Supervised Learning Basics
SC4/SM4 Data Mining and Machine Learning, Hilary Term 2017
Dino Sejdinovic

5.1 Loss and Risk

In the supervised learning framework, we are trying to learn a function $f : \mathcal{X} \to \mathcal{Y}$ from an input space $\mathcal{X}$ into an output space $\mathcal{Y}$ based on a set of paired examples $(x_1, y_1), \ldots (x_n, y_n)$ and a given *loss function* L. It is assumed that examples $(x_1, y_1), \ldots (x_n, y_n)$ are i.i.d. samples from an *unknown* joint probability distribution $P_{X,Y}$ on $\mathcal{X} \times \mathcal{Y}$ and the goal of learning is to find the function f which minimizes the expectation of the loss over $P_{X,Y}$ - called *risk*.

Empirical Risk Minimisation (ERM)
Loss is any function

$$L : \mathcal{Y} \times \mathcal{Y} \times \mathcal{X} \to \mathbb{R}^+. \tag{5.1}$$

Risk of a function $f : \mathcal{X} \to \mathcal{Y}$ is the expected loss:

$$R(f) = \mathbb{E}_{X,Y} L(Y, f(X), X). \tag{5.2}$$

For a given dataset $(x_1, y_1), \ldots (x_n, y_n)$, the *empirical risk* of f is given by

$$\hat{R}(f) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i), x_i). \tag{5.3}$$

The *Empirical Risk Minimisation* is the problem

$$\hat{f} = \operatorname*{argmin}_{f \in \mathcal{H}} \hat{R}(f),$$

where $\mathcal{H}$ is the given class of functions (hypothesis class).

Remark 5.1. The goal of learning is to minimise the true risk - *not* the empirical risk, which is only an estimate of the true risk. But the true risk of any given function is unknown because the distribution $P_{X,Y}$ is unknown.

Remark 5.2. Loss function typically depend on the input x only through $f(x)$, so that with some abuse of notation we often write $L(y, f(x))$ instead of $L(y, f(x), x)$. $L(y, f(x))$ is usually some notion of distance between the actual output y and the predicted output $f(x)$.

Examples of hypothesis classes. Hypothesis classes can be very simple, e.g. for $\mathcal{X} = \mathbb{R}^p$, we can consider all linear functions $f(x) = w^\top x + b$, parametrized by $w \in \mathbb{R}^p$ and $b \in \mathbb{R}$, or we could consider a specific *nonlinear feature expansion* $\varphi : \mathcal{X} \to \mathbb{R}^D$, and a model linear in those features: $f(x) = w^\top \varphi(x) + b$, but nonlinear in the original inputs $\mathcal{X}$, parametrized by $w \in \mathbb{R}^D$ and $b \in \mathbb{R}$. For example, starting with $\mathcal{X} = \mathbb{R}^2$, we can consider $\varphi\left(\begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}\right) = [x_{i1}, x_{i2}, x_{i1}^2, \sqrt{2} x_{i1} x_{i2}, x_{i2}^2]^\top$, such that the resulting function can depend on quadratic and interaction terms as well. An important type of hypothesis classes we will consider in the next lecture are *Reproducing Kernel Hilbert Spaces (RKHS)*, which are also linear in certain feature expansions but those feature expansions could potentially be infinite-dimensional.

Examples of loss functions. Loss functions come in many different forms. One of the main considerations for selecting loss functions is the type of outputs we are trying to predict, i.e., whether it is real-valued or categorical. Note that even if outputs are categorical, $f(x)$ is typically real-valued. For example, in binary classification, the common convention is that the two classes are denoted by -1 and $+1$. One associates predictions of these classes with $\text{sign}(f(x))$, whereas the magnitude of $f(x)$ can be thought of as the confidence in those predictions (not necessarily in a probabilistic sense). The loss can penalize misclassification (wrong sign) as well as the overconfident misclassification (wrong sign and large magnitude) and even underconfident correct classification (correct sign but small magnitude). Thus, they can be often expressed as a function of $yf(x)$.

