Machine Learning

Lecture 4

Optimization and Gradient Descent

Vincent Adam & Vicenç Gómez

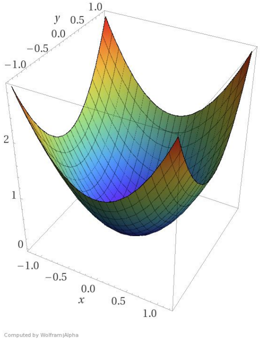
2023-2024

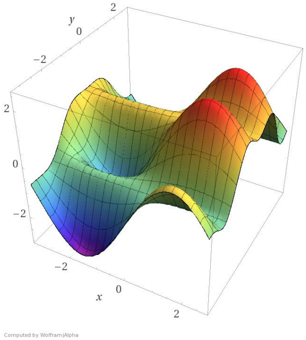
Content

Content

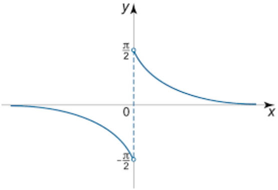
Empirical risk minimization

- Find the hypothesis $h \in \mathcal{H}$ that minimizes the empirical risk $L_S(h)$
- Fundamentally, this is a problem of optimization
- Supervised learning is intrinsically linked to optimization

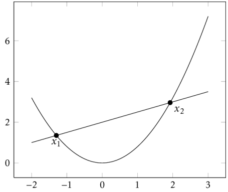




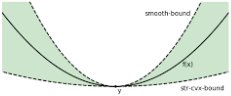
Properties of functions



(Non-)Differentiable

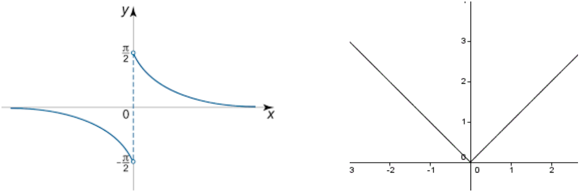


Convex



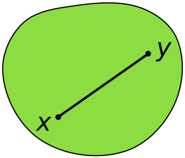
Smooth

Differentiable function



- A function $f : C \to \mathbb{R}$ is differentiable if it is continuous and has a finite gradient in each point $x \in C$
- For continuous functions, we can still compute a sub-gradient

Convex set

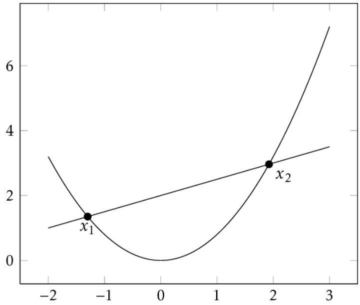


Definition

A set C is convex if, for any two points $x_1, x_2 \in C$, the entire line segment between x_1 and x_2 is also in C. Formally, for any $\alpha \in [0, 1]$,

$$\alpha x_1 + (1 - \alpha) x_2 \in C$$

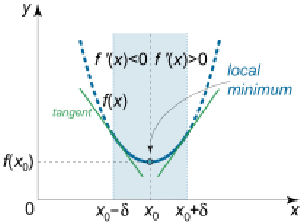
Convex function



Definition

Let C be a convex set. A function $f : C \to \mathbb{R}$ is convex if, for any $x_1, x_2 \in C$ and $\alpha \in [0, 1]$, $f(\alpha x_1 + (1 - \alpha)x_2) \leq \alpha f(x_1) + (1 - \alpha)f(x_2)$

Local minimum

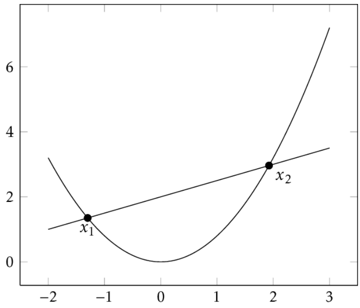


Given $x \in C$ and $\delta \in \mathbb{R}$, let $B(x, \delta) = \{y \in C : \|y - x\| \leq \delta\}$ be a ball of radius δ around x

Definition

Given a function $f : C \to \mathbb{R}$, an element $x_0 \in C$ is a local minimum if there exists $\delta > 0$ such that for each $x \in B(x_0, \delta)$, $f(x_0) \leq f(x)$

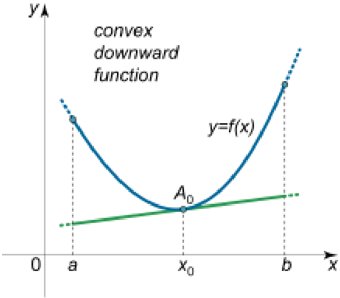
Global minimum



Theorem

Every local minimum of a convex function $f : C \to \mathbb{R}$ is also global.

