Understanding Machine Learning: From Theory to Algorithms (Shalev-Shwartz & Ben-David, 2014)

Ch 12: Convex Learning Problems
Ch 14: Stochastic Gradient Descent

(ML Reading Group, UQ)

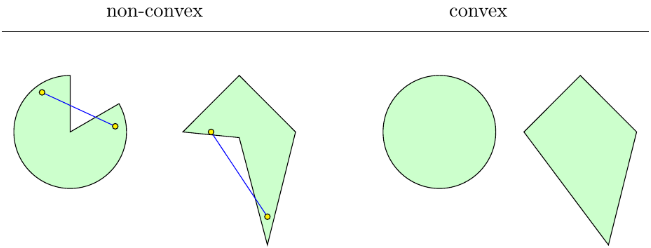
Vektor Dewanto

vektor.dewanto@gmail.com

March 16, 2018

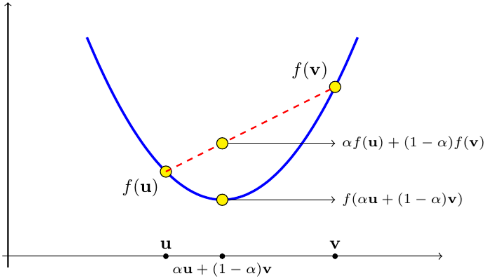
Outline

Convexity: Convex sets



DEFINITION 12.1 (Convex Set) A set C in a vector space is convex if for any two vectors $\mathbf{u}, \mathbf{v}$ in C, the line segment between $\mathbf{u}$ and $\mathbf{v}$ is contained in C. That is, for any $\alpha \in [0, 1]$ we have that $\alpha\mathbf{u} + (1 - \alpha)\mathbf{v} \in C$.

Convexity: Convex functions

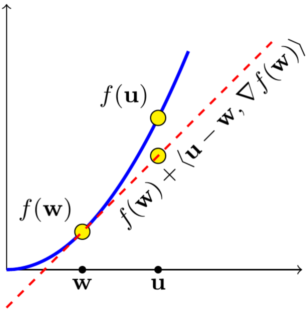


DEFINITION 12.2 (Convex Function) Let C be a convex set. A function $f : C \to \mathbb{R}$ is convex if for every $\mathbf{u}, \mathbf{v} \in C$ and $\alpha \in [0, 1]$,

$$f(\alpha\mathbf{u} + (1-\alpha)\mathbf{v}) \leq \alpha f(\mathbf{u}) + (1-\alpha) f(\mathbf{v}) .$$

Properties of convex fn:

- every local minimum is also a global minimum
- for every **w** we can construct a tangent to f at **w** that lies below f everywhere. If f is differentiable, this tangent is the linear function



Convexity: Convex functions

LEMMA 12.3 *Let $f : \mathbb{R} \to \mathbb{R}$ be a scalar twice differential function, and let f', f'' be its first and second derivatives, respectively. Then, the following are equivalent:*

1. *f is convex*
2. *f' is monotonically nondecreasing*
3. *f'' is nonnegative*

CLAIM 12.4 *Assume that $f : \mathbb{R}^d \to \mathbb{R}$ can be written as $f(\mathbf{w}) = g(\langle \mathbf{w}, \mathbf{x} \rangle + y)$, for some $\mathbf{x} \in \mathbb{R}^d$, $y \in \mathbb{R}$, and $g : \mathbb{R} \to \mathbb{R}$. Then, convexity of g implies the convexity of f.*

CLAIM 12.5 *For $i = 1, \ldots, r$, let $f_i : \mathbb{R}^d \to \mathbb{R}$ be a convex function. The following functions from $\mathbb{R}^d$ to $\mathbb{R}$ are also convex.*

- $g(x) = \max_{i \in [r]} f_i(x)$
- $g(x) = \sum_{i=1}^{r} w_i f_i(x)$, *where for all i, $w_i \geq 0$.*

Lipschitzness

DEFINITION 12.6 (Lipschitzness) Let $C \subset \mathbb{R}^d$. A function $f : \mathbb{R}^d \to \mathbb{R}^k$ is ρ-Lipschitz over C if for every $\mathbf{w}_1, \mathbf{w}_2 \in C$ we have that $\|f(\mathbf{w}_1) - f(\mathbf{w}_2)\| \leq \rho \|\mathbf{w}_1 - \mathbf{w}_2\|$.

