Support Vector Machine review

Chanwoo Lee, March 23, 2020

1 Linear SVM

1.1 Margin and Hard-SVM

Let $S = (\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_m, y_m)$ be a training set, where each $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. The goal is to find a halfspace $(\boldsymbol{w}, b)$, such that $y_i = \text{sign}(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b)$ for all i if there exists. Alternatively this condition can be written as

$$\forall i \in [m], \quad y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b) > 0.$$

Hard-SVM is the learning rule in which we return an hyperplane that separates the training set with the largestpossible margin.

Lemma 1. *The distance between a point $\boldsymbol{x}$ and the hyperplane defined by $(\boldsymbol{w}, b)$ where $\|\boldsymbol{w}\| = 1$ is $|\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b|$*

From this claim, the Hard-SVM rule is

$$\underset{(\boldsymbol{w},b):\|\boldsymbol{w}\|=1}{\arg\max} \min_{i \in [m]} |\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b| \quad \text{s.t. } \forall i, y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b) > 0.$$

We can write an equivalent problem as follows

$$\underset{(\boldsymbol{w},b):\|\boldsymbol{w}\|=1}{\arg\max} \min_{i \in [m]} y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b). \tag{1}$$

The algorithm is as follows

Algorithm 1: Hard-SVM

Input: $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_m, y_m)$

Solve:

$$(\boldsymbol{w}_0, b_0) = \underset{(\boldsymbol{w},b)}{\arg\min} \|\boldsymbol{w}\|^2 \text{ s.t.} \forall i, y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b) \geq 1 \tag{2}$$

Output: $\hat{\boldsymbol{w}} = \frac{\boldsymbol{w}_0}{\|\boldsymbol{w}_0\|}, \quad \hat{b} = \frac{b_0}{\|\boldsymbol{w}_0\|}$

Lemma 2. *Equation (1) and Equation (2) are equivalent.*

In homogenuous case where the bias term b set to be zero, Equation (2) becomes

$$\min_{\boldsymbol{w}} \|\boldsymbol{w}\|^2 \quad \text{s.t.} \forall i, y_i \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle \geq 1. \tag{3}$$

1.1.1 The sample complexity of Hard-SVM

Notice the VC-dimension of halfspace in $\mathbb{R}^d$ is $d + 1$. This VC-dimension implies that the sample complexity of learning halfspaces grows with the dimensionality of the problem. The fundamental theorem of learning tells us that if the number of examples is significantly smaller than d/ϵ, then

no algorithm can learn an ϵ-accurate halfspace. For Hard-SVM, we assume the data is separable with margin γ then the sample complexity is bounded from above by a function of $1/\gamma^2$. To specify this more accurately, we define,

Definition 1. Let $\mathcal{D}$ be a distribution over $\mathbb{R}^d \times \{-1, 1\}$. We say $\mathcal{D}$ is seprable with a (γ, ρ)-margin if there exists $(\boldsymbol{w}^*, b^*)$ such that $\|\boldsymbol{w}^*\| = 1$ and such that with probability 1 over the choice of $(\boldsymbol{x}, y) \sim \mathcal{D}$ we have that $y(\langle \boldsymbol{w}^*, \boldsymbol{x}\rangle + b^*) \geq \gamma$ and $\|\boldsymbol{x}\| \geq \rho$.

From Definition 1, we have the following sample complexity.

Theorem 1.1 (Sample complexity in Hard-SVD)**.** *Let $\mathcal{D}$ be a distribution over $\mathbb{R}^d \times \{-1, 1\}$ that satisfies the (γ, ρ)-separability with margin assumption. Then with probability of at least $1 - \delta$ over the choice of a training set of size m, the 0-1 error of the output of Hard-SVM is at most*

$$\sqrt{\frac{4(\rho/\gamma)^2}{m}} + \sqrt{\frac{2\log(2/\delta)}{m}}.$$

1.2 Soft-SVM and norm regularization

Soft-SVM is a relaxation of the Hard-SVM rule which assumes that the training set is linearly separable. The algorithm is follows

Algorithm 2: Soft-SVM

Input: $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_m, y_m)$
Parameter: $\lambda > 0$
Solve:

$$min_{(\boldsymbol{w},b,\xi)} \left(\lambda \|\boldsymbol{w}\|^2 + \frac{1}{m} \sum_{i=1}^{m} \xi_i \right) \tag{4}$$
$$\text{s.t.} \forall i, y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i\rangle + b) \geq 1 - \xi_i \quad \text{and } \xi_i \geq 0$$

Output: $\boldsymbol{w}, b$

Define the hinge loss:

$$\ell^{\text{hinge}}((\boldsymbol{w}, b), (\boldsymbol{x}, y) = \max\{0, 1 - y(\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b)\}.$$

Given $(\boldsymbol{w}, b)$ and a training set S, the averaged hinge loss on S is denoted by $L_S^{\text{hinge}}((\boldsymbol{w}, b))$.