Tangent

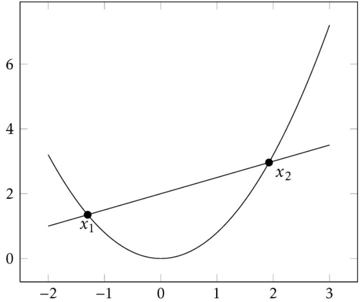


Tangent at $x_0 \in C$: linear function $l(x) = f(x_0) + \langle \nabla f(x_0), x - x_0 \rangle$

Theorem

Given a convex function $f : C \to \mathbb{R}$, the tangent at any point $x_0 \in C$ lies below f, i.e. for each $x \in C$, $f(x_0) + \langle \nabla f(x_0), x - x_0 \rangle \leq f(x)$

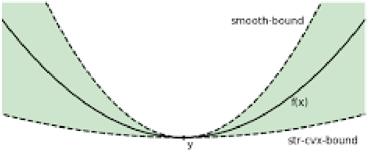
Convexity



For a univariate, twice differentiable function $f : C \to \mathbb{R}$, the following statements are equivalent:

- f is convex
- f' is monotonically non-decreasing
- f'' is non-negative

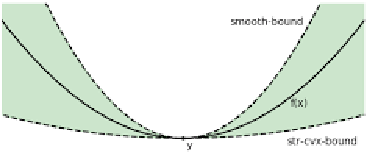
Smoothness



Definition

A differentiable function $f : C \to \mathbb{R}$ is β-smooth if, for each $x_1, x_2 \in C$,
$\|\nabla f(x_2) - \nabla f(x_1)\| \leq \beta \|x_2 - x_1\|$

Smoothness and convexity



- If a function $f : C \to \mathbb{R}$ is β-smooth, then for each $x_1, x_2 \in C$,

$$f(x_2) \leq f(x_1) + \langle \nabla f(x_1), x_2 - x_1 \rangle + \frac{\beta}{2} \|x_2 - x_1\|^2$$

- If the function f is also convex, we obtain

$$0 \leq \underbrace{f(x_2) - f(x_1) - \langle \nabla f(x_1), x_2 - x_1 \rangle}_{D_f(x_1, x_2)} \leq \frac{\beta}{2} \|x_2 - x_1\|^2$$

Convex learning problem

- Consider a supervised learning problem $\langle \mathcal{X}, \mathcal{D}, \mathcal{Y}, f, S \rangle$
- The learner chooses $\langle \mathcal{H}, \ell, \mathcal{A} \rangle$
- The learning problem is convex if the following holds:
  1. The hypothesis class $\mathcal{H}$ is a convex set
  2. The loss ℓ and empirical risk $L_S : \mathcal{H} \to \mathbb{R}$ are convex functions

Recipe for loss functions

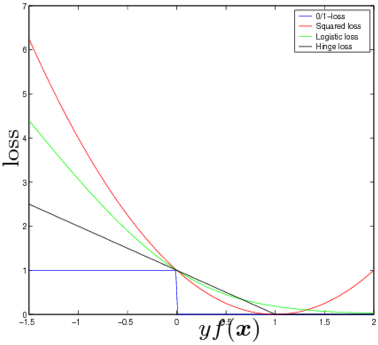
Given convex functions f and g, the following functions are all convex:

1. All norms.
2. $h(x) = a \cdot f(x)$, for any constant $a > 0$.
3. $h(x) = f(x) + g(x)$.
4. $h(x) = \max(f(x), g(x))$.
5. $h(x) = f(Ax + b)$, for any $d \times d$ matrix A and $d \times 1$ vector b.

Learnability of convex learning problems

- For any algorithm $\mathcal{A}$, there are convex learning problems on which it fails
- This holds even if $\mathcal{H}$ is bounded (e.g. $\|w\| \leq B$)
- However, if the loss function is also β-smooth, then the convex learning problem is learnable
- Learnability conditions: convex + bounded + smooth

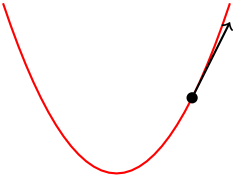
Surrogate loss



- If a learning problem is non-convex, we can approximate it using a convex loss function that upper bounds the original loss
- Example: 0-1 loss

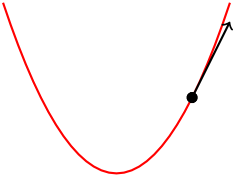
Content

Gradient descent



- Convex, differentiable loss function L_S
- Idea: descend in the opposite direction of the gradient