Properties:

- a Lipschitz function can **not** change too fast
- if the derivative of f is everywhere bounded (in absolute value) by ρ, then the function is ρ-Lipschitz.
- **composition** of Lipschitz functions **preserves** Lipschitzness.

Smoothness

DEFINITION 12.8 (Smoothness) A differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is β-smooth if its gradient is β-Lipschitz; namely, for all $\mathbf{v}, \mathbf{w}$ we have $\|\nabla f(\mathbf{v}) - \nabla f(\mathbf{w})\| \leq \beta \|\mathbf{v} - \mathbf{w}\|$.

Properties:

- when a function is **both** convex and smooth, we have both **upper and lower bounds** on the difference between the function and its first order approximation.
- a composition of a smooth scalar function over a linear function **preserves** smoothness.

  CLAIM 12.9 *Let $f(\mathbf{w}) = g(\langle \mathbf{w}, \mathbf{x} \rangle + b)$, where $g : \mathbb{R} \to \mathbb{R}$ is a β-smooth function, $\mathbf{x} \in \mathbb{R}^d$, and $b \in \mathbb{R}$. Then, f is $(\beta \|\mathbf{x}\|^2)$-smooth.*

Convex Learning Problems: Intro

Recall, we have:

- a set of examples $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$
- a hypothesis class $\mathcal{H}$, which can be an arbitrary set,
  for now consider: $\mathcal{H} \subseteq \mathbb{R}^d$, thus, denote a hypothesis in $\mathcal{H}$ by $\mathbf{w}$
- a loss function $\ell : \mathcal{H} \times \mathcal{Z} \mapsto \mathbb{R}_+$

DEFINITION 12.10 (Convex Learning Problem) A learning problem, $(\mathcal{H}, Z, \ell)$, is called convex if the hypothesis class $\mathcal{H}$ is a convex set and for all $z \in Z$, the loss function, $\ell(\cdot, z)$, is a convex function (where, for any z, $\ell(\cdot, z)$ denotes the function $f : \mathcal{H} \to \mathbb{R}$ defined by $f(\mathbf{w}) = \ell(\mathbf{w}, z)$).

Convex Learning Problems: Intro

Example 12.7:
Linear Regression with the Squared Loss

- to learn a linear function $h : \mathbb{R}^d \mapsto \mathbb{R}$ that best approximates the relationship between "explanatory" and outcome variables.
- Each linear function is parameterized by a vector $\mathbf{w} \in \mathbb{R}^d$, hence $\mathcal{H} = \mathbb{R}^d$
- set of examples: $\mathcal{Z} = \mathcal{X} \times \mathcal{Y} = \mathbb{R}^d \times \mathbb{R} = \mathbb{R}^{d+1}$
- loss function: $\ell(\mathbf{w}, (\mathbf{x}, y)) = (\langle \mathbf{w}, \mathbf{x} \rangle - y)^2$
- Clearly: $\mathcal{H}$ is a convex set and ℓ is convex fn wrt $\mathbf{w}$

LEMMA 12.11 *If ℓ is a convex loss function and the class $\mathcal{H}$ is convex, then the $\mathrm{ERM}_{\mathcal{H}}$ problem, of minimizing the empirical loss over $\mathcal{H}$, is a convex optimization problem (that is, a problem of minimizing a convex function over a convex set).*

Convex Learning Problems: Learnability

QUESTIONS:

- is convexity a sufficient condition for the learnability of a problem?
- are all convex learning problems over R^d learnable?

ANSWERS:

- **not all** convex learning problems over $\mathbb{R}^d$ are learnable.
- Convex problems are **learnable under** some restricting conditions: the properties of convexity, boundedness, and Lipschitzness or smoothness of the loss function are sufficient for learnability.

Convex Learning Problems: Learnability

TWO families of learning problems are learnable.

DEFINITION 12.12 (Convex-Lipschitz-Bounded Learning Problem) A learning problem, $(\mathcal{H}, Z, \ell)$, is called Convex-Lipschitz-Bounded, with parameters ρ, B if the following holds:

- The hypothesis class $\mathcal{H}$ is a convex set and for all $\mathbf{w} \in \mathcal{H}$ we have $\|\mathbf{w}\| \leq B$.
- For all $z \in Z$, the loss function, $\ell(\cdot, z)$, is a convex and ρ-Lipschitz function.

