Section 4: The LASSO and gradient descent

How to use a gradient descent to solve the LASSO?

The main difficulty to overcome is non-differentiability of the absolute value function. The trick is to replace the absolute value with a differentiable function.

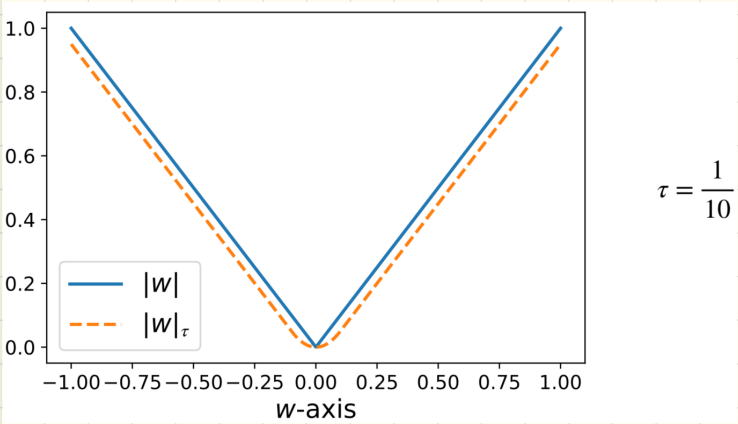
$$|x| = \max_{p \in [-1,1]} xp$$

$$|x|_\tau := \max_{p \in [-1,1]} xp - \frac{\tau}{2} p^2, \quad \tau > 0$$

↑ small perturbation

$|x|_\tau$ - ?

- $|x| \le \tau \Rightarrow |x|_\tau = \frac{x^2}{2\tau}$
- $|x| > \tau \Rightarrow |x|_\tau = |x| - \frac{\tau}{2}$



$\tau = \frac{1}{10}$

Properties of the smoothing:

1. The smooth absolute value is differentiable.

$$\frac{d}{dw}|w|_\tau = \begin{cases} 1, & w > \tau \\ \frac{1}{\tau} w, & |w| \le \tau \\ -1, & w < -\tau \end{cases}$$

2. The smooth absolute value is close to abs.

$$|w_\tau| \le |w| \le |w_\tau| + \frac{\tau}{2}.$$

$$w_\alpha^\tau = \arg\min_w \frac{1}{2}\|\Phi(X)w - y\|^2 + \alpha H_\tau(w)$$

$$H_\tau(w) := \sum_{i=0}^{d} |w_i|_\tau .$$

Can we use normal equation? - No, the system is not linear anymore.
Can we use gradient descent? - Yes, with minor upgrade.

$$E_\tau(w) = \frac{1}{2}\|Xw - y\|^2 + \alpha H_\tau(w)$$

$$w^{k+1} = w^k - \tau\left[X^T(Xw^k - y) + \alpha \nabla H_\tau(w^k)\right]$$

! in practice

forward-splitting

$$w^{k+1/2} = w^k - \frac{\tau}{\alpha} X^T(Xw^k - y)$$

$$w^k = w^{k+1/2} - \tau \nabla H_\tau(w^{k+1/2})$$

Remark: $w - \tau \nabla |w_\tau| = \begin{cases} w - \tau, & w \ge \tau \\ 0 \\ w + \tau, & w \le -\tau \end{cases} =: \text{soft}_\tau(w)$

soft-tresholding

$$w_j^{k+1} = \text{soft}_\tau\left[\left(w^k - \frac{\tau}{\alpha} X^T(Xw^k - y)\right)_j\right]$$

This is known as ISTA (iterative soft-tresholding algorithm) and is a special case of proximal gradient descent

$$w^{k+1} = (1 + \tau \partial R)^{-1}\left(w^k - \tau \nabla L(w^k)\right)$$

proximal map → $(1+\tau\partial R)^{-1}(z) := \arg\min_w \left\{\frac{1}{2}\|w-z\|^2 + \tau R(w)\right\}$

$$L = \frac{1}{2d}\|Xw - y\|^2$$

$$R = \tau |x|$$

Section 5 Proximal gradient descent

Recap: The LASSO problem consist of

$$w_\alpha = \arg\min_w \left\{ \frac{1}{2}\|Xw - y\|^2 + \alpha H_\tau(w) \right\}$$

where $H_\tau(w) = \sum_{j=0}^{d} |w_j|_\tau$, and $|x|_\tau = \begin{cases} |x| - \frac{\tau}{2}, & |x| > \tau \\ \frac{1}{2\tau}|x|^2, & |x| \le \tau \end{cases}$

