Lecture 9

6 Risk and loss functions

In supervised learning, we are given training samples $\{(x_1, y_1), \ldots, (x_n, y_n)\} \subset \mathcal{X} \times \mathcal{Y}$. Our machine learning algorithm uses this data to construct a function $f : \mathcal{X} \to \mathcal{Y}$, which assigns a label $f(x)$ to any $x \in \mathcal{X}$. But the question then arises, how do assess the quality of f? That is the subject of this section.

6.1 Loss functions

Adapted from [1, Chapter 3.1]

We define a loss function. Denote by $(x, y, f(x)) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{Y}$ a triplet consisting of data point x, its label y, and the predicted label $f(x)$. A map $c : \mathcal{X} \times \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ with the property that $c(x, y, y) = 0$ is a *loss function*.

In what follows we give some examples of loss functions to keep in mind.

6.1.1 Examples of loss functions for binary classification

Suppose $\mathcal{Y} = \{-1, +1\}$.

For these first two examples, we assume that our machine learning algorithm constructs a function $f : \mathcal{X} \to \mathcal{Y}$.

- 0–1 loss:
$$c(x, y, f(x)) = \begin{cases} 0 & \text{if } y = f(x), \\ 1 & \text{if } y \neq f(x). \end{cases}$$

- Input dependent loss:
$$c(x, y, f(x)) = \begin{cases} 0 & \text{if } y = f(x), \\ \tilde{c}(x) & \text{if } y \neq f(x). \end{cases}$$

Now suppose instead our machine learned algorithm outputs a function $f : \mathcal{X} \to \mathbb{R}$, which gives the confidence of our prediction. In this case, $\text{sgn} f(x)$ is the class label, and $|f(x)|$ is our confidence in the prediction. In this case some example loss functions are:

- Soft margin loss:

$$c(x, y, f(x)) = \max(0, 1 - yf(x)) = \begin{cases} 0 & \text{if } yf(x) \geq 1, \\ 1 - yf(x) & \text{if } yf(x) < 1. \end{cases}$$

- Quadratic soft margin loss:

$$c(x, y, f(x)) = \max(0, 1 - yf(x))^2$$

- Logistic loss:

$$c(x, y, f(x)) = \log(1 + \exp(-yf(x))).$$

Figure 11 illustrates some of these loss functions.

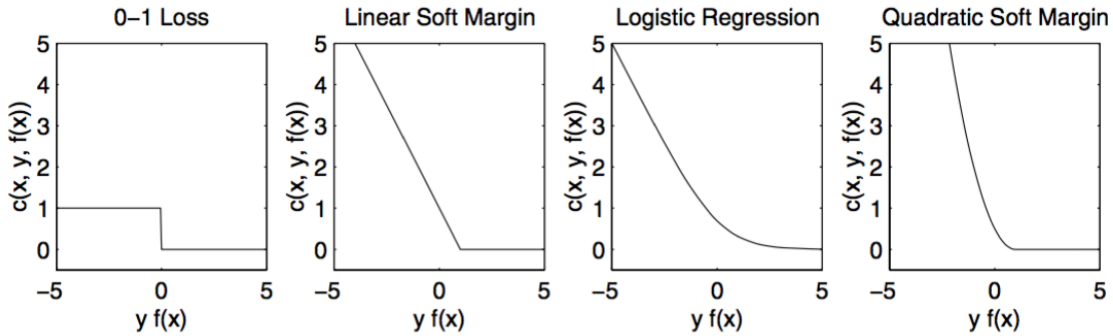

Figure 11: Plots of various loss functions. Note that both of the soft margin loss functions are upper bounds on the 0–1 loss function.

6.1.2 Examples of loss functions for regression

In regression tasks $\mathcal{Y}$ is a continuum of values, often $\mathcal{Y} \subseteq \mathbb{R}$. In this case the difference $y - f(x)$ is usually more important than the product $yf(x)$. Thus in most cases the loss function will be of the type:

$$c(x, y, f(x)) = \tilde{c}(y - f(x)).$$

Here are some examples:

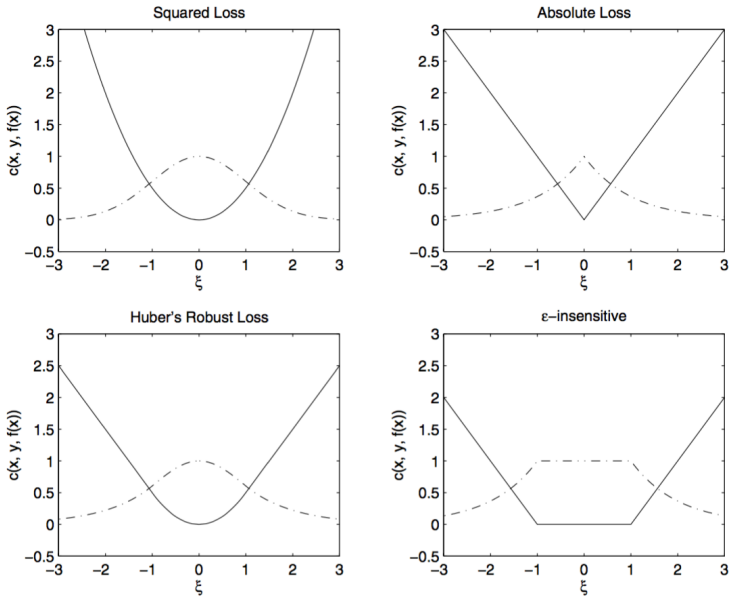

Figure 12: Graphs of regression loss functions in black. Ignore the dotted lines.

