10: Empirical Risk Minimization

Recap

Remember the Unconstrained SVM Formulation

$$\min_{\mathbf{w}} \; \underbrace{C\sum_{i=1}^{n} max[1-y_i\underbrace{(w^\top \mathbf{x}_i+b)}_{h(\mathbf{x}_i)},0]}_{Hinge-Loss} + \underbrace{\|w\|_z^2}_{l_2-Regularizer}$$

The hinge loss is the SVM's loss/error function of choice, whereas the l_2-regularizer reflects the complexity of the solution, and penalizes complex solutions. Unfortunately, it is not always possible or practical to minimize the true error, since it is often not continuous and/or differentiable. However, for most Machine Learning algorithms, it is possible to minimize a "Surrogate" Loss Function, which can generally be characterized as follows:

$$\min_{\mathbf{w}} \frac{1}{n}\sum_{i=1}^{n} \underbrace{l_{(s)}(h_{\mathbf{w}}(\mathbf{x}_i),y_i)}_{Loss} + \underbrace{\lambda r(w)}_{Regularizer}$$

...where the Loss Function is a continuous function which penalizes training error, and the Regularizer is a continuous function which penalizes classifier complexity. Here we define λ as $\frac{1}{C}$.[1] The science behind finding an ideal loss function and regularizer is known as Empirical Risk Minimization or Structured Risk Minimization.

Commonly Used Binary Classification Loss Functions

Different Machine Learning algorithms employ their own loss functions; Table 4.1 shows just a few:

Loss $\ell(h_{\mathbf{w}}(\mathbf{x}_i, y_i))$	**Usage**	**Comments**
1.Hinge-Loss $max[1-h_{\mathbf{w}}(\mathbf{x}_i)y_i, 0]^p$	• Standard SVM($p=1$) • (Differentiable) Squared Hingeless SVM ($p=2$)	When used for Standard SVM, the loss function denotes margin length between linear separator and its closest point in either class. Only differentiable everywhere at $p=2$.
2.Log-Loss $log(1+e^{-h_{\mathbf{w}}(\mathbf{x}_i)y_i})$	Logistic Regression	One of the most popular loss functions in Machine Learning, since its outputs are very well-tuned.
3.Exponential Loss $e^{-h_{\mathbf{w}}(\mathbf{x}_i)y_i}$	AdaBoost	This function is very aggressive. The loss of a mis-prediction increases *exponentially* with the value of $-h_{\mathbf{w}}(\mathbf{x}_i)y_i$.

4.Zero-One Loss $\delta(\text{sign}(h_{\mathbf{w}}(\mathbf{x}_i)) \neq y_i)$	Actual Classification Loss	Non-continuous and thus impractical to optimize.

Table 4.1: Loss Functions With Classification $y \in \{-1, +1\}$

Quiz: What do all these loss functions look like with respect to $z = y * h(\vec{x})$?

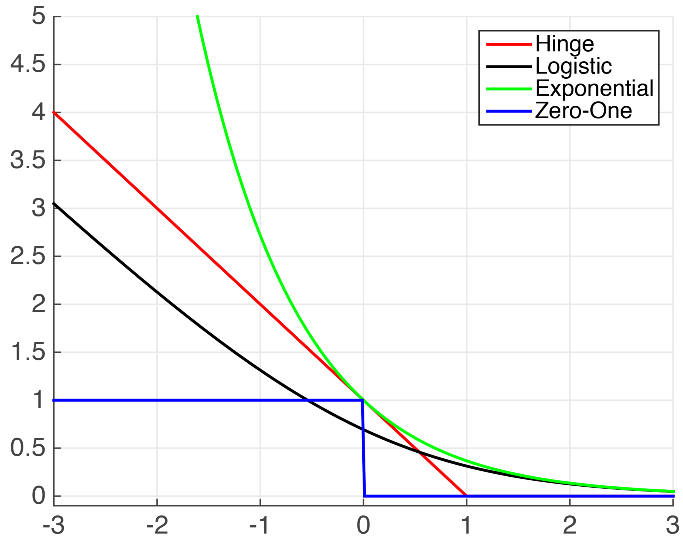

Figure 4.1: Plots of Common Classification Loss Functions - x-axis: $h(\mathbf{x}_i)y_i$, or "correctness" of prediction; y-axis: loss value

Some additional notes on loss functions:
- 1. As hinge-loss decreases, so does training error.
- 2. As $z \to -\infty$, the log-loss and the hinge loss become increasingly parallel.
- 3. The exponential loss and the hinge loss are both upper bounds of the zero-one loss. (For the exponential loss, this is an important aspect in Adaboost, which we will cover later.)
- 4. Zero-one loss is zero when the prediction is correct, and one when incorrect.

