PH 240C Supervised Learning (2): Empirical Risk Minimization and Kernel Methods

Jingshen Wang

September 15, 2021

1 Lagrange duality

To solve the problem in Figure 1, we need to work with kernel SVM which is motivated by the dual problem of the optimization problem. Moreover, the dual problem will also allow us to derive an efficient algorithm to solve SVM better than generic QP software.

Review of Lagrange with equality constraints Instead of considering an inequality constraints in the optimization problem for SVM, consider a problem of a simpler form with equality constraint:

$$\begin{aligned} \min_{\mathrm{w}\in\mathbb{R}^d} \quad & f(\mathrm{w}) \\ \text{s.t.} \quad & h_i(\mathrm{w}) = 0, \quad i = 1, \ldots, n. \end{aligned} \tag{1}$$

We define the **Lagrangian** $L : \mathbb{R}^d \times \mathbb{R}^p \to \mathbb{R}$ associated with the problem (1) as

$$L(\mathrm{w}, \lambda) = \ f(\mathrm{w}) + \sum_{i=1}^{n} \lambda_i h_i(\mathrm{w}).$$

We refer to λ_i as the Lagrange multiplier associated with the ith equality constraint $h_i(\mathrm{w}) = 0$. The vector $\lambda = (\lambda_1, \ldots, \lambda_n)'$ is called the **dual variable** or the **Lagrange multiplier vector** associated with the problem (1). We would then find and set $L(\mathrm{w}, \lambda)$'s partial derivatives to zero:

$$\frac{\partial L(\mathrm{w}, \lambda)}{\partial \mathrm{w}_j} = 0, \quad \frac{\partial L(\mathrm{w}, \lambda)}{\partial \lambda_i} = 0, \quad j = 1, \ldots, d, \ i = 1, \ldots, n,$$

and solve for w and λ.

Lagrange with inequality and equality constraints–primal and dual problems Now we generalize this to constrained optimization problems in which we may have inequality as well as equality constraints:

$$\begin{aligned} \text{Primal problem:} \quad \min_{\mathrm{w}\in\mathbb{R}^d} \quad & f(\mathrm{w}) \\ \text{s.t.} \quad & g_i(\mathrm{w}) \le 0, \quad i = 1, \ldots, k, \\ & h_i(\mathrm{w}) = 0, \quad i = 1, \ldots, n. \end{aligned} \tag{2}$$

We refer to the above problem as the **primal problem**, and the Lagrangian associated with the primal problem is

$$L(\mathrm{w}, \lambda, \nu) = \ f(\mathrm{w}) + \sum_{i=1}^{k} \nu_i g_i(\mathrm{w}) + \sum_{i=1}^{n} \lambda_i h_i(\mathrm{w}).$$

Here, $\nu_i \geq 0$ and λ_i are the Lagrange multipliers. We can show that the solution of the primal problem (2) is also the solution to the following problem (see footnote for proof)[1]:

$$\min_{\mathrm{W}} L_{\texttt{primal}}(\mathrm{w}) = \min_{\mathrm{W}} \max_{\nu \succeq 0, \lambda} \ L(\mathrm{w}, \lambda, \nu).$$

We also define the optimal value of the objective function to be $p^* = \min_{\mathrm{W}} L_{\texttt{primal}}(\mathrm{w})$.

Next, we define the **Lagrange dual function** (or just dual function) as

$$L_{\texttt{dual}}(\nu, \lambda) = \min_{\mathrm{W}} \ L(\mathrm{w}, \lambda, \nu).$$

When the Lagrange is unbounded below in w, the dual function takes on the value $-\infty$. The **dual problem** is then defined as

$$\text{Dual problem:} \quad \max_{\nu \succeq 0, \lambda} \ L_{\texttt{dual}}(\mathrm{w}) = \max_{\nu \succeq 0, \lambda} \min_{\mathrm{W}} \ L(\mathrm{w}, \lambda, \nu).$$

This is exactly the same as our primal problem shown above, except that the order of the "max" and the "min" are now exchanged. We again define the optimal value of the dual problem's objective to be $d^* = \max_{\mathrm{W}} \ L_{\texttt{dual}}(\mathrm{w})$.