Gradient descent in this case has a form

$$w^{k+1} = w^k - \tau \left[X^\top (Xw^k - y) + \alpha \nabla H_\tau(w^k) \right]$$

Alternative: proximal gradient descent

Suppose we want to minimise $E(w) = L(w) + R(w)$

where

1. L is convex, continuously differentiable
2. R is proper, convex, lower-semi continuous

and has a simple proximal map, i.e.

proximal map

$$\text{prox}_{\tau R}(z) := \arg\min_x \left[\frac{1}{2}\|x - z\|^2 + \tau R(x) \right]$$

↑ distance to z

is easy to compute

Def. The proximal gradient descent method is an iterative procedure of the form

$$w^{k+1} = \text{prox}_{\tau R}\left(w^{(k)} - \tau \nabla L(w^{(k)}) \right)$$

for the energy $E(w) = L(w) + R(w)$

If additionally to the above R is differentiable

$$\nabla \left(\frac{1}{2}\|x-z\|^2 + \tau R(x) \right) = x - z + \tau \nabla R(x) = 0$$

$$w^{k+1} + \tau \nabla R(w^{k+1}) - \left(w^k - \tau \nabla L(w^k) \right) = 0$$

$$\boxed{w^{k+1} = w^k - \tau \left(\nabla L(w^k) + \nabla R(w^{k+1}) \right)}$$

↑ implicit - explicit gradient method

Example: $R(w) = \frac{1}{2}\|w\|^2$

$$\text{prox}_{\tau R}(z) = \arg\min_x \left\{ \underbrace{\frac{1}{2}\|x-z\|^2 + \frac{\tau}{2}\|x\|^2}_{E(x)} \right\}$$

$$\nabla E(x) = x - z + \tau x = 0$$

$$\boxed{\hat{x} = \frac{1}{1+\tau} \cdot z}$$

$$\left(1 + \tau \, \partial \, \tfrac{1}{2}\|\cdot\|^2\right)^{-1}(z) = \frac{z}{1+\tau}$$

Example:

$$R(w) = \begin{cases} 0, & w \in C \\ \infty, & w \notin C \end{cases}$$

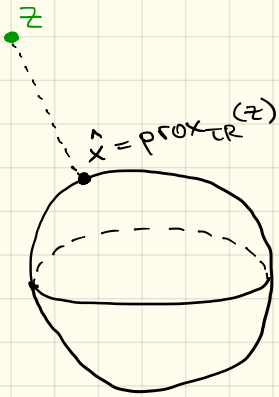
↑ convex set

Remark: This corresponds to a minimisation problem with constraint $w \in C$.

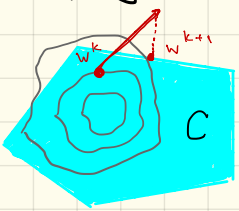
$$\text{prox}_{\tau R}(z) = \arg\min_x \left[\frac{1}{2}\|x-z\|^2 + \tau R(x) \right]$$

$$= \arg\min_{x \in C} \frac{1}{2}\|x-z\|^2$$

↑ projection onto convex set



$$w^{k+1} = \text{proj}_C\left(w^k - \tau \nabla L(w^k)\right)$$



Example: $C = \{x \in \mathbb{R} : x \geq 0\}$

$\text{proj}_C(z) = \max(0, z)$ [Exercise]

$w^{k+1} = \max(0,\ w^k - \tau \nabla L(w^k))$

↑ ensures that the iterate stays positive.

What is the motivation of proximal gradient descent?

$$w^{k+1} = \arg\min_w \{ L(w) + R(w) + D_J(w, w^k) \}$$

where $J = \frac{1}{2\tau}\|w\|^2 - L(w)$

$$D_J(u, v) = J(u) - J(v) - \langle \nabla J(v), u - v \rangle$$

↑ analogue of $\frac{1}{2} J''(v) \cdot (u-v)^2$ in $d = 1$

$$w^{k+1} = \arg\min_w \{ L(w) + R(w) + \tfrac{1}{2\tau} D_{\|\cdot\|^2}(w, w^k) - D_L(w, w^k) \}$$

$$= \arg\min_w \{ L(w^k) - \langle \nabla L(w^k), w - w^k \rangle + R(w) + \tfrac{1}{2\tau}(\|w\|^2 - \|w^k\|^2 - 2\langle w^k, w - w^k \rangle) \}$$

$$= \arg\min_w \{ \tau R(w) + \tfrac{1}{2}\|w - w^k\|^2 - \tau \langle \nabla L(w^k), w - w^k \rangle \}$$

$$= \arg\min_w \{ \tfrac{1}{2}\|w - (w^k - \tau \nabla L(w^k))\|^2 + \tau R(w) \}$$