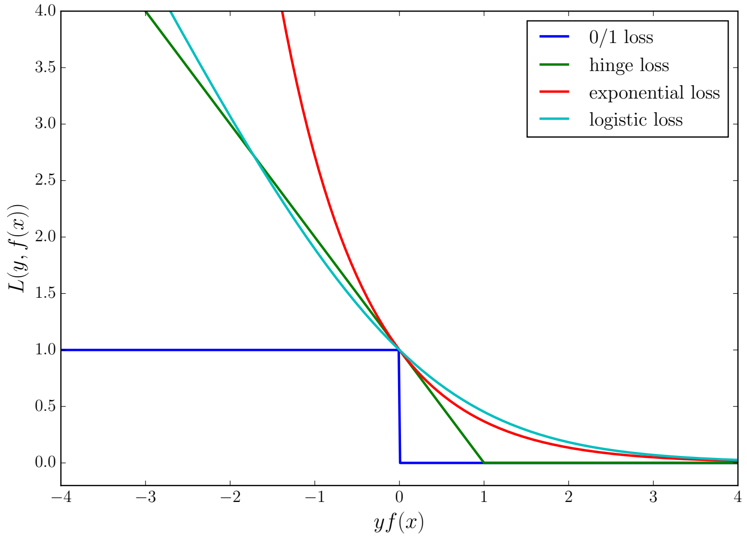

Figure 1: Loss functions for binary classification

Below are some loss functions commonly used in binary classification and regression.

- Binary classification:
  - 0/1 loss $L(y, f(x)) = \mathbf{1}\{yf(x) \leq 0\}$, (also called misclassification loss, optimal solution is called the *Bayes classifier* and is given by $f(x) = \text{argmax}_{k\in\{0,1\}} \mathbb{P}(Y = k|X = x)$),
  - hinge loss $L(y, f(x)) = (1 - yf(x))_+$ (used in *support vector machines* - leads to sparse solutions),
  - exponential loss $L(y, f(x)) = e^{-yf(x)}$ (used in *boosting* algorithms - Adaboost),
  - logistic loss $L(y, f(x)) = \log\left(1 + e^{-yf(x)}\right)$ (used in *logistic regression*, associated with a probabilistic model).

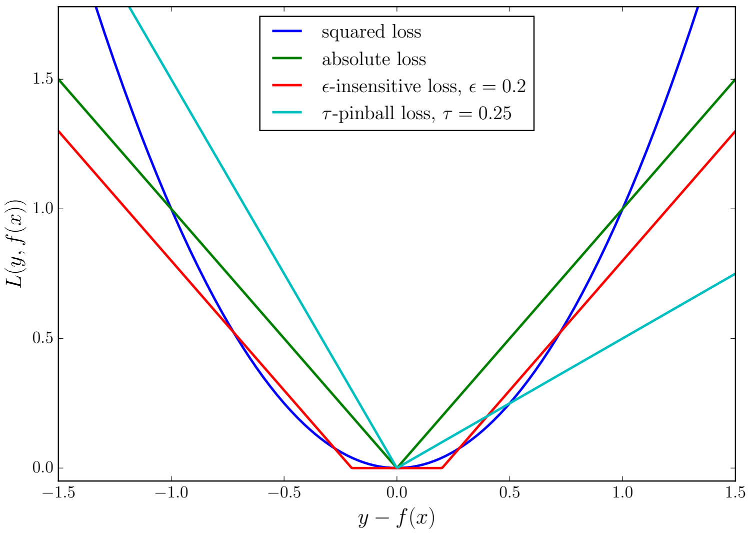

Figure 2: Loss functions for regression

- Regression:
  - squared loss: $L(y, f(x)) = (y - f(x))^2$ (least squares regression: optimal f is the conditional mean $\mathbb{E}[Y|X = x]$),
  - absolute loss: $L(y, f(x)) = |y - f(x)|$ (less sensitive to outliers: optimal f is the conditional median $\text{med}[Y|X = x]$),
  - τ-pinball loss: $L(y, f(x)) = 2 \max\{\tau(y - f(x)), (\tau - 1)(y - f(x))\}$ for $\tau \in (0, 1)$ (quantile regression: optimal f is the τ-quantile of $p(y|X = x)$),
  - ϵ-insensitive (Vapnik) loss: $L(y, f(x)) = \begin{cases} 0, \text{ if } |y - f(x)| \leq \epsilon, \\ |y - f(x)| - \epsilon, \text{ otherwise.} \end{cases}$ (*support vector regression* - leads to sparse solutions).

In binary classification, 0/1 is an idealised version of loss which penalizes misclassification regardless of the magnitude of $f(x)$. However, ERM under 0/1 loss is NP hard[1]. Therefore, we typically use *convex upper bound surrogate losses* (hinge, exponential, logistic[2]). What is the importance of the convexity of loss as a function of $yf(x)$ as shown in Fig. 1? Consider the hypothesis class $f(x) = w^\top \varphi(x)$, with $w \in \mathbb{R}^D$ (we ignore the intercept to simplify notation) and assume that $L(y, f(x)) = \rho(yf(x))$ for a convex differentiable function ρ. Then the empirical risk and its gradient are given by

$$\hat{R}(w) = \frac{1}{n} \sum_{i=1}^{n} \rho\left(y_i w^\top \varphi(x_i)\right), \quad \frac{\partial \hat{R}}{\partial w} = \frac{1}{n} \sum_{i=1}^{n} \rho'\left(y_i w^\top \varphi(x_i)\right) y_i \varphi(x_i).$$

[1] It is NP-hard to even approximately minimize the ERM under 0/1 loss - i.e. there is no known polynomial-time algorithm to obtain a solution which is a small constant worse than the optimum.