How are the primal and dual problems related? It is easy to see that

$$d^* = \max_{\mathrm{W}} \min_{\nu \succeq 0, \lambda} \ L(\mathrm{w}, \lambda, \nu) \leq \min_{\nu \succeq 0, \lambda} \max_{\mathrm{W}} \ L(\mathrm{w}, \lambda, \nu) = p^*.$$

Under certain conditions [2], we can show that the dual problem and the primal problem have no gap (no

[1]

Proof. To see this, consider the quantity

$$L_{\texttt{primal}}(\mathrm{w}) = \max_{\nu \succeq 0, \lambda} \ L(\mathrm{w}, \lambda, \nu).$$

Then, for a given w: (a) If w violates any of the primal constraints, then we can verify that

$$L_{\texttt{primal}}(\mathrm{w}) = \max_{\nu \succeq 0, \lambda} \left(\ f(\mathrm{w}) + \sum_{i=1}^{k} \nu_i \underbrace{g_i(\mathrm{w})}_{\leq 0} + \sum_{i=1}^{n} \lambda_i h_i(\mathrm{w}) \right) = +\infty.$$

Conversely, if the constraints are indeed satisfied for a particular value of w, then $L_{\texttt{primal}}(\mathrm{w}) = f(\mathrm{w})$. Hence, we conclude

$$L_{\texttt{primal}}(\mathrm{w}) = \begin{cases} f(\mathrm{w}) & \text{if w satisfies primal constraints} \\ +\infty & \text{otherwise.} \end{cases}$$

Thus, $L_{\texttt{primal}}(\mathrm{w})$ takes the same value as the objective function in the problem (2) for all w that satisfies the primal constraints, and is positive infinity if the constraints are violated. We conclude

$$\min_{\mathrm{W}} L_{\texttt{primal}}(\mathrm{w}) = \min_{\mathrm{W}} \max_{\nu \succeq 0, \lambda} \ L(\mathrm{w}, \lambda, \nu).$$

Therefore, the solution to the primal problem is also the solution to the unconstrained problem $\min_{\mathrm{W}} \max_{\nu \succeq 0, \lambda} \ L(\mathrm{w}, \lambda, \nu)$. □

[2] One set of sufficient conditions: Suppose f and g_i's are convex function, and h_i's are affine. Further suppose that the

duality gap): $d^* = p^*$. This suggests there must exists $(\mathrm{w}^*, \nu^*, \lambda^*)$ satisfies the *Karush-Kuhn-Tucker (KKT)* conditions, which are defined as

$$\begin{aligned}
\frac{\partial L(\mathrm{w}^*, \nu^*, \lambda^*)}{\partial \mathrm{w}_i} &= 0, \ i = 1, \ldots, p, \\
\frac{\partial L(\mathrm{w}^*, \nu^*, \lambda^*)}{\partial \lambda_i} &= 0, \ i = 1, \ldots, n \\
\nu_i g_i(\mathrm{w}) &= 0, \ i = 1, \ldots, k \\
g_i(\mathrm{w}) &\leq 0, \ i = 1, \ldots, k \\
\nu_i &\leq 0, \ i = 1, \ldots, k.
\end{aligned}$$

2 SVM-continued

2.1 Dual problem of SVM

Let's now circle back to the (primal) SVM optimization problem defined in last class:

$$\begin{aligned}
\min_{a, \mathrm{w}} \ & \frac{1}{2}\mathrm{w}'\mathrm{w} \\
\text{s.t.} \ & -Y_i(a + \mathrm{w}'X_i) + 1 \leq 0, \quad i = 1, \ldots, n.
\end{aligned} \tag{3}$$

