Regularized Regression under Quadratic Loss, Logistic Loss, Sigmoidal Loss, and Hinge Loss

Here we considerthe problem of learning binary classiers. We assume a set $\mathcal{X}$ of possible inputs and we are interested in classifying inputs into one of two classes. For example we might be interesting in predicting whether a given persion is going to vote democratic or republican. We assume a function Φ which assigns a feature vector to each element of x — we assume that for $x \in \mathcal{X}$ we have $\Phi(x) \in R^d$. For $1 \leq i \leq d$ we let $\Phi_i(x)$ be the ith coordinate value of $\Phi(x)$. For example, for a person x we might have that $\Phi(x)$ is a vector specifying income, age, gender, years of education, and other properties. Discrete properties can be represented by binary valued fetures (indicator functions). For example, for each state of the United states we can have a component $\Phi_i(x)$ which is 1 if x lives in that state and 0 otherwise. We assume that we have training data consisting of labeled inputs where, for convenience, we assume that the labels are all either -1 or 1.

$$\begin{aligned} S &= \langle x_1, y_y \rangle, \ldots, \langle x_T, y_T \rangle \\ x_t &\in \mathcal{X} \\ y_t &\in \{-1, 1\} \end{aligned}$$

Our objective is to use the training data to construct a predictor $f(x)$ which predicts y from x. Here we will be interested in predictors of the following form where $\beta \in R^d$ is a parameter vector to be learned from the training data.

$$f_\beta(x) = sign(\beta \cdot \Phi(x)) \tag{1}$$

We are then interested in learning a parameter vector β from the training data. We first define the margin $m_t(\beta)$ as follows.

$$m_t(\beta) = y_t(\beta \cdot \Phi(x_t)) \tag{2}$$

The parameter vector β will usually be clear from context and we will usually write m_t instead of $m_t(\beta)$. Note that we have $m_t \geq 0$ if and only if $y_t = sign(\beta \cdot \Phi(x_t)) = f_\beta(x_t)$. If $m_t > 0$ it is the "margin of safety" by which $f_\beta(x_t)$ is correct. If $m_t \leq 0$ then m_t is a measure of the margin by which $f_\beta(x_t)$ is wrong.

Here we are interested in setting β using the following formula.

$$\beta^* = \operatorname*{argmin}_\beta \sum_{t=1}^{T} L(m_t(\beta)) + \lambda ||\beta||^2 \tag{3}$$

The function L in (3) is called the loss function. We will consider four commonly used loss functions and some motivations for each. In practice the value of λ is tuned so as to maximize the empiricle performance of the resulting value of β^* on holdout data (holdout data is labeled data not used as part of the training data).

1 Quadratic Loss: Ridge Regression

For quadratic loss (also called L_2 loss) we define $L(m_t)$ as follows.

$$\begin{aligned} L(m_t) &= (m_t - 1)^2 \\ &= (y_t(\beta \cdot \Phi(x)) - y_t^2)^2 \\ &= y_t^2(\beta \cdot \Phi(x) - y_t)^2 \\ &= (\beta \cdot \Phi(x) - y_t)^2 \end{aligned} \tag{4}$$

So the general equation (3), under quadratic loss (4), is the same as ridge regression when we have $y_t \in \{-1, 1\}$.

Ridge regression in the general case with $y \in R$ has a simple probabilistic motivation. We can model $P(y|x, \beta)$ as a Gaussian.

$$P(y|x, \beta) = \frac{1}{Z_1} \exp(-\frac{(y - \beta \cdot \Phi(x))^2}{2\sigma^2}) \tag{5}$$

We can also place a Gaussian prior on β.

$$P(\beta) = \frac{1}{Z_2} e^{-\frac{||\beta^2||}{\gamma}} \tag{6}$$

By taking $\lambda = \frac{2\sigma^2}{\gamma}$ in (3) we get the following.