- Squared loss:

$$c(x, y, f(x)) = (y - f(x))^2, \quad \text{i.e.,} \quad \tilde{c}(\xi) = \xi^2.$$

- ε-sensitive loss:

$$c(x, y, f(x)) = \max(|y - f(x)| - \epsilon, 0) \quad \text{i.e.,} \quad \tilde{c}(\xi) = \max(|\xi| - \varepsilon, 0).$$

  When $\varepsilon = 0$ we obtain the absolute loss function.

See Figure 12 for plots of these loss functions.

6.2 Test error and empirical risk

Adapted from [1, Chapter 3.2]

We now combine the individual penalties $c(x, y, f(x))$ to assess the quality of f. We assume there exists a probability distribution $P(x, y)$ on $\mathcal{X} \times \mathcal{Y}$ which governs the data generation and underlying functional dependency.

We assume that the training data $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ are drawn i.i.d. from $P(x, y)$. We also assume that we have no knowledge of what

test patterns x we will be asked to classify or regress with $f(x)$ (if you know in advance the test points, it makes a difference).

Within this framework, we want to minimize the expected error over all possible test points, which is all of $\mathcal{X} \times \mathcal{Y}$. That is, we want to minimize the *risk* of f, also known as the *expected loss*, with respect to P and c:

$$R[f] = \mathbb{E}[c(x, y, f(x))] = \int_{\mathcal{X}\times\mathcal{Y}} c(x, y, f(x))\, dP(x, y).$$

However, minimizing $R[f]$ is impossible because we do not know P and we do not know all of $\mathcal{X} \times \mathcal{Y}$. All we are given is the training data $\{(x_i, y_i)\}_{i\leq n}$. These training points, though, allow us to compute an empirical estimate of $R[f]$, which we call the *empirical risk*:

$$R_{\text{emp}}[f] = \frac{1}{n}\sum_{i=1}^{n} c(x_i, y_i, f(x_i)).$$

With $R_{\text{emp}}[f]$, we now need to settle on a functional class $\mathcal{F}$ over which to minimize it:

$$\inf_{f\in\mathcal{F}} R_{\text{emp}}[f]. \tag{30}$$

However, determining $\mathcal{F}$ is rather difficult (we'll come back to this a bit later). Even with a functional class $\mathcal{F}$ that we may like, the optimization (30) can be ill posed.

To make this more concrete, let's look at the following example. Consider a regression task with the quadratic loss function $c(x, y, f(x)) = (y - f(x))^2$. Suppose that we have a feature map

$$\Phi(x) = (\phi_1(x), \ldots, \phi_m(x)) \in \mathbb{R}^m,$$

and we consider linear regressions over Φ:

$$\mathcal{F} = \left\{ f : \mathcal{X} \to \mathbb{R} : f(x) = \sum_{j=1}^{m} w_j \phi_j(x), \;\; w_j \in \mathbb{R} \right\}.$$

We want to find the minimizer of R_{emp}, i.e.,

$$\inf_{f\in\mathcal{F}} R_{\text{emp}}[f] = \inf_{\mathbf{w}\in\mathbb{R}^m} \frac{1}{n}\sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{m} w_j \phi_j(x_i) \right)^2,$$

where $\mathbf{w} = (w_j)_{j=1}^m$. Overloading notation slightly, we also let Φ stand for the $m \times n$ matrix with entries given by:

$$\Phi_{ji} = \phi_j(x_i) = \begin{pmatrix} \phi_1(x_1) & \phi_1(x_2) & \cdots & \phi_1(x_n) \\ \phi_2(x_1) & \phi_2(x_2) & \cdots & \phi_2(x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_m(x_1) & \phi_m(x_2) & \cdots & \phi_m(x_n) \end{pmatrix}$$

Additionally, set:

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n \quad \text{and} \quad \mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{pmatrix} \in \mathbb{R}^m,$$

as well as:

$$F(\mathbf{w}) = F(w_1, \ldots, w_m) = \sum_{i=1}^n \left(y_i - \sum_{j=1}^m w_j \phi_j(x_i) \right)^2.$$

