EMPIRICAL RISK MINIMIZATION

We will see that several algorithms already discussed fall under a common framework.

Performance Measures for Supervised Learning

Consider a supervised learning problem (classification or regression) with jointly distributed (X, Y). Let $\mathcal{Y}$ denote the output space (regression: $\mathcal{Y} = \mathbb{R}$, binary classification: $\mathcal{Y} = \{-1, +1\}$).

A loss is a function $L(y, t)$ where $t \in \mathbb{R}$ and $y \in \mathcal{Y}$. Let $f : \mathbb{R}^d \to \mathbb{R}$. The L-risk of f is defined to be

$$R_L(f) = \mathbb{E}_{X,Y}\left[L(Y, f(X))\right]$$

where f is a classifier or regression function.

Examples 1

In regression, f is a regression function. If

$$L(y,t) = (y-t)^2$$

squared error loss

then R_L is the mean squared error. If

$$L(y,t) = |y-t|$$

absolute deviation loss

then R_L is the mean absolute error.

In binary classification, f is a decision function that defines a classifier by

$$y \longmapsto \mathrm{sign}(f(x))$$

where $\mathrm{sign}(t) = \begin{cases} 1 & t \geq 0 \\ -1 & t < 0 \end{cases}$. For example,

$f(x) = w^T x + b$ defines a linear classifier. If

$$L(y,t) = \mathbb{1}_{\{y \neq \mathrm{sign}(t)\}}$$

0-1 loss

then R_L is the probability of error.

$\boxed{\text{Empirical Risk Minimization}}$

Empirical Risk Minimization

Given training data $(x_1,y_1), \dots, (x_n,y_n)$, a natural way to learn a good f is solve

$$\min_{f \in \mathcal{F}} \frac{1}{n}\sum_i L(y_i, f(x_i)) + \lambda \Omega(f)$$

set of candidates f's, e.g. linear functions

regularizer, e.g. $\Omega(f) = \|w\|^2$ if $f(x) = w^T x + b$

This problem is called (penalized/regularized) empirical risk minimization. The quantity

$$\hat{R}(f) = \frac{1}{n}\sum_i L(y_i, f(x_i))$$

is called the empirical L-risk of f.

We have already seen ERM in regression in the form of least squares regression and robust regression.

ERM is a nice optimization problem when the loss is convex.

$\boxed{\text{Exercise}}$ If $L(y,t)$ is a convex function of t for each y, then

$$\hat{R}(w,b) = \frac{1}{n}\sum_i L(y_i, w^T x_i + b)$$

is a convex function of $\theta = \begin{bmatrix} b \\ w \end{bmatrix}$.

In binary classification, however, the situation is not so nice. The problem is that the 0/1 loss

$$L(y,t) = \mathbb{1}_{\{y \neq \text{sign}(t)\}}$$

is not convex in t. In fact, it's not even differentiable so we can't even apply gradient descent.

Surrogate Losses

A surrogate loss is a loss that takes the place of another, usually because of nicer computational properties (convexity, differentiability).

Some common surrogate losses for binary classification are

$$L(y,t) = \log(1+e^{-yt})$$ logistic loss

$$L(y,t) = \max(0, 1-yt)$$ hinge loss

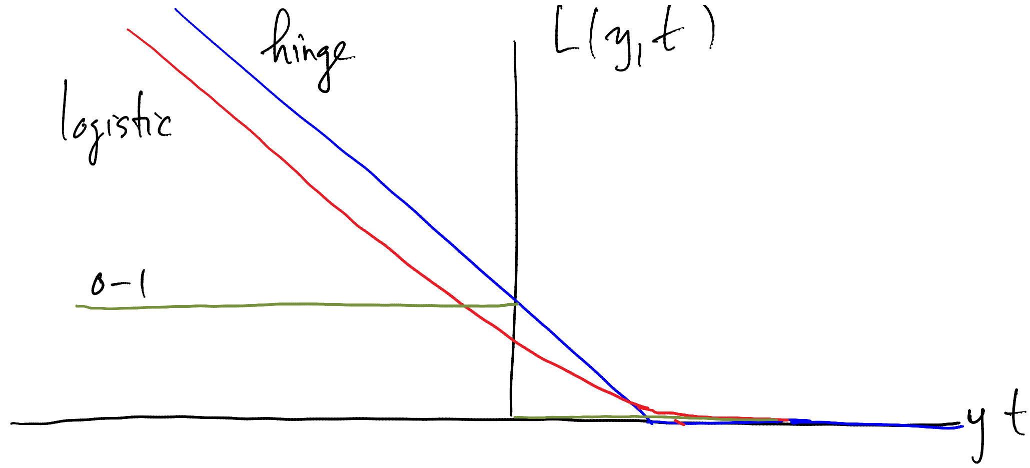
Notice that both depend on y and t only through the product yt, which is sometimes called the algebraic margin, as opposed to the geometric margin of a separating hyperplane.