Lemma 3. *Equation (4) is equivalent to the following Equation (5)*

$$\min_{\boldsymbol{w},b} \left(\lambda \|\boldsymbol{w}\|^2 + L_S^{hinge}((\boldsymbol{w}, b)) \right). \tag{5}$$

Equation (5) is sometimes better to use when implemening algorithm.

1.2.1 The sample complexity of Soft-SVM

This section the sample complexity of Soft-SVM is for the case of homogenuous halfspaces where $b = 0$.

Theorem 1.2. *Let $\mathcal{D}$ be a distribution over $\mathcal{X} \times \{0, 1\}$, where $\mathcal{X} = \{\boldsymbol{x} : \|\boldsymbol{x}\| \leq \rho\}$. Consider running Soft-SVM (Equation (5)) on a training set $S \sim \mathcal{D}^m$ and let $A(S)$ be the solution of Soft-SVM. Then, for every μ,*

$$\mathbb{E}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{hinge}(A(S))] \leq L_{\mathcal{D}}^{hinge}(\mu) + \lambda\|\mu\|^2 + \frac{2\rho^2}{\lambda m}.$$

For everny $B > 0$, if we set $\lambda = \sqrt{\frac{2\rho^2}{B^2 m}}$ then,

$$\mathbb{E}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{0\text{-}1}(A(S))] \leq \mathbb{E}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{hinge}(A(S))] \leq \min_{\boldsymbol{w}:\|\boldsymbol{w}\|\leq B} L_{\mathcal{D}}^{hinge}(\boldsymbol{w}) + \sqrt{\frac{8\rho^2 B^2}{m}}.$$

We therefore wee that we can control the sample complexity of learning a half-space as a function of the norm of that halfspace, independently of the Euclidean dimension of the space over which the halfspace is defined.

1.2.2 Implementing Soft-SVM using SGD

The optimization problem of Soft-SVM is

$$\min_{\boldsymbol{w}} \left(\frac{\lambda}{2}\|\omega\|^2 + \frac{1}{m}\sum_{i=1}^{m} \max\{0, 1 - y\langle \boldsymbol{w}, \boldsymbol{x}_i\rangle\} \right),$$

We rely on the SGD framework for solving regularized loss minimization problems. We ca write the update rule of SGD as

$$\boldsymbol{w}^{(t+1)} = -\frac{1}{\lambda t}\sum_{j=1}^{t} \boldsymbol{v}_j,$$

where $\boldsymbol{v}_j$ is a subgradient of the loss function at $\boldsymbol{w}^{(j)}$ on the random example chosen at iteration j. For the hinge loss, given an example $(\boldsymbol{x}, y)$, we can choose $\boldsymbol{v}_J$ to be 0 if $\langle \boldsymbol{w}^{(j)}, \boldsymbol{x}\rangle \geq 1$ and $\boldsymbol{v}_j = -y\boldsymbol{x}$ otherwise. Denoting $\boldsymbol{\theta} = -\sum_{j<t} \boldsymbol{v}_j$, we have Algorithm 4.

1.3 Optimal conditions and support vectors

The name "Support Vector Machine" stems from the fact that the solution of hard-SVM, $\boldsymbol{w}_0$, is supported by the examples that are exactly at distance $1/\|\boldsymbol{w}_0\|$ from the separating hyperplane. These vectors are called support vectors.

Theorem 1.3. *Let $\boldsymbol{w}_0$ be as defined in equation (3) and let $I = \{i : |langle\boldsymbol{w}_0, \boldsymbol{x}_i\rangle = 1\}$. Then, there exist coefficients $\alpha_1, \cdots, \alpha_m$ such that*

$$\boldsymbol{w}_0 = \sum_{i\in I} \alpha_i \boldsymbol{x}_i.$$

The examples $\{\boldsymbol{x}_i : i \in I\}$ are called support vectors.