$$\begin{aligned} \beta^* &= \operatorname*{argmin}_\beta \sum_{t=1}^{T} (\beta \cdot \Phi(x) - y_t)^2 + \frac{2\sigma^2}{\gamma} ||\beta||^2 \\ &= \operatorname*{argmin}_\beta \sum_{t=1}^{T} \frac{(\beta \cdot \Phi(x) - y_t)^2}{2\sigma^2} + \frac{||\beta||^2}{\gamma^2} \\ &= \operatorname*{argmin}_\beta \sum_{t=1}^{T} \ln(1/(Z_1 P(y_t|x_t, \beta))) + \ln(1/(Z_2 P(\beta))) \\ &= \operatorname*{argmin}_\beta \sum_{t=1}^{T} \ln(1/P(y_t|x_t, \beta)) + \ln(1/P(\beta)) + T \ln(1/Z_1) + \ln(1/Z_2) \\ &= \operatorname*{argmin}_\beta \sum_{t=1}^{T} \ln(1/P(y_t|x_t, \beta)) + \ln(1/P(\beta)) \\ &= \operatorname*{argmax}_\beta P(\beta) P(y_1, \ldots, y_T | x_1, \ldots, x_T, \beta) \\ &= \operatorname*{argmax}_\beta P(\beta) P(x_1, \ldots, x_T) P(y_1, \ldots, y_T | x_1, \ldots, x_T, \beta) \\ &= \operatorname*{argmax}_\beta P(\beta, S) \\ &= \operatorname*{argmax}_\beta P(\beta | S) \end{aligned}$$

This is called a maximum a-posteriori (MAP) value for β. If we set $\lambda = 0$ in (3) then we get the maximum likelihood (ML) value for β. The ML value maximizes $P(S|\beta)$ rather than $P(\beta)P(S|\beta)$. In practice λ is tuned with holdout data.

2 Log Loss: Logistic Regression

Log loss is the following loss function.

$$\begin{aligned} L(m_t) &= \ln(1+\exp(-m_t)) \\ &= \ln(1/P(y_y|x_t,\beta)) \end{aligned} \tag{7}$$

$$P(y|x,\beta) = \frac{1}{Z}\exp(\frac{1}{2}y(\beta\cdot\Phi(x))) \tag{8}$$

$$\begin{aligned} P(y_t|x_t,\beta) &= \frac{\exp(\frac{1}{2}m_t)}{\exp(\frac{1}{2}m_t)+\exp(-\frac{1/2}{m\ t})} \\ &= \frac{1}{1+\exp(-m_t)} \end{aligned}$$

When we know a-priori that we have $y \in \{-1,1\}$, the Gaussian assumption (5) is clearly innappropriate and the conditional probability (8) seems more reasonable — equation (8) gives a distribution on the two element set $\{-1,1\}$ rather than a distribution on all the reals. As in the case of ridge regression, logistic regression has a Bayesian interpretation. Using log loss, assuming the prior (6), and setting λ to $1/\gamma$ in (3) yields the following.

$$\begin{aligned} \beta^* &= \operatorname*{argmin}_\beta \sum_{t=1}^{T} \ln(1/P(y_t|x_t,\beta))) + \ln(1/(Z_2P(\beta))) \\ &= \operatorname*{argmin}_\beta \sum_{t=1}^{T} \ln(1/P(y_t|x_t,\beta)) + \ln(1/P(\beta)) \\ &= \operatorname*{argmax}_\beta P(\beta)P(y_1,\ldots,y_T|x_1,\ldots,x_T,\beta) \\ &= \operatorname*{argmax}_\beta P(\beta|S) \end{aligned}$$

As in the case of quadratic loss, this is the MAP value of β and setting $\lambda = 0$ in (3) gives the ML value of β. In practice λ is tuned with holdout data.

3 0-1 Loss

One sometimes interested in minimizing the error rate.

$$\begin{aligned} \beta^* &= \operatorname*{argmin}_{\beta} \sum_{t=1}^{T} L_{01}(m_t) \qquad (9) \\ L_{01}(m_t) &= I[m_t \leq 0] = I[f_\beta(x_t) \neq y_t] \end{aligned}$$

Unfortunately, one cannot use regularization with (9). If $d >> T$ (where d is the dimensionality of β and $\Phi(x)$) then there often exists a β which fits the data exactly but that overfits — it does not predict well on new data. The standard approach to avoiding overfitting is to require that the prediction rule be simple. We would like to use $||\beta||^2$ as a measure of the simplicity of β. But such regularization fails for (9). The problem is that if β achieves zero training loss then so does $\epsilon\beta$ for arbitrarily small $\epsilon > 0$. So we can achieve zero training error with $||\beta||$ arbitrarily small. To minimize the classification error rate on fresh data one can replace 0-1 loss in the training formula with sigmoidal loss.

