7.2 Loss View

Machine Learning 1: Foundations

Marius Kloft *(TUK)*

Recall:

SVM:

$$\min_{b\in\mathbb{R},\mathbf{w}\in\mathbb{R}^d} \ \frac{1}{2}\|\mathbf{w}\|^2 \ + \ C\sum_{i=1}^{n}\max\Big(0, 1 - y_i(\langle \mathbf{w}, \underset{\substack{\uparrow \\ \text{kernel feature map of a kernel } k}}{\phi_k(\mathbf{x}_i)}\rangle + b)\Big)$$

For simplicity, we had introduced ANNs without bias *b* in the ANN class. Here we use a bias *b*, which makes a lot sense, for the same reasons as it makes sense also in SVMs and LR.

Recall:

SVM:

$$\min_{b\in\mathbb{R},\mathbf{w}\in\mathbb{R}^d} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \max\Big(0, 1 - y_i(\langle \mathbf{w}, \phi_k(\mathbf{x}_i)\rangle + b)\Big)$$

kernel feature map of a kernel k

Logistic Regression (LR):

$$\min_{b\in\mathbb{R},\mathbf{w}\in\mathbb{R}^d} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \ln\Big(1 + \exp(-y_i(\langle \mathbf{w}, \phi_k(\mathbf{x}_i)\rangle + b))\Big)$$

For simplicity, we had introduced ANNs without bias b in the ANN class. Here we use a bias b, which makes a lot sense, for the same reasons as it makes sense also in SVMs and LR.

Recall:

SVM:

$$\min_{b\in\mathbb{R},\mathbf{w}\in\mathbb{R}^d} \quad \frac{1}{2}\|\mathbf{w}\|^2 \;+\; C\sum_{i=1}^{n} \max\Big(0, 1 - y_i(\langle \mathbf{w}, \underset{\substack{\uparrow\\ \text{kernel feature map of a kernel } k}}{\phi_k}(\mathbf{x}_i)\rangle + b)\Big)$$

Logistic Regression (LR):

$$\min_{b\in\mathbb{R},\mathbf{w}\in\mathbb{R}^d} \quad \frac{1}{2}\|\mathbf{w}\|^2 \;+\; C\sum_{i=1}^{n} \ln\Big(1 + \exp(-y_i(\langle \mathbf{w}, \phi_k(\mathbf{x}_i)\rangle + b))\Big)$$

ANN:

$$\min_{b,\mathbf{w},W} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{2}\sum_{l=1}^{L}\|W_l\|_{\text{Fro}}^2 + C\sum_{i=1}^{n}\ln(1+\exp(-y_i(\mathbf{w}^\top\phi_W(\mathbf{x}_i)+b)))$$

For simplicity, we had introduced ANNs without bias b in the ANN class. Here we use a bias b, which makes a lot sense, for the same reasons as it makes sense also in SVMs and LR.

Recall:

SVM:

$$\min_{b\in\mathbb{R},\mathbf{w}\in\mathbb{R}^d} \;\; \frac{1}{2}\|\mathbf{w}\|^2 \;+\; C\sum_{i=1}^{n}\max\Big(0, 1 - y_i(\langle \mathbf{w}, \underbrace{\phi_k}_{\text{kernel feature map of a kernel } k}(\mathbf{x}_i)\rangle + b)\Big)$$

Logistic Regression (LR):

$$\min_{b\in\mathbb{R},\mathbf{w}\in\mathbb{R}^d} \;\; \frac{1}{2}\|\mathbf{w}\|^2 \;+\; C\sum_{i=1}^{n}\ln\Big(1 + \exp(-y_i(\langle \mathbf{w}, \phi_k(\mathbf{x}_i)\rangle + b))\Big)$$

ANN:

$$\min_{b,\mathbf{w},W} \;\; \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{2}\sum_{l=1}^{L}\|W_l\|_{\text{Fro}}^2 + C\sum_{i=1}^{n}\ln(1+\exp(-y_i(\mathbf{w}^\top\phi_W(\mathbf{x}_i)+b)))$$

All these methods can be unified into a single equation.

For simplicity, we had introduced ANNs without bias b in the ANN class. Here we use a bias b, which makes a lot sense, for the same reasons as it makes sense also in SVMs and LR.

Unifying View

Unifying formulation of linear, kernel, and neural learning

$$\min_{[W_\cdot]\ b,\mathbf{w}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \ell(y_i(\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b)) \quad \left[+ \frac{1}{2}\sum_{l=1}^{L} \|W_l\|_{\text{Fro}}^2 \right],$$

where ℓ is:

- $\ell(t) := \max(0, 1-t)$ for SVM
- $\ell(t) := \ln(1+\exp(-t))$ for LR and ANN

ϕ is:

- $\phi = \mathrm{id}$ for <u>linear</u> LR and SVM
- $\phi = \phi_k$ for <u>kernel</u> LR and SVM
- $\phi = \phi_W$ for ANN

Quantities in brackets optional ← – only ANN !!

Five Popular Learning Machines in One Equation

The following table summarizes the result of the previous slide:

loss ℓ \ map ϕ	id	ϕ_k	ϕ_W
hinge	linear SVM	kernel SVM	—[1]
logistic	linear LR	kernel LR	ANN

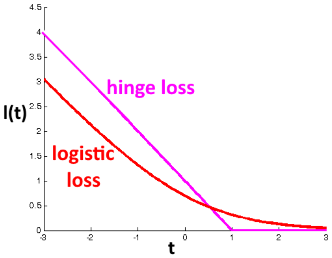
[1] The hinge loss is theoretically possible but uncommon in neural networks.