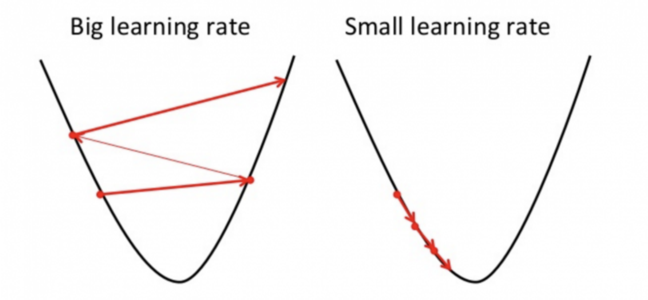
Algorithm



Gradient descent

1. Initialize weight vector $w_0 = 0$
2. Compute the gradient $\nabla_w L_S(w_0)$
3. Update weights as $w_1 \leftarrow w_0 - \eta \nabla_w L_S(w_0)$
4. Repeat from 2. for weight vector w_t, $t = 1, 2, \ldots$

Learning rate



- Learning rate η determines the rate of descent
- Too large: learning oscillates and may never reach minimum
- Too small: learning is slow and may never reach minimum

Convergence of gradient descent

- Let f be a convex and ρ-smooth function
- Let $\mathcal{H} = \{w : \|w\| \leq B\}$ and let $w^* = \arg\min_{w \in \mathcal{H}} f(w)$
- Run gradient descent for T iterations with learning rate $\eta = \frac{B}{\rho\sqrt{T}}$
- Output the average weight vector $\bar{w} = \frac{1}{T}\sum_{t=1}^{T} w_t$

Convergence of gradient descent

- Let f be a convex and ρ-smooth function
- Let $\mathcal{H} = \{w : \|w\| \leq B\}$ and let $w^* = \arg\min_{w \in \mathcal{H}} f(w)$
- Run gradient descent for T iterations with learning rate $\eta = \frac{B}{\rho\sqrt{T}}$
- Output the average weight vector $\bar{w} = \frac{1}{T}\sum_{t=1}^{T} w_t$

Theorem

The output vector $\bar{w}$ satisfies

$$f(\bar{w}) - f(w^*) \leq \frac{B\rho}{\sqrt{T}}$$

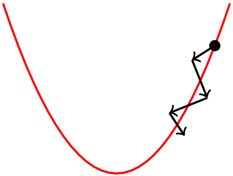
For a desired accuracy ϵ, the number of required iterations is

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

Properties of gradient descent

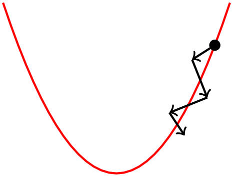
- Provably converges if learning problem is convex and learnable
- However, computing $\nabla_w L_S(w_t)$ requires iterating over S
- If the training set S is large, gradient descent is inefficient

Stochastic gradient descent (SGD)



- Idea: compute partial gradient on subset of data points
- Each partial gradient does not coincide with the full gradient
- In expectation, partial gradients descend towards minimum

Algorithm



SGD

1. Initialize weight vector $w_0 = 0$
2. Compute a partial gradient v_0 such that $\mathbb{E}[v_0 | w_0] = \nabla_w L_S(w_0)$
3. Update weights as $w_1 \leftarrow w_0 - \eta v_0$
4. Repeat from 2. for weight vector w_t, $t = 1, 2, \ldots$

Partial gradient

- Given weight vector w_t, several ways to compute partial gradient:

  1. Compute gradient of loss function $\ell(h(w_t, x), y)$ on data point (x, y)
  2. Compute partial gradient on mini-batch $S' \subset S$

Convergence of stochastic gradient descent

- Let f be a convex and ρ-smooth function
- Let $\mathcal{H} = \{w : \|w\| \leq B\}$ and let $w^* = \arg\min_{w \in \mathcal{H}} f(w)$
- Run SGD for T iterations with learning rate $\eta = \frac{B}{\rho\sqrt{T}}$
- Output the average weight vector $\bar{w} = \frac{1}{T} \sum_{t=1}^{T} w_t$

Convergence of stochastic gradient descent

- Let f be a convex and ρ-smooth function
- Let $\mathcal{H} = \{w : \|w\| \leq B\}$ and let $w^* = \arg\min_{w \in \mathcal{H}} f(w)$
- Run SGD for T iterations with learning rate $\eta = \frac{B}{\rho\sqrt{T}}$
- Output the average weight vector $\bar{w} = \frac{1}{T}\sum_{t=1}^{T} w_t$