DEFINITION 12.13 (Convex-Smooth-Bounded Learning Problem) A learning problem, $(\mathcal{H}, Z, \ell)$, is called Convex-Smooth-Bounded, with parameters β, B if the following holds:

- The hypothesis class $\mathcal{H}$ is a convex set and for all $\mathbf{w} \in \mathcal{H}$ we have $\|\mathbf{w}\| \leq B$.
- For all $z \in Z$, the loss function, $\ell(\cdot, z)$, is a convex, nonnegative, and β-smooth function.

Surrogate Loss Fn: Intro

WHAT:
handle some **non**convex problems by minimizing "surrogate" loss functions that are convex

WHY:
the natural loss function is not convex, e.g. $0-1$ loss

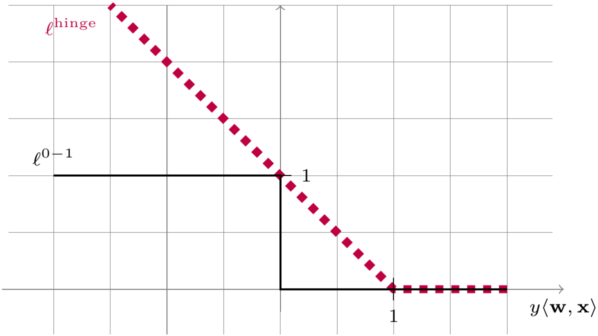
HOW:
to upper bound the nonconvex loss function by a convex surrogate loss function that

- are convex
- upper bounds the original loss.

Surrogate Loss Fn: Example

In the context of learning halfspaces:
Hinge loss as a convex surrogate for the $0 - 1$ loss [1]

$$\ell^{\text{hinge}}(\mathbf{w}, (\mathbf{x}, y)) \stackrel{\text{def}}{=} \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x} \rangle\}$$



[1] https://scicomp.stackexchange.com/questions/5628/confusion-related-to-convexity-of-0-1-loss-function

Gradient Descent: Intro

Recall:

- hypotheses as vectors **w** that
  come from a convex hypothesis class, $\mathcal{H}$
- goal of learning:
  to minimize the risk function $L_D(\mathbf{w})$;
  not the empirical risk $L_S(h)$
- gradient def:

  The gradient of a differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ at $\mathbf{w}$, denoted $\nabla f(\mathbf{w})$, is the vector of partial derivatives of f, namely, $\nabla f(\mathbf{w}) = \left(\frac{\partial f(\mathbf{w})}{\partial w[1]}, \ldots, \frac{\partial f(\mathbf{w})}{\partial w[d]} \right)$.

Gradient descent:

- an iterative optimization procedure
- at each step:
  improve the solution by taking a step along the negative of the gradient of the function to be minimized at the current point

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla f(\mathbf{w}^{(t)}), \tag{14.1}$$

- after T iterations, output either
  - averaged vector, **or**
  - last vector, **or**
  - the best performing vector

LET:

- $\mathbf{w}^*$ be any vector and
- B be an upper bound on $\| \mathbf{w}^* \|$

GOAL:
to obtain an upper bound on $f(\bar{\mathbf{w}}) - f(\mathbf{w}^*)$, where $\bar{\mathbf{w}} = \frac{1}{T} \sum_1^T \mathbf{w}^{(t)}$

RESULT:
From the definition of $\bar{\mathbf{w}}$, and using Jensen's inequality, we obtain:

$$f(\bar{\mathbf{w}}) - f(\mathbf{w}^\star) \;\leq\; \frac{1}{T} \sum_{t=1}^{T} \langle \mathbf{w}^{(t)} - \mathbf{w}^\star, \nabla f(\mathbf{w}^{(t)}) \rangle .$$

LEMMA 14.1 *Let* $\mathbf{v}_1, \ldots, \mathbf{v}_T$ *be an arbitrary sequence of vectors. Any algorithm with an initialization* $\mathbf{w}^{(1)} = 0$ *and an update rule of the form*

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t \tag{14.4}$$

satisfies

$$\sum_{t=1}^{T} \langle \mathbf{w}^{(t)} - \mathbf{w}^\star, \mathbf{v}_t \rangle \leq \frac{\|\mathbf{w}^\star\|^2}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \|\mathbf{v}_t\|^2. \tag{14.5}$$