Borrowing the notation from the previous section, the inequality constraints are

$$g_i(\mathrm{w}) = -Y_i(a + \mathrm{w}'X_i) + 1 \leq 0.$$

Since there is no equality constraint in the problem (3), the Lagrangian for our optimization problem is

$$L(\mathrm{w}, a, \nu) = \frac{1}{2}\mathrm{w}'\mathrm{w} - \sum_{i=1}^{n} \nu_i \big[Y_i(a + \mathrm{w}'X_i) - 1\big]. \tag{4}$$

By checking the conditions listed in the previous section, there is no duality gap between the primal problem (3) and its dual. To get the dual problem of (3), we following the steps introduced in the last section:

1. Find the Lagrange dual function associated with (3):

$$L_{\texttt{dual}}(\nu) = \min_{a, \mathrm{w}} L(a, \mathrm{w}, \nu).$$

   To to do, we take derivative with respect to w and a:

$$\frac{\partial L(a, \mathrm{w}, \nu)}{\partial \mathrm{w}} = \mathrm{w} - \sum_{i=1}^{n} \nu_i Y_i X_i = 0, \quad \Rightarrow \quad \mathrm{w} = \sum_{i=1}^{n} \nu_i Y_i X_i, \tag{5}$$

$$\frac{\partial L(a, \mathrm{w}, \nu)}{\partial a} = \sum_{i=1}^{n} \nu_i Y_i = 0, \quad \Rightarrow \quad \sum_{i=1}^{n} \nu_i Y_i = 0. \tag{6}$$

constraint g_i are (strictly) feasible–meaning that there exists some w such that $g_i(\mathrm{w}) < 0$ for all i. Note there are many different sets of sufficient conditions can be found in the literature.

2. We now take the definition of w in (5) and plug it back into the equation (4):

$$\begin{aligned} L_{\texttt{dual}}(\nu) = \min_{a,\mathrm{w}} L(a,\mathrm{w},\nu) &= \sum_{i=1}^{n}\nu_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} Y_iY_j\nu_i\nu_j X_i'X_j - a\sum_{i=1}^{n}\nu_iY_i \\ &= \sum_{i=1}^{n}\nu_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} Y_iY_j\nu_i\nu_j X_i'X_j. \end{aligned}$$

3. **SVM dual problem**: Dual problem associated with the primal problem (3) is thus

$$\begin{aligned} \max \;\; & L_{\texttt{dual}}(\nu) = \sum_{i=1}^{n}\nu_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} Y_iY_j\nu_i\nu_j X_i'X_j \\ \text{s.t.} \;\; & \nu_i \geq 0,\; i = 1,\ldots,n, \\ & \sum_{i=1}^{n}\nu_iY_i = 0. \end{aligned} \tag{7}$$

4. **Primal and dual relationship**: Suppose we can solve the dual problem (for now) and the optimal solution is ν^*, then we can obtain the primal solution from (5)

$$\mathrm{w}^* = \sum_{i=1}^{n}\nu_i^*Y_iX_i,$$

how about the intercept a^*?

We want to resolves several issues in SVM by working out the dual formulation:

First, what does $\nu_i^* = 0$ tell us? Recall from the KKT condition, we must have (this specific condition is also called KKT dual complementarity condition)

$$\nu_i\big[Y_i(a + \mathrm{w}'X_i) - 1\big] = 0.$$

It says that if $\nu_i^* > 0$, then $Y_i(a+\mathrm{w}'X_i)-1 = 0$, meaning that ith data point is a support vector. Whenever $\nu_i^* = 0$, we have the sample pair (Y_i, X_i) does not live on the decision boundary.