How do we use proximal gradient descent to solve the LASSO problem?

$$\hat{w}_\alpha = \arg\min_w \left\{ \underbrace{\frac{1}{2}\|Xw - y\|^2}_{L(w)} + \underbrace{\alpha \|w\|_1}_{R(w)} \right\}$$

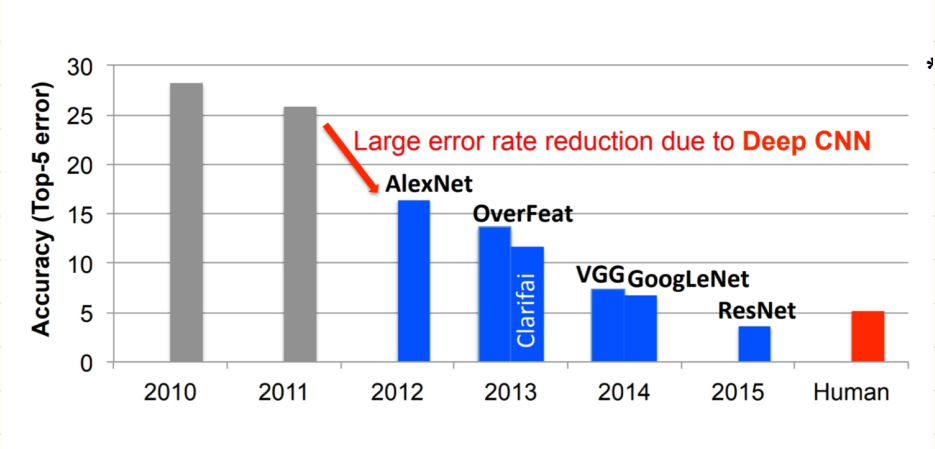
$\text{prox}_{\tau R}(z) = ?$

$$\left[\left(I + \tau \alpha \partial \|\cdot\|_1 \right)^{-1} (z) \right]_j = \begin{cases} z_j - \tau\alpha, & z_j > \tau\alpha \\ 0, & |z_j| \leq \tau\alpha \\ z_j + \tau\alpha, & z_j < -\tau\alpha \end{cases}$$

(Assignment)

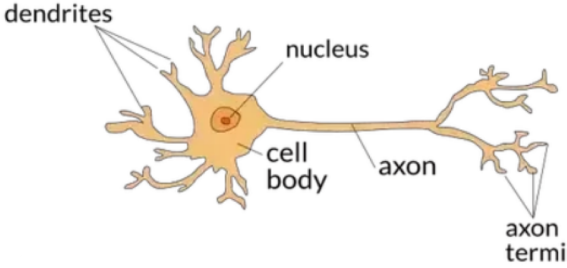
$$w_\alpha^{k+1} = \left(I + \tau \alpha \partial \|\cdot\|_1 \right)^{-1} \left[\left(I - \tau X^T X \right) w^k + \tau X^T y \right]$$