In particular, for every $B, \rho > 0$*, if for all* t *we have that* $\|\mathbf{v}_t\| \leq \rho$ *and if we set* $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$*, then for every* $\mathbf{w}^\star$ *with* $\|\mathbf{w}^\star\| \leq B$ *we have*

$$\frac{1}{T} \sum_{t=1}^{T} \langle \mathbf{w}^{(t)} - \mathbf{w}^\star, \mathbf{v}_t \rangle \leq \frac{B\,\rho}{\sqrt{T}}.$$

COROLLARY 14.2 *Let f be a convex, ρ-Lipschitz function, and let $\mathbf{w}^\star \in \operatorname{argmin}_{\{\mathbf{w}:\|\mathbf{w}\|\le B\}} f(\mathbf{w})$. If we run the GD algorithm on f for T steps with $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$, then the output vector $\bar{\mathbf{w}}$ satisfies*

$$f(\bar{\mathbf{w}}) - f(\mathbf{w}^\star) \le \frac{B\,\rho}{\sqrt{T}}.$$

Furthermore, for every $\epsilon > 0$, to achieve $f(\bar{\mathbf{w}}) - f(\mathbf{w}^\star) \le \epsilon$, it suffices to run the GD algorithm for a number of iterations that satisfies

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

Subgradients: Intro

WHAT:
apply GD to **non**differentiable convex function

HOW:
use subgradient of $f(\mathbf{w})$ at $\mathbf{w}^{(t)}$, instead of the gradient;
(the analysis of the convergence rate remains unchanged)

Subgradients: Intro

LEMMA 14.3 *Let S be an open convex set. A function $f : S \to \mathbb{R}$ is convex iff for every $\mathbf{w} \in S$ there exists $\mathbf{v}$ such that*

$$\forall \mathbf{u} \in S, \quad f(\mathbf{u}) \geq f(\mathbf{w}) + \langle \mathbf{u} - \mathbf{w}, \mathbf{v} \rangle. \tag{14.8}$$

DEFINITION 14.4 (Subgradients) A vector $\mathbf{v}$ that satisfies Equation (14.8) is called a *subgradient* of f at $\mathbf{w}$. The set of subgradients of f at $\mathbf{w}$ is called the *differential set* and denoted $\partial f(\mathbf{w})$.

Subgradients: Intro

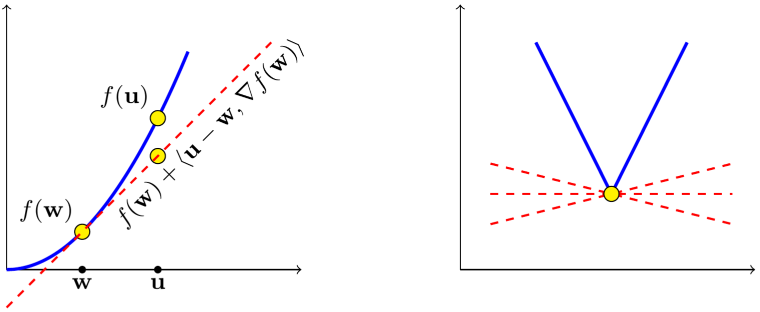


Figure 14.2 Left: The right-hand side of Equation (14.7) is the tangent of f at $\mathbf{w}$. For a convex function, the tangent lower bounds f. Right: Illustration of several subgradients of a nondifferentiable convex function.

Subgradients: Calculating Subgradients

How do we construct subgradients of a given convex function?

For pointwise maximum functions:

CLAIM 14.6 *Let* $g(\mathbf{w}) = \max_{i\in[r]} g_i(\mathbf{w})$ *for* r *convex differentiable functions* $g_1, \ldots, g_r$. *Given some* $\mathbf{w}$, *let* $j \in \operatorname{argmax}_i g_i(\mathbf{w})$. *Then* $\nabla g_j(\mathbf{w}) \in \partial g(\mathbf{w})$.

Example 14.2 (A Subgradient of the Hinge Loss) Recall the hinge loss function from Section 12.3, $f(\mathbf{w}) = \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}\rangle\}$ for some vector $\mathbf{x}$ and scalar y. To calculate a subgradient of the hinge loss at some $\mathbf{w}$ we rely on the preceding claim and obtain that the vector $\mathbf{v}$ defined in the following is a subgradient of the hinge loss at $\mathbf{w}$:

$$\mathbf{v} = \begin{cases} \mathbf{0} & \text{if } 1 - y\langle \mathbf{w},\ \mathbf{x}\rangle \le 0 \\ -y\mathbf{x} & \text{if } 1 - y\langle \mathbf{w},\ \mathbf{x}\rangle > 0 \end{cases}$$

SGD: Intro

WHAT:
Stochastic Gradient Descent (SGD);

WHY:
do not know D, so do not know the gradient of $L_D(w)$.