Second, when we make prediction at a new data point $x \in \mathbb{R}^d$, we would then calculate $\mathrm{w}^{*'}x + a^*$ and then predict

$$y = \begin{cases} 1 & \mathrm{w}^{*'}x + a^* \geq 0 \\ -1 & \mathrm{w}^{*'}x + a^* < 0. \end{cases}$$

Notice that the quantity $\mathrm{w}^{*'}x + a^*$ can be rewritten as

$$\begin{aligned} \mathrm{w}^{*'}x + a^* &= \left(\sum_{i=1}^{n}\nu_i^*Y_iX_i\right)'x + a^* \\ &= \sum_{i=1}^{n}\nu_i^*Y_iX_i'x + a^* \\ &= a^* + \sum_{i:\nu_i^*\neq 0}\mathbf{1}_{(Y_i=1)}X_i'x - \sum_{i:\nu_i^*\neq 0}\mathbf{1}_{(Y_i=-1)}X_i'x. \end{aligned}$$

Now we can see that decision made from SVM is also quite intuitive: if the new data point x is more "similar" to the support vectors labelled as one–hence there is a higher "chance" that $\sum_{i:\nu_i \neq 0} \mathbf{1}_{(Y_i=1)} X_i' x > \sum_{i:\nu_i \neq 0} \mathbf{1}_{(Y_i=-1)} X_i' x$– thereby it classifies x as 1. Can this motivate you to classify sample in a different way?

By examining the dual form of the optimization problem, we gained significant insight into the structure of the problem, and were also able to write the entire algorithm in terms of only inner products between input feature vectors. In the next section, we will exploit this property to apply the kernels to our classification problem. The resulting algorithm, support vector machines, will be able to efficiently learn in very high dimensional spaces.

2.2 Kernel SVM

From our drug dosage example, we can transform our original input *attributes* to new *features* as in Figure 1.

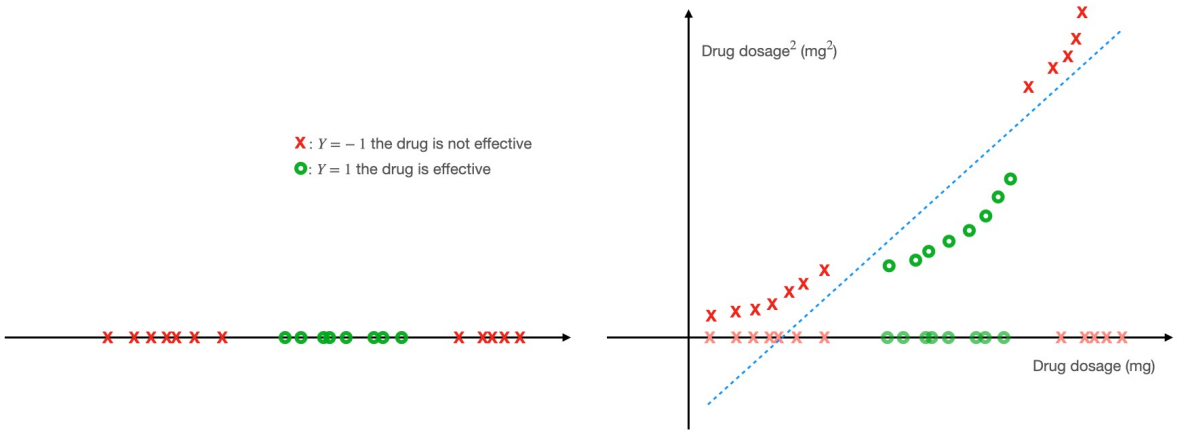

Figure 1: Can SVM be used to predict the drug-effective outcome? Yes!

Let's formalize this transformation. Let ϕ denote the *feature mapping*, which maps from the attributes to the features. For instance, in our example, we use

$$\phi(x) = \begin{bmatrix} x \\ x^2 \end{bmatrix}.$$

Rather than applying SVMs using the original input attributes x, we may instead want to learn using some features $\phi(x)$. To do so, we simply need to go over our previous algorithm, and replace X_i everywhere in it with $\phi(X_i)$.