Computing the partial derivatives of F we obtain:

$$\begin{aligned} \frac{\partial F}{\partial w_k}(w) &= \sum_{i=1}^n 2 \left(y_i - \sum_{j=1}^m w_j \phi_j(x_i) \right) \phi_k(x_i), \\ &= 2 \sum_{i=1}^n \phi_k(x_i) y_i - 2 \sum_{i=1}^n \phi_k(x_i) \sum_{j=1}^m w_j \phi_j(x_i), \\ &= 2\Phi\mathbf{y}[k] - 2 \sum_{i=1}^n \phi_k(x_i) \Phi^T \mathbf{w}[i], \\ &= 2\Phi\mathbf{y}[k] - 2\Phi\Phi^T \mathbf{w}[k]. \end{aligned} \tag{31}$$

Noting that the minimizer $\mathbf{w}^\star$ of F occurs when $\partial F / \partial w_k(\mathbf{w}^\star) = 0$, using (31) we obtain:

$$\Phi\mathbf{y} = \Phi\Phi^T \mathbf{w}^\star.$$

It thus follows that:

$$\mathbf{w}^\star = \left(\Phi\Phi^T\right)^{-1} \Phi\mathbf{y},$$

where $(\Phi\Phi^T)^{-1}$ denotes the inverse of the matrix $\Phi\Phi^T$ if $\Phi\Phi^T$ has full rank, and the pseudo-inverse otherwise.

If $\Phi\Phi^T$ is full rank but its condition number is large, then it will be numerically very difficult to compute $\mathbf{w}$. On the other hand, if $m > n$ (i.e. the dimension of our feature space is larger than the size of the training set), then $\Phi\Phi^T$ will not be full rank and there will be a linear subspace of solutions of dimension at least $m - n$, and thus the solution will not be unique. Even if $n = m$, and there is an f such that $f(x_i) = y_i$ for all $i = 1, \ldots, n$, this does not guarantee that f will generalize well to data points x not in the training set. We'll discuss how to account of this problem in the next section.

Exercises

Exercise 19. Do the kernel version of the previous calculation. In other words, consider the class of regression functions:

$$\mathcal{F} = \left\{ f : \mathcal{X} \to \mathbb{R} : f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x) \right\}.$$

Again we want to minimize the empirical risk for the quadratic loss function, which is given by:

$$\inf_{f \in \mathcal{F}} R_{\text{emp}}[f] = \inf_{\alpha \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{n} \alpha_i k(x_i, x_j) \right)^2. \tag{32}$$

Show that if k is symmetric and the Gram matrix of k with respect to the training data $\{x_1, \ldots, x_n\}$ is invertible, the optimal $\alpha^\star$ that minimizes (32) is given by:

$$\alpha^\star = K^{-1}\mathbf{y}, \tag{33}$$

where $K_{ij} = k(x_i, x_j)$.

Exercise 20. Let's return to the MNIST data set. Consider again a binary classification problem between pairs of numbers, e.g, 1 and 4, 3 and 5, etc. Take the labels to be $\mathcal{Y} = \{+1, -1\}$, $+1$ for one digit, -1 for the other digit. We can use the kernel regression algorithm from the previous exercise to define a classifier. Indeed, a class label can be defined as:

$$y = \operatorname{sgn} f(x) = \operatorname{sgn} \left(\sum_{i=1}^{n} \alpha_i k(x_i, x) \right).$$

In this case, the magnitude $|f(x)|$ can be interpreted as a measure of confidence in the prediction - the larger the value, the more confident the prediction. Train such a regression with the quadratic polynomial $k(x, x') = (\langle x, x' \rangle + c)^2$ using (33). Compare your results to the linear classifier you developed previously. What is your interpretation? Note 1: Do this for all pairs of numbers, which will give you a better sense of the difference between the two classification algorithms. Note 2: You most likely will have to do a 50% training, 50% testing split (or something like this), because otherwise the training kernel K can get quite large in memory. Thus to make a fair comparison, you will have to rerun your linear classifier.

References

[1] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive Computation and Machine Learning. The MIT Press, 2002.

[2] Afonso S. Bandeira. Ten lectures and forty-two open problems in the mathematics of data science. MIT course *Topics in Mathematics of Data Science*, 2015.

[3] Jon Shlens. A tutorial on principal component analysis. arXiv:1404.1100, 2014.

[4] Karl Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine, Series 6*, 2(11):559–572, 1901.

[5] V. A. Marchenko and L. A. Pastur. Distribution of eigenvalues in certain sets of random matrices. *Mat. Sb. (N.S.)*, 72(114):507–536, 1967.

[6] J. Baik, G. Ben-Arous, and S. Péché. Phase transition of the largest eigenvalue for nonnull complex sample covariance matrices. *The Annals of Probability*, 33(5):1643–1697, 2005.

[7] Debashis Paul. Asymptotics of sample eigenstructure for a large dimensional spiked covariance model. *Statistica Sinica*, pages 1617–1642, 2007.