Theorem

The output vector $\bar{w}$ satisfies

$$\mathbb{E}[f(\bar{w})] - f(w^*) \leq \frac{B\rho}{\sqrt{T}}$$

For a desired accuracy ϵ, the number of required iterations is

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

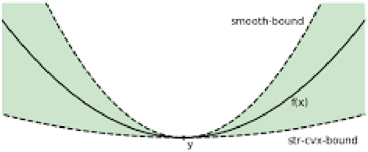
Projection

- Problem: updated weight vector may not be bounded (i.e. in $\mathcal{H}$)
- Solution: project the weight vector back onto $\mathcal{H}$
- The convergence proof remains the same

SGD with projection

1. Initialize weight vector $w_0 = 0$
2. Compute a partial gradient v_0 such that $\mathbb{E}[v_0 | w_0] = \nabla_w L_S(w_0)$
3. Update weights as $w_{\frac{1}{2}} \leftarrow w_0 - \eta v_0$
4. Project weights as $w_1 \leftarrow \arg\min_{w \in \mathcal{H}} \| w - w_{\frac{1}{2}} \|$
5. Repeat from 2. for weight vector w_t, $t = 1, 2, \ldots$

Strong convexity



Definition

A differentiable function $f : C \to \mathbb{R}$ is α-strongly convex if, for each $x_1, x_2 \in C$,

$$\frac{\alpha}{2}\|x_2 - x_1\|^2 \leq f(x_2) - f(x_1) - \langle \nabla f(x_1), x_2 - x_1 \rangle$$

Strong convexity

- Let f be a λ-strongly convex and ρ-smooth function
- Let $\mathcal{H} = \{w : \|w\| \leq B\}$ and let $w^* = \arg\min_{w \in \mathcal{H}} f(w)$
- Run SGD for T iterations with learning rate $\eta = \frac{1}{\lambda t}$
- Output the average weight vector $\bar{w} = \frac{1}{T} \sum_{t=1}^{T} w_t$

Strong convexity

- Let f be a λ-strongly convex and ρ-smooth function
- Let $\mathcal{H} = \{w : \|w\| \leq B\}$ and let $w^* = \arg\min_{w \in \mathcal{H}} f(w)$
- Run SGD for T iterations with learning rate $\eta = \frac{1}{\lambda t}$
- Output the average weight vector $\bar{w} = \frac{1}{T}\sum_{t=1}^{T} w_t$

Theorem

The output vector $\bar{w}$ satisfies

$$\mathbb{E}[f(\bar{w})] - f(w^*) \leq \frac{\rho^2}{2\lambda T}(1 + \log(T))$$

Regularization

- In regularization, we minimize the augmented loss function

$$L_{aug}(w) = L_S(w) + \frac{\lambda}{2}\|w\|^2$$

- The augmented loss function is λ-strongly convex!

Perceptron learning algorithm

The perceptron learning algorithm can be viewed as stochastic gradient descent with $\eta = 1$ and $v_t = -y_i x_i$!

Perceptron learning algorithm

1. Initialize weight vector $w_0 = 0$
2. Find a mistake (x_i, y_i) such that $h(x_i) \neq y_i$
3. Update weights as $w_1 \leftarrow w_0 + y_i x_i$
4. Repeat from 2. for weight vector w_t, $t = 1, 2, \ldots$

Practical advise for stochastic gradient descent

- Use largest mini-batch size that your computer can handle
- (Optional) Choose a momentum for mixing the gradient with the previous gradient
- Choose largest learning rate η that does not cause divergence
- Once learning plateaus, reduce η e.g. dividing by 10 and repeat

Convergence of prediction errors

- Track evolution of prediction errors rather than weights

$$(\hat{y}_{t+1} - y) = \Phi(\eta, \hat{y}_t - y)$$

- Sometimes (stochastic) gradient descent converges on the prediction errors even when it does not converge on the loss function!

Content

Constrained optimization

- Minimize a function subject to a set of constraints
- A constrained optimization problem can be written as

$$\begin{aligned} &\min_{w} f(w) \\ s.t.\ & g_i(w) = 0,\ \forall i = 1, \ldots, n \\ & h_j(w) \geq 0,\ \forall j = 1, \ldots, k \end{aligned}$$