Since the SVM dual problem (7) solely works with the inner products $\langle X_i, X_j \rangle$, this means that we would replace all those inner products with $\langle \phi(X_i), \phi(X_j) \rangle$. More specifically, given a feature mapping ϕ, we define its corresponding *Kernel* to be

$$K(X_i, X_j) = \langle \phi(X_i), \phi(X_j) \rangle = \phi(X_i)' \phi(X_j).$$

Then, everywhere we previously had $\langle X_i, X_j \rangle$, we could simply replace it with $K(X_i, X_j)$ and our algorithm would now be learning using the feature mapping ϕ.

Suppose $x, z \in \mathcal{X} \subseteq \mathbb{R}^d$, given a feature mapping ϕ, we could easily compute $K(x, z)$ by finding $\phi(x)$, $\phi(z)$ and taking their inner product. But what's more interesting is that, very often, $K(x, z)$ is very inexpensive to calculate, even though $\phi(x)$ itself may be very expensive to calculate. In such setting, kernel SVM incorporates easy-to-compute kernel function $K(x, z)$, and hence can learn in high-dimensional feature space (given by ϕ) without having to explicitly find or represent $\phi(x)$.

Example 1 (Degree-p polynomials kernel)**.** *For two input attributes defined in the attribute space $\mathcal{X}$, $x, z \in \mathcal{X} \subseteq \mathbb{R}^d$, we consider a Kernel*

$$K(x, z) = (x'z + c)^p.$$

When $p = 2$, we have

$$\begin{aligned} K(x, z) =& (x'z + c)^2 \\ =& \sum_{k=1}^{d} \sum_{l=1}^{d} (x_k x_l)(z_k z_l) + \sum_{k=1}^{d} (2\sqrt{c}x_k)(2\sqrt{c}z_k) + c^2. \end{aligned}$$

This means if we direct work with the original feature mapping, we have to work with a long feature vector. Take $d = 3$ for example:

$$\phi(x) = \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_1 x_3 \\ x_2^2 \\ x_2 x_1 \\ x_2 x_3 \\ x_3 x_2 \\ x_3^2 \\ \sqrt{2c}x_1 \\ \sqrt{2c}x_2 \\ \sqrt{2c}x_3 \\ c \end{bmatrix} \in \mathbb{R}^{12}, \quad \textit{where } x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \in \mathbb{R}^{d=3}.$$

More broadly speaking, the degree-p kernel $K(x, z) = (x'z + c)^p$ corresponds to a feature mapping to an $\binom{p+d}{d}-$dimensional feature space. Such a feature space is spanned by all monomials of the $x_{i_1}, \ldots x_{i_k}$ that are up to order p– thus this is a space of order d^p. Nevertheless, computing $K(x, z)$ takes only $O(d)$ time. Hence, we never need to explicitly represent feature vector in this very high-dimensional feature space.

Although the above description sounds mathematically complex, the kernel function is essentially measuring the similarity between two input attributes x and z. As long as you come up with some function $K(x, z)$ that you think might be a reasonable measure of how similar x and z are, it would be a reasonable kernel. Another popular choice is the Gaussian Kernel:

Example 2 (Gaussian Kernel)**.** *For two input attributes defined in the attribute space $\mathcal{X}$, $x, z \in \mathcal{X} \subseteq \mathbb{R}^d$, we consider a Kernel*

$$K(x, z) = \exp\left(-\frac{||x - z||_2^2}{2\sigma^2}\right)$$

This Gaussian kernel corresponds to an infinite dimensional feature mapping ϕ. But think about more broadly, given some random function K, how can we tell if it is a valid kernel? In other words, can we tell if there exist a feature mapping ϕ so that $K(x,z) = \phi(x)'\phi(z)$ for all x, z? This requires us to understand the Mercer's Theorem. If you are interested, you can take a look at Chapter 14.2 in Murphy (2012).