The 0-1 loss satisfies

$$\mathbb{1}_{\{y \neq \mathrm{sign}(t)\}} \leq \mathbb{1}_{\{yt \leq 0\}}$$

"almost" an equality

These losses can be compared graphically



So surrogate losses still penalize mistakes and not correct predictions.

Many classification algorithms can be viewed as ERM for a certain L, $\mathcal{F}$, and Ω. In fact, we have already seen two of them.

On the homework, you will show that

$$-\ell(\theta) = \sum_{i=1}^{n} L(y_i, f_\theta(x_i))$$

where $\ell(\theta)$ is the logistic regression log-likelihood, L is the logistic loss, and $f_\theta(x_i) = \theta^T \tilde{x}_i$.

Recall the optimal soft margin hyperplane solves

(OSM) $$\min_{w,b,\xi} \quad \frac{1}{2}\|w\|^2 + \frac{C}{n}\sum_{i=1}^{n} \xi_i$$

$$\text{s.t.} \quad y_i(w^T x_i + b) \geq 1 - \xi_i \qquad \forall i$$

$$\xi_i \geq 0 \qquad \forall i$$

If $\lambda = \frac{1}{C}$, then the solution (w^*, b^*) also solves

(ERM-hinge) $$\min_{w,b} \quad \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum \max(0, 1 - y_i(w^T x_i + b))$$

This can be seen by scaling the objective function of (OSM) by $\frac{1}{C}$, which doesn't change the solution, and merging the constraints into a single constraint (for each i):

$$\begin{matrix} y_i(w^T x_i + b) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{matrix} \iff \xi_i \geq \max(0, 1 - y_i(w^T x_i + b))$$

So (OSM) reduces to

$$\min_{w,b,\xi} \quad \frac{1}{2}\|w\|^2 + \frac{1}{n}\sum \xi_i$$

$$\text{s.t.} \quad \xi_i \geq \max\{0, 1 - y_i(w^T x_i + b)\}$$

Clearly the solution must satisfy

$$\xi_i = \max\{0, 1 - y_i(w^T x_i + b)\} \qquad \forall i$$

(otherwise we could decrease the objective), which reduces the problem to (ERM-hinge).

Big Picture

Different choices of L, $\mathcal{F}$, and Ω give rise to different methods. We will see several other examples.

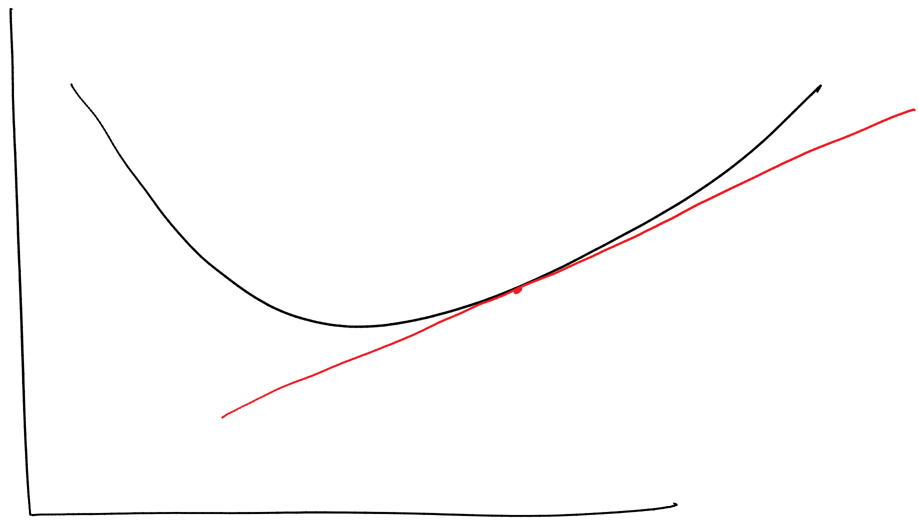
One advantage of this framework is that it makes it easier to compare and contrast different methods. Another is that there are optimization strategies that can be used to solve large classes of ERM methods. Examples include majorization/minimization, gradient descent, and subgradient methods.