Lagrangian

- For equality constraints, problems are solved using a Lagrangian
- Consider the constrained optimization problem

$$\min_{w} f(w)$$
$$s.t.\ g_i(w) = 0,\ \forall i = 1, \ldots, n$$

- The Lagrangian is given by

$$\mathcal{L}(w, \lambda) = f(w) + \sum_i \lambda_i g_i(w)$$

- The elements of λ are Lagrange multipliers

Dual

- To solve for w we set the gradient of the Lagrangian to 0:

$$\nabla_w \mathcal{L}(w^*, \lambda) = 0$$

- The dual optimization problem is given by

$$\max_{\lambda} g(\lambda) = \mathcal{L}(w^*, \lambda)$$

- To obtain λ^* we have to solve the dual

KKT conditions

- For inequality constraints, problems are solved using Karush–Kuhn–Tucker (KKT) conditions
- Consider the constrained optimization problem

$$\min_w f(w)$$
$$s.t.\ h_j(w) \geq 0,\ \forall j = 1, \ldots, k$$

- The Lagrangian is given by

$$\mathcal{L}(w, \beta) = f(w) + \sum_j \beta_j h_j(w)$$

- The KKT conditions are given by

$$\nabla_w \mathcal{L}(w, \beta) = 0$$
$$\beta_j h_j(w) = 0 \ \ \forall j = 1, \ldots, k$$

Linear and quadratic optimization

- In linear optimization, function f and constraints g, h are linear

$$\begin{aligned} \min_{w} f(w) &= c^\top w \\ s.t.\ Gw &= 0 \\ Hw &\geq 0 \end{aligned}$$

- In quadratic optimization, function f is quadratic

$$f(x) = w^\top A w + c^\top w$$

Content

Convexity of linear regression

- Show that the univariate function $f(x) = x^2$ is convex

Convexity of linear regression

$$\begin{cases} f(x) = x^2 \\ \\ \end{cases}$$

Convexity of linear regression

$$\begin{cases} f(x) = x^2 \\ f'(x) = 2x \\ \end{cases}$$

Convexity of linear regression

$$\begin{cases} f(x) = x^2 \\ f'(x) = 2x \\ f''(x) = 2 > 0 \end{cases}$$

Convexity of linear regression

$$\begin{cases} f(x) = x^2 \\ f'(x) = 2x \\ f''(x) = 2 > 0 \end{cases}$$

Convexity of linear regression

$$\begin{cases} f(x) = x^2 \\ f'(x) = 2x \\ f''(x) = 2 > 0 \end{cases}$$

Since f is univariate and $f''(x)$ is non-negative, f is convex

Convexity of linear regression

Alternative proof:

$$\begin{cases} \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1 - x_2)^2 \\ \\ \\ \end{cases}$$

Convexity of linear regression

Alternative proof:

$$\left\{ \begin{aligned} &\alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1 - x_2)^2 \\ &= \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1^2 + x_2^2 - 2x_1 x_2) \end{aligned} \right.$$

Convexity of linear regression

Alternative proof:

$$
\left\{
\begin{aligned}
&\alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1 - x_2)^2 \\
&= \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1^2 + x_2^2 - 2x_1x_2) \\
&= \alpha^2 x_1^2 + (1-\alpha)^2 x_2^2 + 2\alpha(1-\alpha)x_1x_2
\end{aligned}
\right.
$$

Convexity of linear regression

Alternative proof:

$$
\begin{cases}
\begin{aligned}
&\alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1 - x_2)^2 \\
&= \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1^2 + x_2^2 - 2x_1x_2) \\
&= \alpha^2 x_1^2 + (1-\alpha)^2 x_2^2 + 2\alpha(1-\alpha)x_1x_2 \\
&= (\alpha x_1 + (1-\alpha)x_2)^2
\end{aligned}
\end{cases}
$$

Convexity of linear regression

Alternative proof:

$$
\left\{
\begin{aligned}
& \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1 - x_2)^2 \\
& = \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1^2 + x_2^2 - 2x_1x_2) \\
& = \alpha^2 x_1^2 + (1-\alpha)^2 x_2^2 + 2\alpha(1-\alpha)x_1x_2 \\
& = (\alpha x_1 + (1-\alpha)x_2)^2 \\
\Rightarrow f(\alpha x_1 + (1-\alpha)x_2) & = \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1 - x_2)^2
\end{aligned}
\right.
$$

Convexity of linear regression

Alternative proof:

$$
\begin{cases}
\begin{aligned}
& \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1 - x_2)^2 \\
& = \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1^2 + x_2^2 - 2x_1x_2) \\
& = \alpha^2 x_1^2 + (1-\alpha)^2 x_2^2 + 2\alpha(1-\alpha)x_1x_2 \\
& = (\alpha x_1 + (1-\alpha)x_2)^2 \\
\Rightarrow f(\alpha x_1 + (1-\alpha)x_2) & = \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1 - x_2)^2 \\
& \leq \alpha f(x_1) + (1-\alpha)f(x_2) + 0
\end{aligned}
\end{cases}
$$