Section 6 Deep Learning

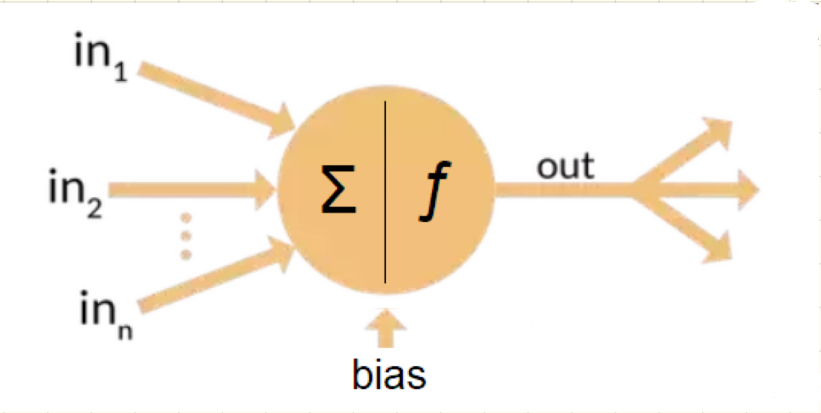


Speech and image recognition, image captioning, natural language processing

Human neural network



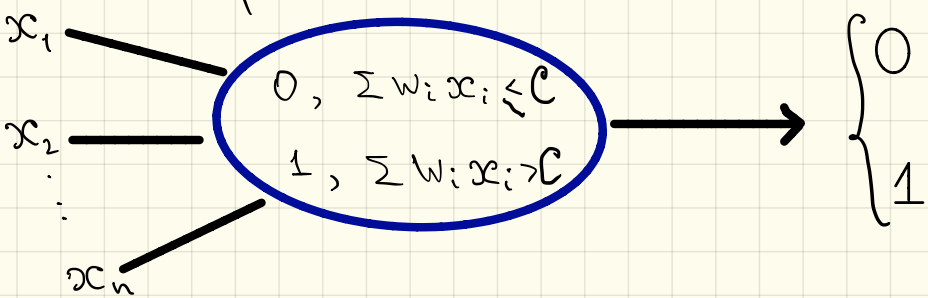
McCulloch, Pitts, Rosenblatt, 1950s - 60s



Artificial neural network

Perceptron

Example



x_1 Is the weather good
x_2 Will the partner join
x_3 Is there a public transport connection

$x_i \in \{0, 1\}$

$w = (6, 3, 2)^T$

$C = 5$

Outcome:
1. The weather is good -> going
2. The weather is bad -> not going

In the previous sections we considered simple one step optimisation problems. In deep learning one deals with the objective function being defined as a convolution of several simple functions (called activation functions)

$$f_w(x) = \varphi_L\left(\varphi_{L-1}\left(\dots \varphi_1\left(\varphi_0(x, w_0), w_1\right)\dots, w_{L-1}\right), w_L\right)$$

here $\{\varphi_\ell\}_{\ell=1}^{L}$ are activation functions that are parametrised with weights $w := \{w_\ell\}_{\ell=1}^{L}$.

Def. The above is called a deep neural network with L layers.

What are the typical activation functions?

1. Affine-linear transformations

$$\varphi(x; W, b) = W^T x + b$$

(W^T: weight matrix; b: bias vector)

$$x \in \mathbb{R}^d, \quad W \in \mathbb{R}^{d \times m}, \quad b \in \mathbb{R}^m$$

2. Heaviside function

$$H(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}, \quad \varphi(x) = \left(H(x_1), \dots, H(x_d)\right)^T$$

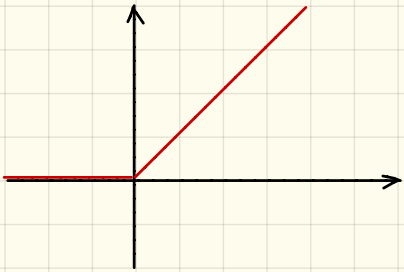
3. Perceptron

$$\varphi(x; W, b) = H(W^T x + b)$$

↑ a two layer neural network itself

4. Rectifier

$$\varphi(x) = (\max(0, x_1), \max(0, x_2), \dots, \max(0, x_d))^T$$



$$\max(0, x) = \text{prox}_{\chi_{[0,+\infty)}}(x)$$

Remark: all the above functions form Rectified Linear Unit (ReLU), and the general function from this set takes a form

$$\varphi(x; W, b) = \max\{0, W^T x + b\}.$$

(max: element-wise; $W^T x + b$: vectors in $\mathbb{R}^m$)

Typical ReLU neural network is:

$$\varphi(x; w, b) = \max(0, W_L^T \cdot (\max(0, W_{L-1}^T \cdot \dots \cdot \max(0, W_1^T x + b_1) \dots + b_{L-1}) + b_L)$$

$$\varphi(x; w, b) = \max(0, W_L^T x^{(L)} + b^{(L)})$$

$$x^{(\ell)} = \begin{cases} \max(0, W_\ell^T x^{(\ell-1)} + b^{(\ell)}), & 2 \le \ell \le L \\ \max(0, W_1^T x + b^{(1)}), & \ell = 1 \end{cases}$$

5. Softmax

$$\varphi(x_1, x_2, \dots, x_n) = \left(\frac{e^{x_1}}{\sum e^{x_i}}, \frac{e^{x_2}}{\sum e^{x_i}}, \dots, \frac{e^{x_n}}{\sum e^{x_i}} \right)$$

This can be thought as mapping vectors to probability distributions, indeed for

$$p_k = \frac{e^{x_k}}{\sum e^{x_i}}$$

one has $\sum_{k=1}^{n} p_k = 1$ and $p_k > 0$, $k = 1, \dots, n$.

