi(x_i)$$
- If $\left\|w^{(t+1)}\right\|_2 > B$, then $w^{(t+1)} \leftarrow B\frac{w^{(t+1)}}{\left\|w^{(t+1)}\right\|_2}$

$O\left(\frac{B^2R^2}{\epsilon^2}\right)$ iterations, **$\boldsymbol{O}(\boldsymbol{b})$ vector operations per iteration** ➔ $O\left(\frac{B^2R^2}{\epsilon^2}\ \boldsymbol{b}\ d\right)$

$$w \leftarrow w - \eta \textstyle\sum g_i \left\{ \begin{array}{l} w \leftarrow w - \eta g_1 \\ w \leftarrow w - \eta g_2 \\ w \leftarrow w - \eta g_3 \end{array} \right.$$

$$w \leftarrow w - \eta \textstyle\sum g_i \left\{ \begin{array}{l} w \leftarrow w - \eta g_4 \\ w \leftarrow w - \eta g_5 \end{array} \right.$$

$\vdots \qquad \vdots$

$$w \leftarrow w - \eta \textstyle\sum g_i \left\{ \begin{array}{l} w \leftarrow w - \eta g_{m-1} \\ w \leftarrow w - \eta g_m \end{array} \right.$$

$g_1 = \nabla loss\big(w \text{ on } (x_1, y_1)\big)$	$\mathbf{x}_1, y_1$
$g_2 = \nabla loss\big(w \text{ on } (x_2, y_2)\big)$	$\mathbf{x}_2, y_2$
$g_3 = \nabla loss\big(w \text{ on } (x_3, y_3)\big)$	$\mathbf{x}_3, y_3$
$g_4 = \nabla loss\big(w \text{ on } (x_4, y_4)\big)$	$\mathbf{x}_4, y_4$
$g_5 = \nabla loss\big(w \text{ on } (x_5, y_5)\big)$	$\mathbf{x}_5, y_5$
	$\vdots$
$g_m = \nabla loss\big(w \text{ on } (x_m, y_m)\big)$	$\mathbf{x}_m, y_m$

Initialize $w^{(0)} = 0$

At iteration t:

- Obtain $g^{(t)}$ s.t. $\mathbb{E}\left[g^{(t)}\right] \in \partial \mathrm{F}\left(\mathrm{w}^{(\mathrm{t})}\right)$
- $w^{(t+1)} \leftarrow \Pi_{\mathcal{W}}\left(w^{(t)} - \eta_t g^{(t)} \right)$