Convexity of linear regression

Alternative proof:

$$
\begin{cases}
\begin{aligned}
& \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1 - x_2)^2 \\
&= \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1^2 + x_2^2 - 2x_1 x_2) \\
&= \alpha^2 x_1^2 + (1-\alpha)^2 x_2^2 + 2\alpha(1-\alpha)x_1 x_2 \\
&= (\alpha x_1 + (1-\alpha)x_2)^2 \\
\Rightarrow f(\alpha x_1 + (1-\alpha)x_2) &= \alpha x_1^2 + (1-\alpha)x_2^2 - \alpha(1-\alpha)(x_1 - x_2)^2 \\
&\leq \alpha f(x_1) + (1-\alpha)f(x_2) + 0 \\
&= \alpha f(x_1) + (1-\alpha)f(x_2)
\end{aligned}
\end{cases}
$$

Convexity of logistic regression

- Show that the univariate function $f(x) = \ln(1 + \exp(x))$ is convex

Convexity of logistic regression

$$\begin{cases} f(x) = \ln(1 + \exp(x)) \\ \\ \\ \end{cases}$$

Convexity of logistic regression

$$\begin{cases} f(x) = \ln(1+\exp(x)) \\ f'(x) = \dfrac{1}{1+\exp(x)}\exp(x) = \dfrac{\exp(x)}{\exp(x)}\dfrac{1}{\exp(-x)+1} = \dfrac{1}{1+\exp(-x)} \\ \end{cases}$$

Convexity of logistic regression

$$\begin{cases} f(x) = \ln(1+\exp(x)) \\ f'(x) = \dfrac{1}{1+\exp(x)}\exp(x) = \dfrac{\exp(x)}{\exp(x)}\dfrac{1}{\exp(-x)+1} = \dfrac{1}{1+\exp(-x)} \\ f''(x) = -\dfrac{1}{(1+\exp(-x))^2}(-\exp(-x)) = \dfrac{\exp(-x)}{(1+\exp(-x))^2} > 0 \end{cases}$$

Convexity of logistic regression

$$\begin{cases} f(x) = \ln(1 + \exp(x)) \\ f'(x) = \dfrac{1}{1+\exp(x)} \exp(x) = \dfrac{\exp(x)}{\exp(x)} \dfrac{1}{\exp(-x)+1} = \dfrac{1}{1+\exp(-x)} \\ f''(x) = -\dfrac{1}{(1+\exp(-x))^2} (-\exp(-x)) = \dfrac{\exp(-x)}{(1+\exp(-x))^2} > 0 \end{cases}$$

Convexity of logistic regression

$$\begin{cases} f(x) = \ln(1 + \exp(x)) \\ f'(x) = \dfrac{1}{1 + \exp(x)} \exp(x) = \dfrac{\exp(x)}{\exp(x)} \dfrac{1}{\exp(-x) + 1} = \dfrac{1}{1 + \exp(-x)} \\ f''(x) = -\dfrac{1}{(1 + \exp(-x))^2} (-\exp(-x)) = \dfrac{\exp(-x)}{(1 + \exp(-x))^2} > 0 \end{cases}$$

Since f is univariate and $f''(x)$ is non-negative, f is convex

Convexity of norms

- Show that any norm $\|\cdot\|$ is convex

Convexity of norms

- Any norm satisfies the triangle inequality $\|v + w\| \leq \|v\| + \|w\|$

Convexity of norms

- Any norm satisfies the triangle inequality $\|v + w\| \leq \|v\| + \|w\|$
- Hence for any vectors v, w and any $\alpha \in [0, 1]$ we have

$$\|\alpha v + (1 - \alpha) w\| \leq \|\alpha v\| + \|(1 - \alpha) w\| = \alpha \|v\| + (1 - \alpha) \|w\|$$

Convexity of norms

- Any norm satisfies the triangle inequality $\|v + w\| \leq \|v\| + \|w\|$
- Hence for any vectors v, w and any $\alpha \in [0, 1]$ we have

$$\|\alpha v + (1 - \alpha)w\| \leq \|\alpha v\| + \|(1 - \alpha)w\| = \alpha\|v\| + (1 - \alpha)\|w\|$$

- This is precisely the definition of a convex function!