Keep in mind however that the idea of kernels has significantly broader applicability than SVMs. Specifically, if you have any learning algorithm that you can write in terms of only inner products $\langle x, z\rangle$ between input attribute vectors, then by replacing this with $K(x,z)$ where K is a kernel, you can "magically" allow your algorithm to work efficiently in the high dimensional feature space corresponding to K.

Lastly, to end this section on the support vector machines, we summarize its primal and dual problems:

$$\begin{aligned} \text{SVM primal problem:} \quad & \min_{a,\mathrm{w}} \ \frac{1}{2}\mathrm{w}'\mathrm{w} \\ & \text{s.t.} \ \ -Y_i\big(a + \mathrm{w}'\phi(X_i)\big) + 1 \le 0, \quad i = 1, \dots, n. \end{aligned} \tag{8}$$

and the dual problem

$$\begin{aligned} \text{SVM dual problem:} \quad & \max_{\nu} \ \sum_{i=1}^{n} \nu_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} Y_i Y_j \nu_i \nu_j K(X_i, X_j) \\ & \text{s.t.} \ \ \nu_i \ge 0, \ i = 1, \dots, n, \\ & \qquad \sum_{i=1}^{n} \nu_i Y_i = 0, \end{aligned} \tag{9}$$

where $K(X_i, X_j) = \langle \phi(X_i), \phi(X_j)\rangle$. The primal and dual solution can be linked by

$$\mathrm{w} = \sum_{i=1}^{n} \nu_i Y_i X_i.$$

2.3 A short story of SVM

The very first paper on the support vector machines with kernels is published in 1992; see Boser et al. (1992). You can read a short story on this webpage.

3 Empirical Risk Minimization

Recall in the SVM with soft constraint problem, we work with

$$\begin{aligned} \min_{a,\mathrm{w}} \ & \mathrm{w}'\mathrm{w} + C\sum_{i=1}^{n} \xi_i \\ \text{s.t.} \ & Y_i(a + \mathrm{w}' X_i) \ge 1 - \xi_i, \\ & \xi_i \ge 0, \quad i = 1, \dots, n. \end{aligned}$$

Whenever $C \neq 0$, the optimization problem will always try to minimize ξ_i as much as possible. Then the equation must hold as an equality and we have

$$\xi_i = \begin{cases} 0 & \text{if } Y_i(a + \mathrm{w}'X_i) \geq 1 \\ 1 - Y_i(a + \mathrm{w}'X_i) & \text{if } Y_i(a + \mathrm{w}'X_i) < 1. \end{cases}$$

Therefore, the slack variable ξ_i has a closed form expression:

$$\xi_i = \max\{1 - Y_i(a + \mathrm{w}'X_i), 0\}$$

When we plug this closed form into the objective of the SVM primal problem, we obtain the following unconstrained version as loss function:

$$\min_{a,\mathrm{W}} \; C \sum_{i=1}^{n} \underbrace{\max\Big[1 - Y_i(a + \mathrm{w}'X_i), 0\Big]}_{\text{hinge loss}} + \underbrace{\mathrm{w}'\mathrm{w}}_{l_2 \text{ penalty}} .$$

The hinge loss is intuitive to understand: the classifier h learnt by SVM is parametrized by w, and we make decisions based on

$$h_{\mathrm{W}}(X_i) = \begin{cases} 1 & a + \mathrm{w}'X_i \geq 1 \\ -1 & a + \mathrm{w}'X_i \leq -1 \end{cases} = 2\mathbf{1}(a + \mathrm{w}'X_i \geq 1) - 1 \triangleq 2\mathbf{1}(f_{\mathrm{W}}(X_i) \geq 1) - 1.$$

Whenever the classifier makes a mistake, we have $Y_i \cdot (a + \mathrm{w}'X_i) < 0$–generating a loss. Therefore, we define a hinge loss for the classifier h_{W} as

$$l\big(h_{\mathrm{W}}(X_i), Y_i\big) = \max\Big[1 - Y_i(a + \mathrm{w}'X_i), 0\Big] \triangleq \max\Big[1 - Y_i f_{\mathrm{W}}(X_i), 0\Big].$$