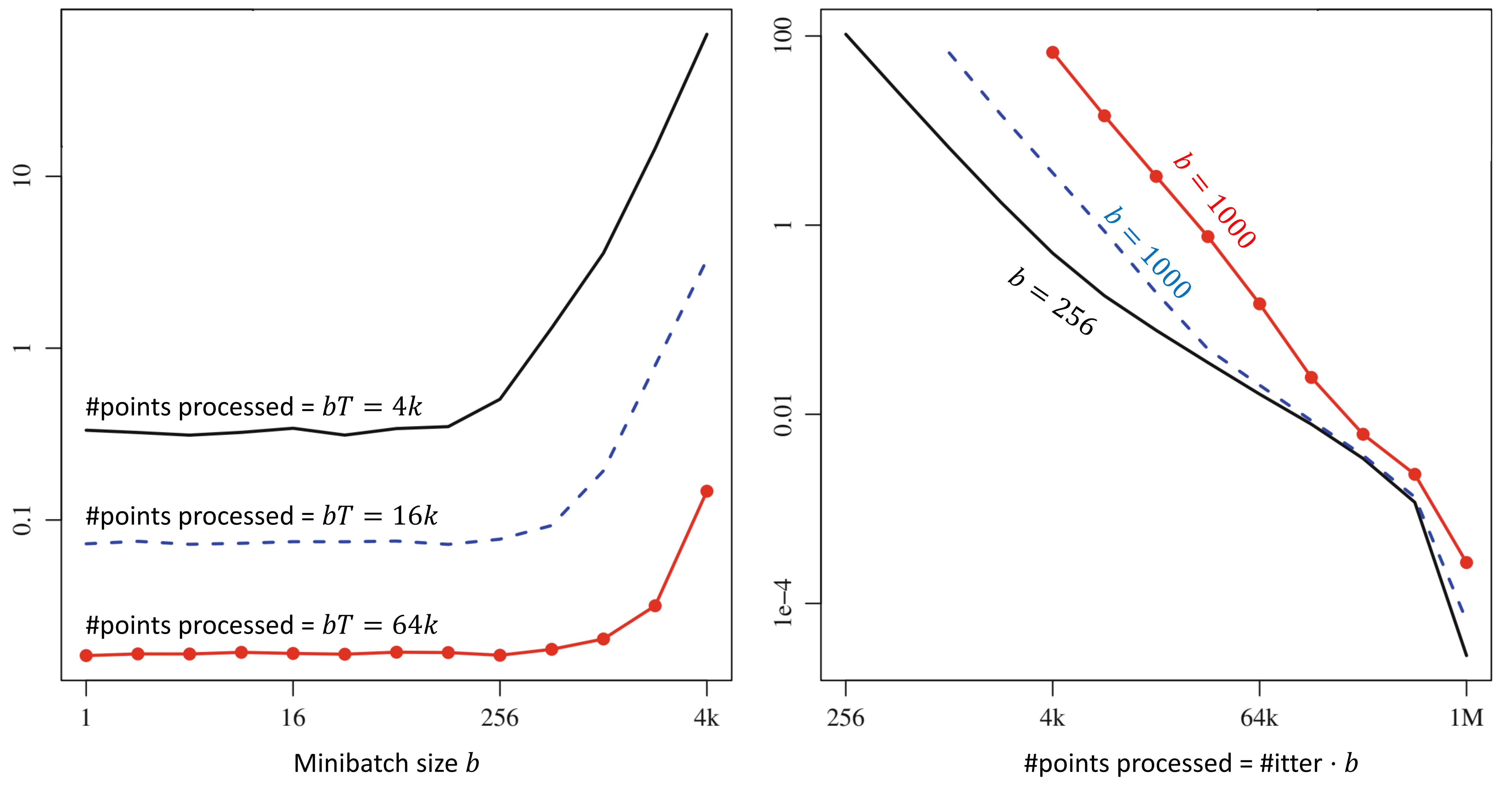
Initialize $w^{(0)} = 0$

At iteration t:

- Pick $\mathcal{B}_t \subseteq [1..m]$, $|\mathcal{B}_t| = \boldsymbol{b}$ at random
- $w^{(t+1)} = w^{(t)}$
- For $i \in \mathcal{B}_t$

  $$w^{(t+1)} \mathrel{+}= \frac{\eta_t}{b}\, loss'\left(\left\langle w^{(t)}, \phi(x_i) \right\rangle, y_i\right) \phi(x_i)$$
- If $\left\| w^{(t+1)} \right\|_2 > B$, then $w^{(t+1)} \leftarrow B \dfrac{w^{(t+1)}}{\left\| w^{(t+1)} \right\|_2}$

$O\left(\frac{B^2 R^2}{\epsilon^2}\right)$ iterations, **$\boldsymbol{O}(\boldsymbol{b})$ vector operations per iteration ➔** $O\left(\frac{B^2 R^2}{\epsilon^2}\, \boldsymbol{b}\, d\right)$



#points processed = $bT = 4k$
#points processed = $bT = 16k$
#points processed = $bT = 64k$
10
1
0.1
1
16
256
4k
Minibatch size b
100
1
0.01
1e−4
$b = 1000$
$b = 1000$
$b = 256$
256
4k
64k
1M
#points processed = #itter · b

- Stochastic Gradient Descent Runtime: $O\left(\frac{B^2R^2}{\epsilon_{opt}^2} d\right)$
- Batch Gradient Descent Runtime: $O\left(\frac{B^2R^2}{\epsilon_{opt}^2} d \cdot m\right)$
- Mini-Batch Gradient Descent Runtime: $O\left(\frac{B^2R^2}{\epsilon_{opt}^2} b \cdot m\right)$