HOW:
take a step along a **random** direction (vector), as long as
its **expected value** at each iteration will **equal the gradient direction**
(more generally, a subgradient of the function at the current vector)

SGD: Intro

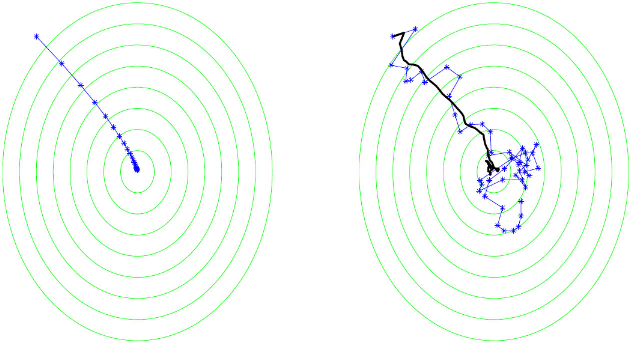


Figure 14.3 An illustration of the gradient descent algorithm (left) and the stochastic gradient descent algorithm (right). The function to be minimized is $1.25(x+6)^2 + (y-8)^2$. For the stochastic case, the black line depicts the averaged value of $\mathbf{w}$.

SGD: Intro

`Stochastic Gradient Descent (SGD) for minimizing`

$$f(\mathbf{w})$$

parameters: Scalar $\eta > 0$, integer $T > 0$
initialize: $\mathbf{w}^{(1)} = \mathbf{0}$
for $t = 1, 2, \ldots, T$
  choose $\mathbf{v}_t$ at random from a distribution such that $\mathbb{E}[\mathbf{v}_t \mid \mathbf{w}^{(t)}] \in \partial f(\mathbf{w}^{(t)})$
  update $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$
output $\bar{\mathbf{w}} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{w}^{(t)}$

SGD: Analysis for Convex-Lipschitz-Bounded Fn

THEOREM 14.8 *Let $B, \rho > 0$. Let f be a convex function and let $\mathbf{w}^\star \in \operatorname{argmin}_{\mathbf{w}:\|\mathbf{w}\| \le B} f(\mathbf{w})$. Assume that SGD is run for T iterations with $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$. Assume also that for all t, $\|\mathbf{v}_t\| \le \rho$ with probability 1. Then,*

$$\mathbb{E}\left[f(\bar{\mathbf{w}})\right] - f(\mathbf{w}^\star) \le \frac{B\,\rho}{\sqrt{T}}.$$

Therefore, for any $\epsilon > 0$, to achieve $\mathbb{E}[f(\bar{\mathbf{w}})] - f(\mathbf{w}^\star) \le \epsilon$, it suffices to run the SGD algorithm for a number of iterations that satisfies

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

Learning with SGD: Risk Minimization

Recall, in learning:

- want to minimize the risk function, $L_D(\mathbf{w}) = \mathbb{E}_{z \sim D}[\ell(\mathbf{w}, z)]$
- do not know D, so cannot simply calculate $\nabla L_D(\mathbf{w}^{(t)})$
- as an estimate to minimizing $L_D(w)$: minimize $L_S(w)$ [2]

SGD minimizes $L_D(w)$ directly:
find an **unbiased estimate** of the gradient of $L_D(\mathbf{w})$, that is,
a random vector whose conditional expected value is $\nabla L_D(\mathbf{w}^{(t)})$

[2] empirical risk

Learning with SGD: Risk Minimization

Construction of the random vector $\mathbf{v}_t$ for a differentiable risk fn L_D:

- sample $z \sim D$
- define $\mathbf{v}_t$ to be the gradient of $\ell(\mathbf{w}, z)$ wrt $\mathbf{w}$, at $\mathbf{w}^{(t)}$
- by the linearity of the gradient we have:

$$\mathbb{E}[\mathbf{v}_t | \mathbf{w}^{(t)}] = \mathop{\mathbb{E}}_{z \sim \mathcal{D}}[\nabla \ell(\mathbf{w}^{(t)}, z)] = \nabla \mathop{\mathbb{E}}_{z \sim \mathcal{D}}[\ell(\mathbf{w}^{(t)}, z)] = \nabla L_{\mathcal{D}}(\mathbf{w}^{(t)}). \qquad (14.13)$$

Thus, the gradient of the loss function $\ell(w, z)$ at $\mathbf{w}^{(t)}$ is

- unbiased estimate of the gradient of the risk function $L_D(w^{(t)})$ and
- constructed by sampling a single fresh example $z \sim D$ at each iteration t.