Subgradient Methods

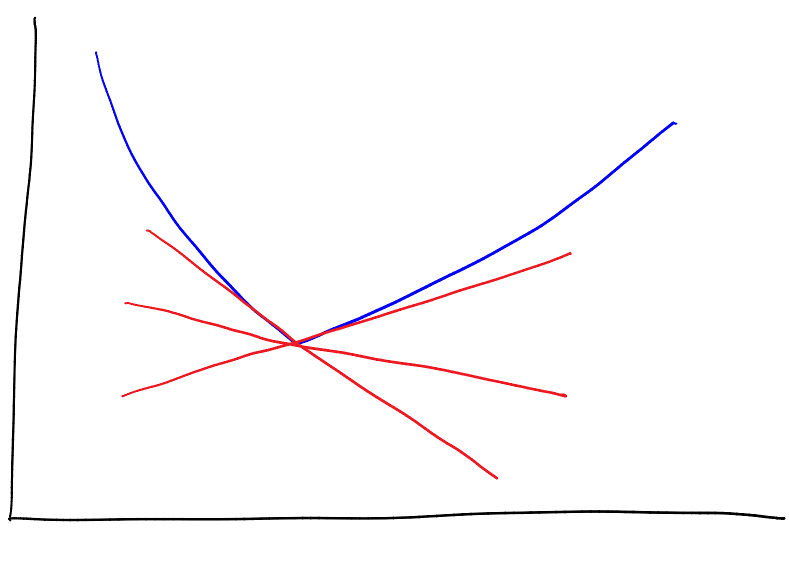
The subgradient method is a generalization of gradient descent that applies to nondifferentiable, convex functions, like ERM with hinge loss.

Let $g : \mathbb{R}^d \to \mathbb{R}$ be convex, and let $x \in \mathbb{R}^d$. If g is differentiable, then $u = \nabla g(x)$ is then only vector such that

$$☆ \quad g(y) \geq g(x) + u^T (y - x) \qquad \forall y.$$



If g is convex but not differentiable, then for some x, there may be many u satisfying ☆.
We define the subdifferential of g at x, denoted $\partial g(x)$, to be the set of all u satisfying ☆.
A subgradient is any element of the subdifferential.



In the figure above, the subdifferential is the interval $[g'_-(x), g'_+(x)]$ where g'_-, g'_+ denote the left and right derivatives.

In the subgradient method, we update the parameter just as in gradient descent, but where the gradient is replaced by any subgradient. Here's the pseudocode for minimizing $g(\theta)$

- initialize θ_0
- $t \leftarrow 0$
- Repeat
  - select $u_t \in \partial g(\theta_t)$
  - $\theta_{t+1} \leftarrow \theta_t - \alpha_t u_t$
  - $t \leftarrow t+1$

  Until stopping criterion satisfied

If it is possible to write $g(\theta) = \sum_{i=1}^{n} g_i(\theta)$, If it is possible to write $g(\theta) = \sum_{i=1}^{n} g_i(\theta)$, then we can also have a stochastic subgradient method. You'll get to experiment with this on the homework.

Note / Unlike gradient descent, where one can always find a step-size such that the objective decreases (unless you're at a local min), the objective will occaisionally increase as you iterate a subgradient method.