  Optimal b (in theory and in practice if counting #vector ops): $b = 1$

  In practice, moderate b just as good as $b = 1$, reduces overhead, allows parallalization

- GD actually better for smooth objectives, and under other assumptions
- Also alternate methods, eg Newton, approx. Newton (including BFGS) for smooth objective, Interior Point methods for handling non-smoothness/constraints
- Goal of much of optimization: $\log 1/\epsilon$ dependence (and many of the above achieve this)

- How small should ϵ_{opt} be?
- What about $L_{\mathcal{D}}(w)$, which is what we really care about?

Overall Analysis of $L_{\mathcal{D}}(w)$

- Recall for ERM: $\underbrace{L_{\mathcal{D}}(\widehat{w})}_{\widehat{w} = \arg\min_{\|w\|\le B} L_S(w)} \le \underbrace{L_{\mathcal{D}}(w^*)}_{w^* = \arg\min_{\|w\|\le B} L_{\mathcal{D}}(w)} + 2\sup_{w} |L_{\mathcal{D}}(w) - L_S(w)|$

- For ϵ_{opt} suboptimal ERM $\overline{w}$:

$$L_{\mathcal{D}}(\overline{w}) \le \underbrace{L_{\mathcal{D}}(w^*)}_{\epsilon_{aprox}} + \underbrace{2\sup_{w} |L_{\mathcal{D}}(w) - L_S(w)|}_{\epsilon_{est} \le 2\sqrt{\frac{B^2R^2}{m}}} + \underbrace{\left(L_S(\overline{w}) - L_S(\widehat{w})\right)}_{\epsilon_{opt} \le \sqrt{\frac{B^2R^2}{T}}}$$

- Take $\epsilon_{opt} \approx \epsilon_{est}$, i.e. $\#iter\ T \approx sample\ size\ m$

- To ensure $L_{\mathcal{D}}(w) \le L_{\mathcal{D}}(w^*) + \epsilon$:

$$T, m = O\left(\frac{B^2R^2}{\epsilon^2}\right)$$

SGD as an Ultimate Optimization Algorithm

- Runtime of SGD is $O(T \cdot d) = O\left(\frac{B^2R^2}{\epsilon^2}d\right) = O(m \cdot d)$

- Linear in the number of required examples m
  - vs full GD which requires quadratic time $O(Tmd) = O(m^2d)$
  - or interior point or matrix inversion methods, which require cubic or worse runtime

- SGD runtime is linear in time it takes to read data set

  ➔Can’t be improved beyond small constant factor **without additional assumptions**

SGD as a Learning Algorithm: SGD on $L_{\mathcal{D}}(w)$

$$\min_{w} L_{\mathcal{D}}(w)$$

use $g^{(t)} = \nabla_w loss\left(h_{w^{(t)}}(x); y\right)$ for random $y, x \sim \mathcal{D}$ ➔ $\mathbb{E}\left[g^{(t)}\right] = \nabla L_{\mathcal{D}}(w)$

Initialize $w^{(0)} = 0$

At iteration t:

- Draw $x_t, y_t \sim \mathcal{D}$
- $w^{(t+1)} \leftarrow w^{(t)} - \eta_t \nabla_w loss\left(h_{w^{(t)}}(x); y\right) = w^{(t)} - \eta_t loss'\left(\langle w^{(t)}, \phi(x)\rangle; y\right)\phi(x)$

Return $\overline{w}^{(T)} = \frac{1}{T}\sum_{t=1}^{T} w^{(t)}$

$h_w(x) = \langle w, \phi(x)\rangle$

SGD suboptimality guarantee: $L_{\mathcal{D}}\left(\overline{w}^{(T)}\right) \leq \inf_{\|w\|_2 \leq B} L_{\mathcal{D}}(w) + \sqrt{\frac{B^2R^2}{T}}$

➔ $m = T = O\left(\frac{B^2R^2}{\epsilon^2}\right)$