Then the SVM primal problem can be understood from a penalized "regression" perspective (suppose we ignore the intercept b for simplicity[3])

$$\min_{\mathrm{W} \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} l_{\texttt{svm}}\big(h_{\mathrm{W}}(X_i), Y_i\big) + \lambda \cdot \text{Penalty}(\mathrm{w}).$$

In your lab, you should have worked with GLM Lasso (or some general penalty) when we label the outcome as $Y_i \in \{-1, 1\}$ (try this one on your own, the optimization problem we worked out in GLM section is for $Y_i \in \{0, 1\}$):

$$\begin{aligned} \widehat{\beta} &= \arg\min_{b \in \mathbb{R}^d} \; \sum_{i=1}^{n} \log\big(1 + \exp(-X_i'b \cdot Y_i)\big) + \lambda \cdot \text{Penalty}(b), \\ &= \arg\min_{b \in \mathbb{R}^d} \; \sum_{i=1}^{n} l_{\texttt{logit}}\big(h_b(X_i), Y_i\big) + \lambda \cdot \text{Penalty}(b), \end{aligned}$$

[3]This works as once we have solved SVM for some w^*, then optimal value for b^* can be calculated immediately.

where the classifier h_b does the following:

$$h_b(X_i) = \begin{cases} 1 & \text{if } b'X_i \geq 0 \\ -1 & \text{if } b'X_i < 0. \end{cases}$$

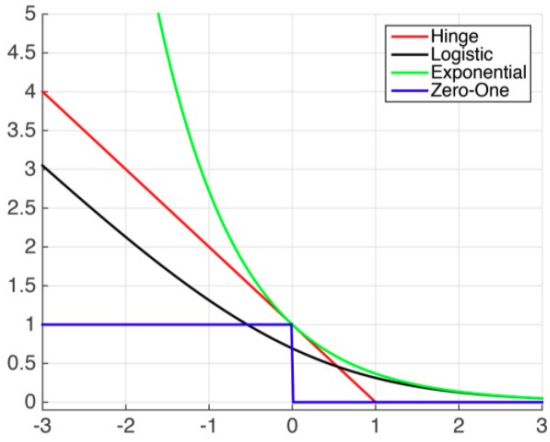

Figure 2: Common Classification Loss Functions on the x-axis: $f(X_i)Y_i$

Now, the logistic regression and SVM look oddly similar. Both say that if we misclassify more pairs (X_i, Y_i), the loss function (either $l_{\texttt{logit}}$ or $l_{\texttt{svm}}$) is large; the penalty is trying to make the classifier h simple (remember the Occam's razor principle?). In other words, we are trying to find a classifier that (1) makes less mistakes, and (2) as simple as possible. This principle applies in many famous estimators (ridge regression, Lasso, Elastic Net, etc.).

Let's now look into the difference between the SVM and the logistic losses in Figure 2. The exponential loss is of the form (this is a very sensitive/aggressive loss)

$$l_{\texttt{exponential}}(h(X_i), Y_i) = \exp\big(-h(X_i)Y_i\big).$$

Which loss do you like better? Why? Think about it from a statistical *inference* point of view.

We end this section by defining the general empirical risk minimization problem with a loss function l and some penalty function p:

$$\min_{\mathrm{W}} \underbrace{\frac{1}{n}\sum_{i=1}^{n} l\big(h_{\mathrm{W}}(X_i), Y_i\big)}_{\text{loss}} + \underbrace{\lambda \cdot p(\mathrm{w})}_{\text{penalty}},$$

where the loss function $l(\cdot, \cdot)$ is a function that penalize the difference between Y_i and $h_{\mathrm{W}}(X_i)$, and the penalty function $p(\cdot)$ penalizes the classifier's complexity.

References

Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.