[2] to make it into an upper bound on 0/1, divide the logistic loss by log(2) - rescaling of the loss does not change the ERM problem

Furthermore, the Hessian matrix of the empirical risk is given by

$$\frac{\partial^2 \hat{R}}{\partial w \partial w^\top} = \frac{1}{n}\sum_{i=1}^{n} \rho''\left(y_i w^\top \varphi(x_i)\right) \varphi(x_i)\varphi(x_i)^\top, \tag{5.4}$$

using $y_i^2 = 1$. This Hessian is now a positive semidefinite matrix which can be seen from $\rho''(t) \geq 0$ $\forall t$ and

$$\alpha^\top \frac{\partial^2 \hat{R}}{\partial w \partial w^\top}\alpha \quad = \quad \frac{1}{n}\sum_{i=1}^{n} \rho''\left(y_i w^\top \varphi(x_i)\right)\left(\alpha^\top \varphi(x_i)\right)^2 \geq 0.$$

for any $\alpha \in \mathbb{R}^D$. Thus, empirical risk is a convex function of w and thus has a *unique minimum.* Typically, there is no closed form solution for w and iterative optimisation techniques like *gradient ascent* or *Newton-Raphson algorithm* are used.

5.2 Regularisation

Recall that we are not after the exact minimizer of the empirical risk but after that of the true risk. ERM risks *overfitting*, when the hypothesis class is complex, one can easily find a function that matches the observed examples exactly but does not *generalise* to new examples.

The idea behind *regularisation* is to limit the flexibility of hypothesis class in order to prevent overfitting. For the hypothesis space $\mathcal{H} = \{f_\theta : \theta \in \Theta\}$, this is achieved by adding the term which *penalises the large values of parameters* θ to the ERM criterion:

$$\min_\theta \hat{R}(f_\theta) + \lambda\|\theta\|_\rho^\rho = \min_\theta \frac{1}{n}\sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \lambda\|\theta\|_\rho^\rho$$

where $\rho \geq 1$, and $\|\theta\|_\rho = (\sum_{j=1}^p |\theta_j|^\rho)^{1/\rho}$ is the L_ρ norm of θ (also of interest when $\rho \in [0,1)$, but this is no longer a norm). These methods are also known as *shrinkage* methods since their effect is to shrinking parameters towards 0. Note that we have an additional *tuning parameter* (or *hyperparameter*) λ which controls the amount of regularisation, and resulting complexity of the model.

The most common forms of regularisation include *Ridge regression / Tikhonov regularization*: $\rho = 2$, *LASSO* penalty: $\rho = 1$, and *elastic net* regularization with a mixed L_1/L_2 penalty:

$$\min_\theta \frac{1}{n}\sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \lambda\left[(1-\alpha)\|\theta\|_2^2 + \alpha\|\theta\|_1\right].$$

In some hypothesis classes, it is possible to directly penalise some notion of *smoothness* of function f, e.g. for $\mathcal{X} = \mathbb{R}$, the regularisation term can consist of the *Sobolev norm*

$$\|f\|_{W^1}^2 = \int_{-\infty}^{+\infty} f(x)^2 dx + \int_{-\infty}^{+\infty} f'(x)^2 dx, \tag{5.5}$$

which penalises functions with large derivative values.

5.3 Examples of ERM

5.3.1 Regularised Least Squares / Ridge Regression

This corresponds to the squared loss $L(y, f(x)) = (y - f(x))^2$. For linear functions $f(x) = w^\top x + b$, we have

$$\min_{w,b} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^\top x_i - b)^2 + \frac{\lambda}{n} \|w\|_2^2. \tag{5.6}$$

Note the rescaling of the regularisation term and that the bias term b is not included in the regularisation. This is important as otherwise the predictions would depend on the origin for the response variables y (i.e. adding a constant c to each target would result in different predictions from simply shifting the original predictions by c). Fortunately, when using centred inputs, i.e., $\sum_{i=1}^{n} x_i = 0$, b can be estimated by $\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$, so we can also assume that the responses are centred and remove the intercept from the model. We obtain the problem

$$\min_{w} \|\mathbf{y} - \mathbf{X}w\|_2^2 + \lambda \|w\|_2^2. \tag{5.7}$$

Differentiating and setting to zero gives the closed form solution

$$w = \left(\mathbf{X}^\top \mathbf{X} + \lambda I\right)^{-1} \mathbf{X}^\top \mathbf{y}. \tag{5.8}$$