Smoothness of linear regression

- Show that the univariate function $f(x) = x^2$ is 2-smooth

Smoothness of linear regression

- We first show that if f is univariate and $f''(x) \leq \beta$ for some constant β, then f is β-smooth

Smoothness of linear regression

- We first show that if f is univariate and $f''(x) \leq \beta$ for some constant β, then f is β-smooth
- Since $f''(x) \leq \beta$, for any x_1, x_2 we have

$$f'(x_2) - f'(x_1) \leq \beta(x_2 - x_1),$$

which is the definition of β-smooth for univariate functions

Smoothness of linear regression

- We first show that if f is univariate and $f''(x) \leq \beta$ for some constant β, then f is β-smooth
- Since $f''(x) \leq \beta$, for any x_1, x_2 we have

$$f'(x_2) - f'(x_1) \leq \beta(x_2 - x_1),$$

which is the definition of β-smooth for univariate functions

- For $f(x) = x^2$, we have $f''(x) = 2$, implying that f is 2-smooth

Smoothness of logistic regression

- Show that the univariate function $f(x) = \ln(1 + \exp(x))$ is $\frac{1}{4}$-smooth

Smoothness of logistic regression

For $f(x) = \ln(1 + \exp(x))$, we already derived the following expression of $f''(x)$:

$$\begin{cases} f''(x) = \dfrac{\exp(-x)}{(1 + \exp(-x))^2} \end{cases}$$

Smoothness of logistic regression

For $f(x) = \ln(1 + \exp(x))$, we already derived the following expression of $f''(x)$:

$$\begin{cases} f''(x) = \dfrac{\exp(-x)}{(1 + \exp(-x))^2} \\ \qquad\quad = \dfrac{\exp(-x)}{\exp(-x)} \dfrac{1}{(1 + \exp(-x))(1 + \exp(x))} \end{cases}$$

Smoothness of logistic regression

For $f(x) = \ln(1 + \exp(x))$, we already derived the following expression of $f''(x)$:

$$
\begin{cases}
f''(x) = \dfrac{\exp(-x)}{(1+\exp(-x))^2} \\
 = \dfrac{\exp(-x)}{\exp(-x)} \dfrac{1}{(1+\exp(-x))(1+\exp(x))} \\
 = \dfrac{1}{(1+\exp(-x))(1+\exp(x))}
\end{cases}
$$

Smoothness of logistic regression

For $f(x) = \ln(1 + \exp(x))$, we already derived the following expression of $f''(x)$:

$$\begin{cases} f''(x) &= \dfrac{\exp(-x)}{(1+\exp(-x))^2} \\ &= \dfrac{\exp(-x)}{\exp(-x)} \dfrac{1}{(1+\exp(-x))(1+\exp(x))} \\ &= \dfrac{1}{(1+\exp(-x))(1+\exp(x))} \\ &\leq \dfrac{1}{4} \end{cases}$$

Smoothness of logistic regression

For $f(x) = \ln(1 + \exp(x))$, we already derived the following expression of $f''(x)$:

$$\begin{cases} f''(x) &= \dfrac{\exp(-x)}{(1+\exp(-x))^2} \\ &= \dfrac{\exp(-x)}{\exp(-x)} \dfrac{1}{(1+\exp(-x))(1+\exp(x))} \\ &= \dfrac{1}{(1+\exp(-x))(1+\exp(x))} \\ &\leq \dfrac{1}{4} \end{cases}$$

Smoothness of logistic regression

For $f(x) = \ln(1 + \exp(x))$, we already derived the following expression of $f''(x)$:

$$\begin{cases} f''(x) &= \dfrac{\exp(-x)}{(1+\exp(-x))^2} \\ &= \dfrac{\exp(-x)}{\exp(-x)} \dfrac{1}{(1+\exp(-x))(1+\exp(x))} \\ &= \dfrac{1}{(1+\exp(-x))(1+\exp(x))} \\ &\leq \dfrac{1}{4} \end{cases}$$

Hence f is $\frac{1}{4}$-smooth

Constrained optimization

- Solve the following constrained optimization problem in 2 dimensions:

$$\min_{w} 2w_1^2 + w_2^2$$
$$s.t.\ w_1 + w_2 = 1$$

Constrained optimization

- First form the Lagrangian:

$$\mathcal{L}(w, \lambda) = 2w_1^2 + w_2^2 + \lambda(1 - w_1 - w_2)$$

Constrained optimization

- First form the Lagrangian:

$$\mathcal{L}(w,\lambda) = 2w_1^2 + w_2^2 + \lambda(1 - w_1 - w_2)$$

- Then set the gradient of the Lagrangian equal to 0

$$\nabla_w \mathcal{L}(w,\lambda) = \begin{pmatrix} \frac{\partial \mathcal{L}(w,\lambda)}{\partial w_1} \\ \frac{\partial \mathcal{L}(w,\lambda)}{\partial w_2} \end{pmatrix} = \begin{pmatrix} 4w_1 - \lambda \\ 2w_2 - \lambda \end{pmatrix} = 0$$