4 Sigmoidal Loss

Sigmoidal loss is the following loss function.

$$L(m_t) = \frac{1}{1 + \exp(m_t)} \qquad (10)$$

Sigmoidal loss is similar to L_{01} except that it makes a continuous transition from 1 to 0 around $m_t = 0$. Even if β s infinite dimensional, as discussed in the notes on kernels, sigmoidal loss is "consistent" for optimizing the 0-1 error rate on fresh data. Intuitively, consistency means that the learning algorithm approaches optimal behavior as the size of the training data approaches infinity. To make this precise, recall that the sample S is a sequence of pairs $\langle x_1, y_1\rangle, \ldots, \langle x_T, y_T\rangle$ where $\langle x_t, y_t\rangle$ is drawn from a fixed distribution D on $\mathcal{X} \times \{-1, 1\}$. We write $S \sim D^T$ to indicate that S is a sequence of T items drawn IID according to D. Now consider the following quantities.

$$\begin{aligned}
\beta(S,\lambda) &= \operatorname*{argmin}_{\beta} \left(\sum_{\langle x,y\rangle \in S} \frac{1}{1+\exp(y(\beta\cdot\Phi(x)))} \right) + \lambda||\beta||^2 \\
L_{01}(\beta) &= P_{\langle x,y\rangle\sim D}(f_\beta(x) \neq y) \\
L_{01}(\lambda,T) &= \mathrm{E}_{S\sim D^T}\left[L_{01}(\beta(S,\lambda))\right]
\end{aligned}$$

We have that $L_{01}(\lambda, T)$ is the expected error rate when we learn β on a sample of size T using regularization parameter λ. Sigmoidal loss is consistent in the sense that, as the sample size goes to infinity and λ is tuned with holdout, we get an optimal parameter vector β. More formally we have the following.

$$\lim_{T\to\infty} \inf_{\lambda} L_{01}(T,\lambda) = \inf_{\beta} L_{01}(\beta) \tag{11}$$

For infinite dimensional β one must use regularization — unregularized training continues to overfit even as $T \to \infty$.

Unfortunately consistency does not imply any statement about the error rate of the parameter vector β learned from a finite sample. For a variant of sigmoidal loss a finite sample theorem is possible. It is possible to state a generalization theorem guaranteeing the performance of a certain stochastic classification algorithm on new data. In partcular, for any $\beta \in R^d$ define Q_β to be the following probability density.

$$Q_\beta(\beta') = \frac{1}{Z}\exp(-\frac{1}{2}||\beta'||^2)$$

In other words Q_β is a isotropic (same in all directions) Gaussian of unit variance centered at β. We consider the prediction algorithm that samples a vector β' from Q_β and then returns the prediction $f_{\beta'}(x)$. For technical reasons this stochastic classifier yields a simpler theorem than a direct analysis of the error rate of f_β. In particular we have the following theorem where $\forall^\delta S\ \Gamma(S,\delta)$ means that with probability at least $1-\delta$ over the choice of the sample S we have that $\Gamma(S,\delta)$ holds.

$$\begin{aligned}
\forall^\delta S\ \ \forall \beta \in R^d\ \ \mathrm{E}_{\beta'\sim Q_\beta}\left[L_{01}(\beta')\right] &\leq \left(\frac{1}{T}\sum_{t=1}^{T} S(m_t)\right) + \sqrt{\frac{\frac{1}{2}||\beta||^2 + \ln\frac{T+1}{\delta}}{2T}} \\
S(m) &= \int_m^\infty \frac{1}{\sqrt{2\pi}}\exp(-x^2/2)dx
\end{aligned}$$

In this theorem we have that $S(m)$ is a cummulative of a Gausssian. This is a sigmoidal function that can be viewed as a smooth approximation of 0-1 loss.