Algorithm 3: SGD for solving Soft-SVM

Input: $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_m, y_m)$
Parameter: **T,** $\lambda > 0$
For $t = 1, \cdots, T$
  Let $\boldsymbol{w}^{(t)} = \frac{1}{\lambda t}\boldsymbol{\theta}^{(t)}$
  Choose i uniformly at random from $[m]$
  If $(y_i \langle \boldsymbol{w}^{(t)}, \boldsymbol{x}_i \rangle < 1)$
    Set $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + y_i \boldsymbol{x}_i$
  Else
    Set $= \boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)}$
Output: $\bar{\boldsymbol{w}} = \frac{1}{T}\sum_{i=1}^{T} \boldsymbol{w}^{(t)}$

1.4 Duality

Many of the properties of SVM have been obtained by considering the daul of Equation (3). We derive the daul of Equation (3). Consider the function

$$g(\boldsymbol{w}) = \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \geq 0} \sum_{i=1}^{m} \alpha_i (1 - y_i \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle) = \begin{cases} 0 & \text{if } \forall i, y_i \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle \geq 1 \\ \infty & otherwise \end{cases}.$$

Therefore, we can rewrite Equation (3) as

$$\begin{aligned} \min_{\boldsymbol{w}} (\|\boldsymbol{w}\|^2 + g(\boldsymbol{w})) &= \min_{\boldsymbol{w}} \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \geq 0} \left(\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_{i=1}^{m} \alpha_i (1 - y_i \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle) \right) \\ &\geq \max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \geq 0} \min_{\boldsymbol{w}} \left(\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_{i=1}^{m} \alpha_i (1 - y_i \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle) \right). \end{aligned}$$

By using weak duality(Lemma 4) we have the last inequality. It turns out that in our case, strong duality also holds; namely, the inequality holds with equality. Therefore the dual problem is

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \geq 0} \min_{\boldsymbol{w}} \left(\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_{i=1}^{m} \alpha_i (1 - y_i \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle) \right). \tag{6}$$

By first order necessary condition, we have

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i.$$

Therefore, Equation (6) can be written as

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^m : \boldsymbol{\alpha} \geq 0} \left(\sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \right).$$

This property is important when implementing SVM with kernels.

Lemma 4 (Weak Duality)**.** *For any function f, we have*

$$\min_{\boldsymbol{x} \in \mathcal{X}} \max_{\boldsymbol{y} \in \mathcal{Y}} f(\boldsymbol{x}, \boldsymbol{y}) \geq \max_{\boldsymbol{y} \in \mathcal{Y}} \min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}, \boldsymbol{y}).$$

2 Kernel SVM

2.1 The kernel trick

Given an embedding ψ for some domain space $\mathcal{X}$ into some Hilbert space, define the kernel function $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \psi(\boldsymbol{x}), \psi(\boldsymbol{x}') \rangle$. We can think of K as specifying similarity between instances and of the embedding ψ as mapping the domain set $\mathcal{X}$ into a space where these similarities are realized as inner products. We aim to construct kernel SVM algorithm on the basis of the values of the kernel function over pairs of domain points.

Notice all versions of the SVM optimization problem we have derived in the previous chapter are instances of the following general problem:

$$\min_{\boldsymbol{w}}(f(\langle \boldsymbol{w}, \psi(\boldsymbol{x}_1) \rangle, \cdots, \langle \boldsymbol{w}, \psi(\boldsymbol{x}_m) \rangle) + R(\|\boldsymbol{w}\|), \tag{7}$$

where $f : \mathbb{R}^m \to \mathbb{R}$ is an arbitrary function and $R : \mathbb{R}_+ \to \mathbb{R}$ is a monotonically nondecreasing function. The following theorem shows that there exists an optimal solution of Equation (7) that likes in the span of $\{\psi(\boldsymbol{x}_1), \cdots, \psi(\boldsymbol{x}_m)\}$.

Theorem 2.1 (Representer Theorem)**.** *Assume that ψ is a mapping from $\mathcal{X}$ to a Hilbert space. Then, there exists a vector $\boldsymbol{\alpha} \in \mathbb{R}^m$ such that $\boldsymbol{w} = \sum_{i=1}^m \alpha_i \psi(\boldsymbol{x}_i)$ is an optimal solution of Equation (7).*

Based on the representer theorem, we have the following equivalent problem.

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} f\left(\sum_{j=1}^m \alpha_j K(\boldsymbol{x}_j, \boldsymbol{x}_1), \cdots, \sum_{j=1}^m \alpha_j K(\boldsymbol{x}_j, \boldsymbol{x}_m)\right) + R\left(\sqrt{\sum_{i,j=1}^m \alpha_i \alpha_j K(\boldsymbol{x}_j, \boldsymbol{x}_i)}\right). \tag{8}$$

To solve Equation (8) we solely need to know the value of the $m \times m$ matrix G s.t. $G_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, which is often called the Gram matrix.

In particular, specifying the preceding to the Soft-SVM problem, we can write the problem given in Equation (5) as

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^m} \left(\lambda \boldsymbol{\alpha}^T G \boldsymbol{\alpha} + \frac{1}{2} \sum_{i=1}^m \max\{0, 1 - y_i (G\boldsymbol{\alpha}_i)\}\right).$$

2.2 Characterizing kernel function

Consider a given function of the form $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. The question that "Is it a valid kernel function?" is solved by the following lemma.