Constrained optimization

- First form the Lagrangian:

$$\mathcal{L}(w, \lambda) = 2w_1^2 + w_2^2 + \lambda(1 - w_1 - w_2)$$

- Then set the gradient of the Lagrangian equal to 0

$$\nabla_w \mathcal{L}(w, \lambda) = \begin{pmatrix} \frac{\partial \mathcal{L}(w,\lambda)}{\partial w_1} \\ \frac{\partial \mathcal{L}(w,\lambda)}{\partial w_2} \end{pmatrix} = \begin{pmatrix} 4w_1 - \lambda \\ 2w_2 - \lambda \end{pmatrix} = 0$$

- Solving for w gives us the following expression for w^*:

$$w^* = \frac{\lambda}{4} \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Constrained optimization

- Now form the dual by inserting w^* into the Lagrangian:

$$
\begin{aligned}
g(\lambda) = \mathcal{L}(w^*, \lambda) &= \frac{2\lambda^2}{16} + \frac{4\lambda^2}{16} + \lambda(1 - \frac{\lambda}{4} - \frac{2\lambda}{4}) \\
&= \frac{3\lambda^2}{8} + \lambda(1 - \frac{3\lambda}{4}) = \lambda(1 - \frac{3\lambda}{8})
\end{aligned}
$$

Constrained optimization

- Now form the dual by inserting w^* into the Lagrangian:

$$g(\lambda) = \mathcal{L}(w^*, \lambda) = \frac{2\lambda^2}{16} + \frac{4\lambda^2}{16} + \lambda(1 - \frac{\lambda}{4} - \frac{2\lambda}{4})$$
$$= \frac{3\lambda^2}{8} + \lambda(1 - \frac{3\lambda}{4}) = \lambda(1 - \frac{3\lambda}{8})$$

- Maximize the dual by setting the gradient to 0

$$\nabla_\lambda g(\lambda) = 1 - \frac{3\lambda}{8} - \frac{3\lambda}{8} = 1 - \frac{3\lambda}{4} = 0$$

Constrained optimization

- Now form the dual by inserting w^* into the Lagrangian:

$$\begin{aligned} g(\lambda) = \mathcal{L}(w^*, \lambda) &= \frac{2\lambda^2}{16} + \frac{4\lambda^2}{16} + \lambda(1 - \frac{\lambda}{4} - \frac{2\lambda}{4}) \\ &= \frac{3\lambda^2}{8} + \lambda(1 - \frac{3\lambda}{4}) = \lambda(1 - \frac{3\lambda}{8}) \end{aligned}$$

- Maximize the dual by setting the gradient to 0

$$\nabla_\lambda g(\lambda) = 1 - \frac{3\lambda}{8} - \frac{3\lambda}{8} = 1 - \frac{3\lambda}{4} = 0$$

- Solving yields $\lambda^* = \frac{4}{3}$, which we can insert into x^* to obtain

$$x^* = \frac{1}{4} \cdot \frac{4}{3} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \frac{1}{3} \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Constrained optimization

- Solve the following constrained optimization problem in 1 dimension:

$$\min_{w} \, (-w^2)$$
$$s.t. \; w \leq 5$$
$$w \geq 0$$

Constrained optimization

- First form the Lagrangian:

$$\mathcal{L}(w, \beta) = -w^2 + \beta_1(5 - w) + \beta_2 w$$

Constrained optimization

- First form the Lagrangian:

$$\mathcal{L}(w, \beta) = -w^2 + \beta_1(5 - w) + \beta_2 w$$

- The KKT conditions are given by

$$\nabla_w \mathcal{L}(w, \lambda) = -2w - \beta_1 + \beta_2 = 0 \quad (1)$$

$$\beta_1(5 - w) = 0 \quad (2)$$

$$\beta_2 w = 0 \quad (3)$$

Constrained optimization

- First form the Lagrangian:

$$\mathcal{L}(w, \beta) = -w^2 + \beta_1(5 - w) + \beta_2 w$$

- The KKT conditions are given by

$$\nabla_w \mathcal{L}(w, \lambda) = -2w - \beta_1 + \beta_2 = 0 \quad (1)$$
$$\beta_1(5 - w) = 0 \quad (2)$$
$$\beta_2 w = 0 \quad (3)$$

- Analyze by cases:

$$w = 0 : \beta_1 = 0\ (2), \beta_2 = 0\ (1), -w^2 = 0$$
$$w = 5 : \beta_2 = 0\ (3), \beta_1 = -10\ (1), -w^2 = -25$$
$$\text{other } w : \beta_1 = 0\ (2), \beta_2 = 0\ (3), w = 0\ (1), -w^2 = 0$$