Commonly Used Regression Loss Functions

Regression algorithms (where a prediction can lie anywhere on the real-number line) also have their own host of loss functions:

Loss $\ell(h_{\mathbf{w}}(\mathbf{x}_i, y_i))$	**Comments**
1.Squared Loss $(h(\mathbf{x}_i) - y_i)^2$	• Most popular regression loss function • Estimates Mean Label • ADVANTAGE: Differentiable everywhere • DISADVANTAGE: Somewhat sensitive to outliers/noise • Also known as Ordinary Least Squares (OLS)
2.Absolute Loss $\lvert h(\mathbf{x}_i) - y_i \rvert$	• Also a very popular loss function • Estimates Median Label

	• ADVANTAGE: Less sensitive to noise • DISADVANTAGE: Not differentiable at 0
3.Huber Loss • $\frac{1}{2}\left(h(\mathbf{x}_i)-y_i\right)^2$ if $\lvert h(\mathbf{x}_i)-y_i\rvert<\delta$, • otherwise $\delta(\lvert h(\mathbf{x}_i)-y_i\rvert-\frac{\delta}{2})$	• Also known as Smooth Absolute Loss • ADVANTAGE: "Best of Both Worlds" of Squared and Absolute Loss • Once-differentiable • Takes on behavior of Squared-Loss when loss is small, and Absolute Loss when loss is large.
4.Log-Cosh Loss $log(cosh(h(\mathbf{x}_i)-y_i))$, $cosh(x)=\frac{e^x+e^{-x}}{2}$	ADVANTAGE: Similar to Huber Loss, but twice differentiable everywhere

Table 4.2: Loss Functions With Regression, i.e. $y \in \mathbb{R}$

Quiz: What do the loss functions in Table 4.2 look like with respect to $z=\left(h(\mathbf{x}_i)-y_i\right)^2$?

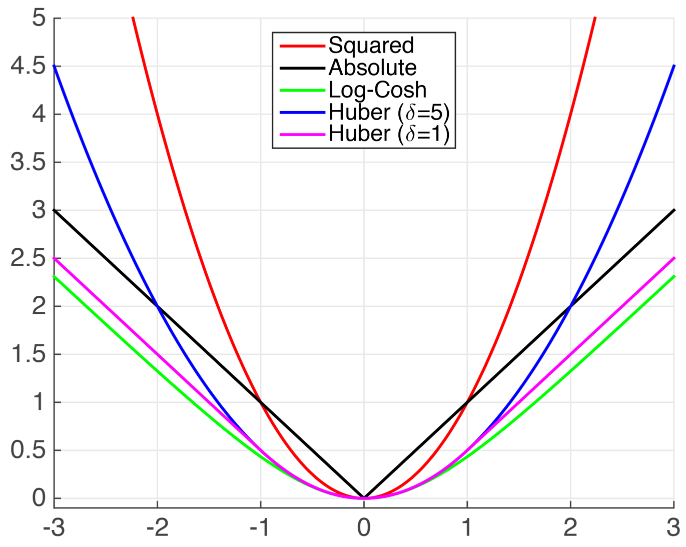

Figure 4.2: Plots of Common Regression Loss Functions - x-axis: $h(\mathbf{x}_i)y_i$, or "error" of prediction; y-axis: loss value

Regularizers

$$\min_{\mathbf{w},b}\sum_{i=1}^{n}\ell(\mathbf{w}^\top\vec{x}_i+b,y_i)+\lambda r(\mathbf{w})\Leftrightarrow\min_{\mathbf{w},b}\sum_{i=1}^{n}\ell(\mathbf{w}^\top\vec{x}_i+b,y_i)\text{ subject to: }r(w)\leq B$$

For each $\lambda \geq 0$, there exists $B \geq 0$ such that the two formulations in (4.1) are equivalent, and vice versa. In previous sections, l_2-regularizer has been introduced as the component in SVM that reflects the complexity of solutions. Besides the l_2-regularizer, other types of useful regularizers and their properties are listed in Table 4.3.