Optimisation problem

$$\hat{w} = \arg\min_w \left\{ \frac{1}{s} \sum_{i=1}^{s} \ell(f_w(x_i), y_i) \right\}$$

Example: MSE

$$\hat{w} = \arg\min_w \frac{1}{s} \sum_{i=1}^{s} |x_i^{(L)} - y_i|^2, \text{ where}$$

$$\begin{cases} x_i^{(\ell)} = \varphi_\ell(x_i^{(\ell-1)}, w^{(\ell)}), & \ell = 1, \dots, L \\ x_i^{(0)} = x_i \end{cases}$$

Training of deep learning models

Main problem: objective function is not convex. But we will still apply our tools with a hope for convergence. We can use the gradient descent method in the form of

$$L(W_1, \dots, W_L, b_1, \dots, b_L) = \frac{1}{S} \sum_{i=1}^{S} \ell(x_i^{(L)}, y_i)$$

where

$$x_i^{(\ell)} = \sigma(z_i^{(\ell)})$$

$$z_i^{(\ell)} = W_\ell^T x_i^{(\ell-1)} + b_\ell \quad , \ell = 1, \dots, L$$

$$x_i^{(0)} = x_i$$

We want to apply the gradient descent and thus need to find the gradient of L with respect to all parameters $w_{jk}^{(\ell)}$, $b_j^{(\ell)}$

Example: $L = 2 \qquad d = m_1 = m_2 = 1$

$$L = \frac{1}{2S} \sum_{i=1}^{S} \left| \sigma\left(W^{(2)} \left[\sigma\left(W^{(1)} x_i + b^{(1)} \right) \right] + b^{(2)} \right) - y_i \right|^2$$

$z_i^{(2)}$, $z_i^{(1)}$, $x_i^{(1)}$, $x_i^{(2)}$

$S = 1$

$$\frac{\partial L}{\partial W^{(2)}} = \frac{\partial L}{\partial x^{(2)}} \cdot \frac{\partial x^{(2)}}{\partial W^{(2)}} = \frac{\partial L}{\partial x^{(2)}} \cdot x^{(1)}$$

$$\frac{\partial L}{\partial x^{(2)}} = (x^{(2)} - y)$$

$$\frac{\partial L}{\partial W^{(1)}} = \frac{\partial L}{\partial x^{(1)}} \cdot \frac{\partial x^{(1)}}{\partial W^{(1)}} = \frac{\partial L}{\partial x^{(1)}} \cdot x \quad ; \quad \frac{\partial L}{\partial x^{(1)}} = \frac{\partial L}{\partial x^{(2)}} \cdot \frac{\partial x^{(2)}}{\partial x^{(1)}}$$

$$= \frac{\partial L}{\partial x^{(2)}} \cdot \sigma'(z^{(2)}) \cdot W^{(2)}$$

Theorem

Let $f_w(x)$ be a neural network defined as: $f_w(x) = x^{(L)}$

$$x^{(\ell)} = \begin{cases} x, & \ell = 0 \\ \sigma(z^{(\ell)}), & \ell = 1, \dots, L \end{cases} \qquad z^{(\ell)} = W_\ell^T x^{(\ell-1)} + b^{(\ell)}, \quad \ell = 1, \dots, L$$

Let also the empirical risk function takes the form

$$L(W_1, \dots, W_L, b^{(1)}, \dots, b^{(L)}) = \frac{1}{S} \sum_{i=1}^{S} \ell(f_w(x_i), y_i)$$

Then the gradient of the empirical risk function with respect to parameters can be evaluated by a backpropagation scheme:

$$\delta_j^{(L)} := \frac{\partial L}{\partial x_j^{(L)}} = \frac{1}{S} \ell_x'(x^{(L)}, y_i)$$

$$\delta_j^{(\ell)} := \frac{\partial L}{\partial x_j^{(\ell)}} = \sum_{k=1}^{d} \frac{\partial L}{\partial x_k^{(\ell+1)}} \cdot \sigma'(z^{(\ell)}) \cdot (W_{\ell+1})_{jk}$$

$$= \sigma'(z^{(\ell)}) \cdot \left[W_{\ell+1} \delta^{(\ell+1)} \right]$$

$$\frac{\partial L}{\partial b_j^{(\ell)}} = \delta_j^{(\ell)} \qquad \frac{\partial L}{\partial w_{jk}^{(\ell)}} = \delta_j^{(\ell)} x_k^{(\ell-1)}$$

for $S = 1$