The Loss

The unifying equation contains, for every training example $(\mathbf{x}_i, y_i)$, a term

$$\ell(\underbrace{y_i \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b}_{=:t_i}),$$

the "loss" of the ith example.

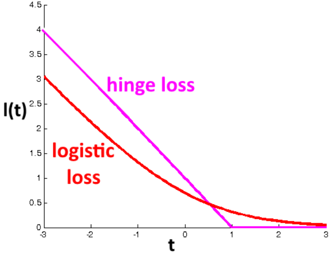


The Loss

The unifying equation contains, for every training example $(\mathbf{x}_i, y_i)$, a term

$$\ell(\underbrace{y_i \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b}_{=:t_i}),$$

the "loss" of the ith example.



Interpretation?

→ $\min \sum_i \ell(t_i)$

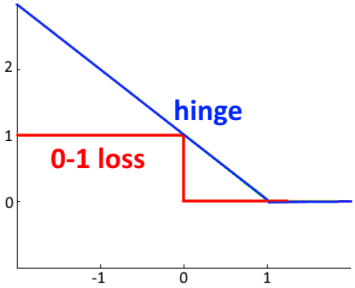
⇝ promotes small t_i

→ makes sense! $t_i > 0$ means, we classify ith example correctly

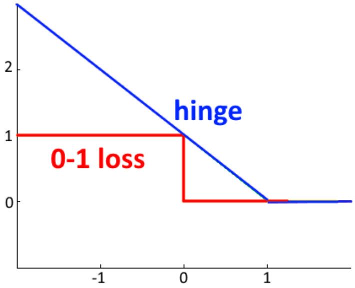
Understanding the Loss

Definition

The function $\ell(t) := \operatorname{sign}(-t)$ is called **0-1 loss**.



Understanding the Loss



Definition

The function $\ell(t) := \text{sign}(-t)$ is called **0-1 loss**.

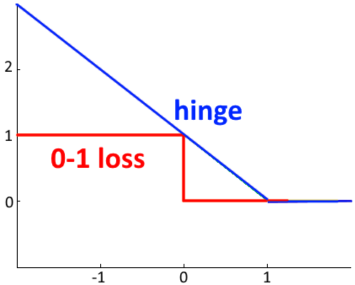
Observations:

- The 0-1 loss is one when the label is misclassified ($t < 0$) and zero otherwise

Understanding the Loss

Definition

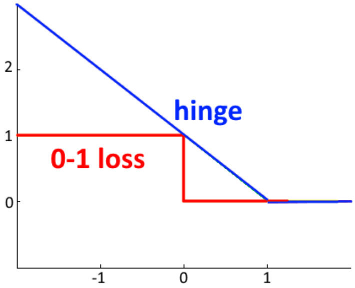
The function $\ell(t) := \text{sign}(-t)$ is called **0-1 loss**.



Observations:

- The 0-1 loss is one when the label is misclassified ($t < 0$) and zero otherwise
- Thus the cumulative 0-1 loss $\sum_{i=1}^{n} \ell\left(y_i\left(\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b\right)\right)$ measures the number of training errors

Understanding the Loss



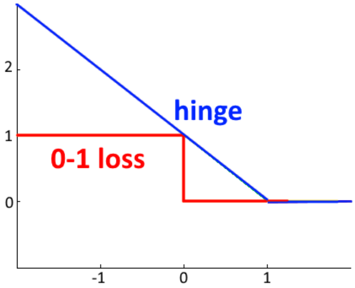
Definition

The function $\ell(t) := \text{sign}(-t)$ is called **0-1 loss**.

Observations:

- The 0-1 loss is one when the label is misclassified ($t < 0$) and zero otherwise
- Thus the cumulative 0-1 loss $\sum_{i=1}^{n} \ell\left(y_i\left(\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b\right)\right)$ measures the number of training errors
- Makes sense to get this error as small as possible

Understanding the Loss



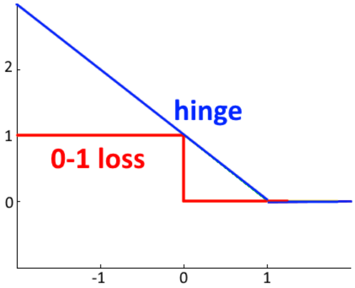
Definition

The function $\ell(t) := \text{sign}(-t)$ is called **0-1 loss**.

Observations:

- The 0-1 loss is one when the label is misclassified ($t < 0$) and zero otherwise
- Thus the cumulative 0-1 loss $\sum_{i=1}^{n} \ell\left(y_i\left(\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b\right)\right)$ measures the number of training errors
- Makes sense to get this error as small as possible
- Turns out to be an NP hard problem! :(

Understanding the Loss



Definition

The function $\ell(t) := \text{sign}(-t)$ is called **0-1 loss**.

Observations:

- The 0-1 loss is one when the label is misclassified ($t < 0$) and zero otherwise
- Thus the cumulative 0-1 loss $\sum_{i=1}^{n} \ell\left(y_i\left(\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b\right)\right)$ measures the number of training errors
- Makes sense to get this error as small as possible
- Turns out to be an NP hard problem! :(

Idea in SVM, LR, and ANN: replace the difficult 0-1 loss by a convex approximation—the hinge or logistic loss