Regularizer $r(\mathbf{w})$	**Properties**

1.l_2-Regularization $r(\mathbf{w}) = \mathbf{w}^\top \mathbf{w} = (\\|\mathbf{w}\\|_2)^2$	• ADVANTAGE: Strictly Convex • ADVANTAGE: Differentiable • DISADVANTAGE: Uses weights on all features, i.e. relies on all features to some degree (ideally we would like to avoid this) - these are known as Dense Solutions.
2.l_1-Regularization $r(\mathbf{w}) = \\|\mathbf{w}\\|_1$	• Convex (but not strictly) • DISADVANTAGE: Not differentiable at 0 (the point which minimization is intended to bring us to • Effect: Sparse (i.e. not Dense) Solutions
3.Elastic Net $\alpha\\|\mathbf{w}\\|_1 + (1-\alpha)(\\|\mathbf{w}\\|_2)^2$ $\alpha \in [0, 1)$	• ADVANTAGE: Strictly convex (i.e. unique solution) • DISADVANTAGE: Non-differentiable
4.lp-Norm often $0 < p \leq 1$ $\\|\mathbf{w}\\|_p = (\sum_{i=1}^{d} v_i^p)^{1/p}$	• DISADVANTAGE: Non-convex • ADVANTAGE: Very sparse solutions • Initialization dependent • DISADVANTAGE: Not differentiable

Table 4.3: Types of Regularizers

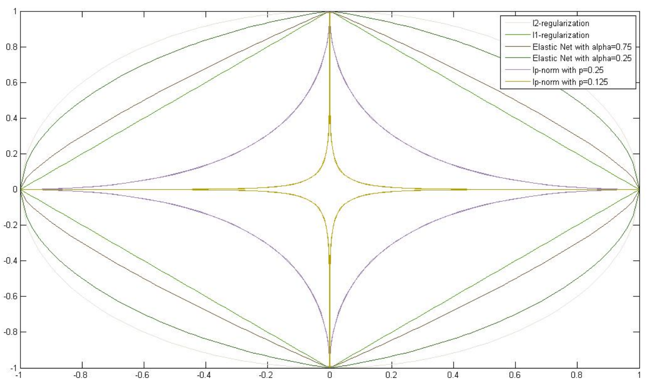

Figure 4.3: Plots of Common Regularizers

Famous Special Cases

This section includes several special cases that deal with risk minimization, such as Ordinary Least Squares, Ridge Regression, Lasso, and Logistic Regression. Table 4.4 provides information on their loss functions, regularizers, as well as solutions.

Loss and Regularizer	Classification	Solutions

1.Ordinary Least Squares $\min_{\mathbf{w}} \frac{1}{n}\sum_{i=1}^{n}(\mathbf{w}^\top x_i - y_i)^2$	• Squared Loss • No Regularization	• $\mathbf{w} = (\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{y}^\top$ • $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ • $Y = [y_1, \ldots, y_n]$
2.Ridge Regression $\min_{\mathbf{w}} \frac{1}{n}\sum_{i=1}^{n}(\mathbf{w}^\top x_i - y_i)^2 + \lambda\|w\|_2^2$	• Squared Loss • l_2-Regularization	• $w = (\mathbf{X}\mathbf{X}^\top + \lambda\mathbb{I})^{-1}\mathbf{X}^\top\mathbf{y}^\top$
3.Lasso $\min_{\mathbf{w}} \frac{1}{n}\sum_{i=1}^{n}(\mathbf{w}^\top \mathbf{x}_i - \overrightarrow{y_i})^2 + \lambda\|\mathbf{w}\|_1$	• + sparsity inducing (good for feature selection) • + Convex • - Not strictly convex (no unique solution) • - Not differentiable (at 0)	• Solve with (sub)-gradient descent or SVEN
4.Logistic Regression $\min_{\mathbf{w},b} \frac{1}{n}\sum_{i=1}^{n}\log\left(1 + e^{-y_i(\mathbf{w}^\top \mathbf{x}_i + b)}\right)$	• Often l_1 or l_2 Regularized	• Estimation: • $\Pr(y = +1 \mid x) = \frac{1}{1+e^{-y(\mathbf{w}^\top x + b)}}$

Table 4.4: Special Cases

Some additional notes on the Special Cases:

- 1. Ridge Regression is very fast if data isn't too high dimensional.
- 2. Ridge Regression is one of the first ways to optimize in MATLAB in a Machine Learning setting.
- 3. A noteworthy counterpart to Ordinary Least Squares is PCA (Principal Component Analysis) also minimizes square loss, but looks at perpendicular loss (the horizontal distance between each point and the regression line) instead.

---

[1] In Bayesian Machine Learning, it is possible to optimize λ, but for the purposes of this class, it is assumed to be fixed.