Learning with SGD: Risk Minimization

Stochastic Gradient Descent (SGD) for minimizing $L_{\mathcal{D}}(\mathbf{w})$

parameters: Scalar $\eta > 0$, integer $T > 0$
initialize: $\mathbf{w}^{(1)} = \mathbf{0}$
for $t = 1, 2, \ldots, T$
  sample $z \sim \mathcal{D}$
  pick $\mathbf{v}_t \in \partial \ell(\mathbf{w}^{(t)}, z)$
  update $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$
output $\bar{\mathbf{w}} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{w}^{(t)}$

Same for nondifferentiable loss functions,
simply let $\mathbf{v}_t$ be a subgradient of $\ell(\mathbf{w}, z)$ at $\mathbf{w}^{(t)}$

Learning with SGD: Convex-Smooth Learning Problems

THEOREM 14.13 *Assume that for all z, the loss function $\ell(\cdot, z)$ is convex, β-smooth, and nonnegative. Then, if we run the SGD algorithm for minimizing $L_{\mathcal{D}}(\mathbf{w})$ we have that for every $\mathbf{w}^\star$,*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}})] \leq \frac{1}{1-\eta\beta}\left(L_{\mathcal{D}}(\mathbf{w}^\star) + \frac{\|\mathbf{w}^\star\|^2}{2\eta T}\right).$$

COROLLARY 14.14 *Consider a convex-smooth-bounded learning problem with parameters β, B. Assume in addition that $\ell(\mathbf{0}, z) \leq 1$ for all $z \in Z$. For every $\epsilon > 0$, set $\eta = \frac{1}{\beta(1+3/\epsilon)}$. Then, running SGD with $T \geq 12B^2\beta/\epsilon^2$ yields*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}})] \leq \min_{\mathbf{w}\in\mathcal{H}} L_{\mathcal{D}}(\mathbf{w}) + \epsilon.$$

Learning with SGD: Regularized Loss Minimization

WHAT:

to solve the regularized loss minimization:

$$\min_{\mathbf{w}} \left(\frac{\lambda}{2} \|\mathbf{w}\|^2 + L_S(\mathbf{w}) \right). \tag{14.14}$$

WHY:

- SGD enjoys the same worst-case sample complexity bound as regularized loss minimization
- on some distributions, regularized loss minimization may yield a better solution.

HOW:

...

Learning with SGD: Regularized Loss Minimization

WHAT:
to solve the regularized loss minimization:

$$\min_{\mathbf{w}} \left(\frac{\lambda}{2} \|\mathbf{w}\|^2 + L_S(\mathbf{w}) \right). \qquad (14.14)$$

HOW:

- define $f(\mathbf{w}) = \frac{\lambda}{2} \parallel w \parallel^2 + L_S(\mathbf{w})$.
  - f is a λ-strongly convex function;
  - therefore, apply the SGD variant with $\mathcal{H} = \mathbb{R}^d$.
- construct an unbiased estimate of a subgradient of f at $\mathbf{w}^{(t)}$
  - pick z uniformly at random from S,
  - choose $\mathbf{v}_t$ in $\partial \ell(\mathbf{w}^{(t)}, z)$
  - (then) the expected value of $\lambda \mathbf{w}^{(t)} + \mathbf{v}t$ is a subgradient of f at $\mathbf{w}^{(t)}$.

Conclusions

SGD can directly minimize the risk function

- by sampling a point i.i.d from D **and**
- using a subgradient of the loss of the current hypothesis at this point as an unbiased estimate of the (sub)gradient of the risk function.

SGD's **number of iterations**
guarantees an expected objective of at most ϵ plus the optimal objective

WARN: Skipped (Subsub)sections:

- any proofs
- 14.2.2 Subgradients of Lipschitz Functions
- 14.4 Variants

References I

Shalev-Shwartz, S., & Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. New York, NY, USA: Cambridge University Press.

Discussion time and thank you.