5.3.2 Support Vector Machines

Support Vector Machines (SVMs) for classification use hinge loss, $L(y, f(x)) = \max\{0, 1 - yf(x)\}$. Thus, for a linear classifier $f(x) = w^\top x + b$, we obtain

$$\min_{w,b} \frac{1}{n} \sum_{i=1}^{n} \max\{0, 1 - y_i(w^\top x_i + b)\} + \frac{\lambda}{n} \|w\|_2^2. \tag{5.9}$$

This no longer has a closed form solution and requires numerical optimisation. Eq. (5.9) is not how you would typically see an SVM written in the literature. Rather, we introduce a substitution $\xi_i = \max\{0, 1 - y_i(w^\top x_i + b)\}$, which implies that $\xi_i \geq 0$, $y_i(w^\top x_i + b) \geq 1 - \xi_i$ and with a reparametrisation of the regularisation parameter $C = 1/2\lambda$ obtain the equivalent form, called C-SVM:

$$\begin{aligned} &\min_{w,b,\xi} \left(\frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i \right), \\ &\text{subject to} \quad \xi_i \geq 0, \qquad y_i \left(w^\top x_i + b \right) \geq 1 - \xi_i. \end{aligned} \tag{5.10}$$

SVMs have the following nice property: the normal vector w of the hyperplane determining the classification rule can be written as $w = \sum_{i=1}^{n} \alpha_i y_i x_i$ where a large number of α-coefficients is typically zero. Thus, only a small number of datapoints (*support vectors*, those with a non-zero α)

determine the learned classification rule. α-coefficients are called the *dual coefficients*. They can be obtained as a solution to the following dual C-SVM problem

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^\top x_j, \tag{5.11}$$
$$\text{subject to} \quad 0 \leq \alpha_i \leq C, \quad \sum_{i=1}^{n} y_i \alpha_i = 0.$$

5.3.3 Logistic Regression

Logistic regression uses the logistic loss $L(y, f(x)) = \log\left(1 + e^{-yf(x)}\right)$. Hence, again for a linear classifier $f(x) = w^\top x + b$,

$$\min_{w,b} \; \frac{1}{n} \sum_{i=1}^{n} \log\left(1 + e^{-y_i(w^\top x_i + b)}\right) + \frac{\lambda}{n} \|w\|_2^2. \tag{5.12}$$

Unlike SVMs, logistic regression can also be associated to the probabilistic model. Namely, assume that the function of interest $f(x) = w^\top x + b$ models the log-odds ratio:

$$\log \frac{p(y_i = +1|w, b, x_i)}{p(y_i = -1|w, b, x_i)} = w^\top x_i + b. \tag{5.13}$$

Then the conditional distribution of $Y|X$ is given by

$$p(y_i = +1|w, b, x_i) = \frac{1}{1 + e^{-(w^\top x_i + b)}} = \sigma(w^\top x_i + b), \tag{5.14}$$
$$p(y_i = -1|w, b, x_i) = \frac{1}{1 + e^{w^\top x_i + b}} = \sigma(-w^\top x_i - b), \tag{5.15}$$

where we denoted by $\sigma(t) = 1/(1 + e^{-t})$ the *logistic function* which maps the real line to $(0, 1)$ interval. Note that the logistic function satisfies $\sigma(-t) = 1 - \sigma(t)$. Thus, we can write (5.14) and (5.15) as $p(y_i|w, b, x_i) = \sigma(y_i(w^\top x_i + b))$ and the conditional log-likelihood of the outputs given the inputs is

$$\log p(\mathbf{y}|w, b, \mathbf{X}) = \log \prod_{i=1}^{n} \sigma(y_i(w^\top x_i + b)) = -\sum_{i=1}^{n} \log\left(1 + e^{-y_i(w^\top x_i + b)}\right).$$

Thus finding the parameters w and b that maximise the conditional log-likelihood is equivalent to minimising the empirical risk corresponding to the logistic loss, which is the negative log-likelihood of the linear log-odds model. Moreover, the regularisation term can be interpreted as a normal prior on w in *Bayesian logistic regression*. Again, there is no closed form solution in logistic regression, but the objective is convex and differentiable and the optimisation using gradient ascent or Newton-Raphson algorithm can be used.

The connection between maximisation of the log-likelihood and minimisation of the empirical risk extends beyond logistic regression. Indeed, in the context of classification, whenever $p(y_i|x_i, \theta)$ is a log-concave function of $y_i f_\theta(x_i)$, we can define a convex loss $\rho(y f_\theta(x)) = -\log p(y_i|x_i, \theta)$. But the converse is not true, e.g. hinge loss does not correspond to a negative log-likelihood in any probabilistic model (unless additional artificial classes are introduced).