5 Convexity

The general regulalrized equation (3) has the property that if $L(m_t)$ is convex in m_t then the right hand side of (3) is convex in β and the optimization can be done in polynommial time. To see this we note that $m_t(\beta)$ is linear in β. A convex function of a linear function is convex. Hence $L(m_t(\beta))$ is convex in β. For $\lambda \geq 0$ we have that $\lambda||\beta||^2$ is convex. The convexity of (3) as a function of β then follows from the fact that a sum of convex functions is convex.

Quadratic loss and log loss are both convex functions of m_t. However, 0-1 loss and sigmoidal loss are not convex. Minimizing 0-1 loss is NP hard.

6 Hinge Loss: Support Vector Machines

Hinge Loss is the following.

$$L(m_t) = \max(0, 1 - m_t) \tag{12}$$

Recall that an algorithm is consistent if it behaves optimally in the limit of infinite training data. Hinge loss is consistent for minimizing 0-1 loss provided that β is infinite dimensional and any function $f(x)$ can be approximated arbitrarily closely by $\beta \cdot \Phi(x)$. But Hinge loss need not be consistent for optimizing 0-1 loss when d is finite. However, unlike sigmoidal loss, hinge loss is convex. Furthermore, equation (3) under hinge loss defines a convex quadratic program which can be solved more directly than can the optimization problem of logistic regression. Logistic regression is often solved by gradient descent. Equation (3) under hinge loss is called a support vector machine.

7 Rescaling the Loss Function

We now consider the case where we are interested in the error rate of the learned vector β^* on new data. As mentioned above, to use regularization we cannot directly use 0-1 loss in the training formula (3). Therefore, even though we are ultimately interested in generalization performance as measured by 0-1 loss, the loss L used in (3) must be some other loss such as quadratic loss, log loss, sigmoidal loss or hinge loss.

Consider equation (3) for an arbitrary loss function L. Define L' in terms of $\alpha, \gamma > 0$ as follows.

$$L'(m_t) = \alpha L(\gamma m_t))$$

The loss function L' is a rescaling of L involving both an arbitrary resclaling of the margin and an arbirtrary rescaling of the loss quantity. It is possible to show that when λ is tuned with holdout data to optimize holdout 0-1 loss, the rescaled loss L' performs exactly like L. To see this assume that λ is an optimal regularization parameter for L and T. We then have the following.

$$\begin{aligned}
\beta^* &= \operatorname*{argmin}_{\beta} \sum_{t=1}^{T} L(m_t(\beta)) + \lambda||\beta||^2 \\
&= \operatorname*{argmin}_{\beta} \sum_{t=1}^{T} \alpha L(m_t(\beta)) + \alpha\lambda||\beta||^2 \\
&= \operatorname*{argmin}_{\beta} \sum_{t=1}^{T} \alpha L(\gamma m_t(\frac{\beta}{\gamma})) + \alpha\lambda||\beta||^2 \\
\beta'^* &= \operatorname*{argmin}_{\beta'} \sum_{t=1}^{T} \alpha L(\gamma m_t(\beta')) + \alpha\gamma^2\lambda||\beta'||^2 \\
&= \operatorname*{argmin}_{\beta} \sum_{t=1}^{T} L'(m_t) + \lambda'||\beta||^2 \\
\beta^* &= \gamma\beta'^*
\end{aligned}$$

Because β^* and β'^* are in the same direction, they give the same $0-1$ loss. Hence training with L' and λ' will result in exactly the same 0-1 holdout performance as training with L and λ. This suggest that the exact choice of loss function when optimizing 0-1 holdout loss is not very critical. Two loss functions can be made to look similar when we can rescale both the margin value and the loss value arbitrarily. It also implies that there is no performance advantage in generalizing Hinge loss to be of the form $\max(0, \alpha - \gamma m_t)$ for parameters α and γ or in generalizing sigmoidal loss to allow sigmoids of different widths.