Lemma 5. *A symmetric function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ implements an inner product in some Hilbert space if and only if it is positive semi-definite; namely, for all $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_m$, the Gram matrix, $G_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, is a positive semi-definite matrix.*

2.3 Implementing Soft-SVM with kernels

While we could have designed an algorithm for solving Equation (8), there is an even simpler approach that directly tackles the Soft-SVM optimization problem in the feature space,

$$\min_{\boldsymbol{w}} \left(\frac{\lambda}{2}\|\omega\|^2 + \frac{1}{m}\sum_{i=1}^{m} \max\{0, 1 - y\langle \boldsymbol{w}, \psi(\boldsymbol{x}_i)\rangle\} \right),$$

while only using kernel evaluations. The basic observation is that the vector $\boldsymbol{w}^{(t)}$ maintained by the SGD procedure

Algorithm 4: SGD for solving Soft-SVM with Kernels

Input: $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_m, y_m)$
Parameter: T, $\lambda > 0$
Initizlize: $\boldsymbol{\beta}^{(0)} = 0$
For $t = 1, \cdots, T$
  Let $\boldsymbol{\alpha}^{(t)} = \frac{1}{\lambda t}\boldsymbol{\beta}^{(t)}$
  Choose i uniformly at random from $[m]$
  For all $j \neq i$ set $\beta_j^{(t+1)} = \beta_j^{(t)}$
  If $(y_i \sum_{j=1}^{m} \alpha_j^{(t)} K(\boldsymbol{x}_j, \boldsymbol{x}_i) < 1)$
    Set $\beta^{(t+1)} = \beta^{(t)} + y_i$
  Else
    Set $= \beta^{(t+1)} = \beta^{(t)}$
Output: $\bar{\boldsymbol{w}} = \sum_{i=1}^{m} \alpha_i \psi(\boldsymbol{x}_i)$ where $\bar{\boldsymbol{\alpha}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{\alpha}^{(t)}$

3 Generalization

3.1 Binary class classification

SVM objective function can be expressed more generally as

$$\min_{f \in \mathcal{H}} \frac{1}{m}\sum_{i=1}^{m} [1 - y_i f(\boldsymbol{x}_i)]_+ + \lambda J(f), \tag{9}$$

where $\mathcal{H}$ is the structured space of functions, and $J(f)$ an appropriate regularizer on that space. For example of $\mathcal{H}$, we can think $f(x) = b + \sum_{i=1}^{m} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i)$. Lin (2002) showed that the SVM solution $\hat{f}$ to Equation (9) targets directly at $\text{sign}[p_1(\boldsymbol{x}) - \frac{1}{2}]$, where $p_1(\boldsymbol{x}) = P(Y = 1 | \boldsymbol{X} = \boldsymbol{x})$. One can solve the regularization problem based on the weighted hinge loss

$$\min_{f \in \mathcal{H}} \frac{1}{m} \left[(1 - \pi) \sum_{y_i = 1} [1 - y_i f(\boldsymbol{x}_i)]_+ + \pi \sum_{y_i = -1} [1 - y_i f(\boldsymbol{x}_i)]_+ \right] + \lambda J(f), \tag{10}$$

for $\pi \in (0, 1)$. Wang,shen, and Liu (2008) showed that the minimizer to Equation (10) is a consistent estimate of $\text{sign}[p_1(\boldsymbol{x}) - \pi]$. Therefore, one can repeatedly solve Equation (10) using different $0 = \pi_1 < \cdots < \pi_{k+1} = 1$ and search $\hat{j}$ such that $\text{sign}[p_1(\boldsymbol{x}) - \pi_{\hat{j}}] \leq \text{sign}[p_1(\boldsymbol{x}) - \pi_{\hat{j}+1}]$. The probability can be estimated as $\hat{p}_1(\boldsymbol{x}) = \frac{1}{2}(\pi_{\hat{j}} + \pi_{\hat{j}+1})$.

3.2 Multiclass classification

Here observation space of y is $\{1, 2, \cdots, K\}$. A classifier seeks the function vector $\boldsymbol{f} = (f_1, \cdots, f_K)$, where f_k is a map from the input domain S to $\mathbb{R}$ representing the class k. To ensure uniqueness of the solution, a sum to zero contraint $\sum_{i=1}^{K} f_k = 0$ is usually employed. For any new input vector $\boldsymbol{x}$, its label is estimated via a decision rule $\hat{y} = \arg\max_{k=1,\cdots,K} f_k(\boldsymbol{x})$. Let us define

$$\boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y) = \{f_y(\boldsymbol{x}) - f_k(\boldsymbol{x}), k \neq y.\}$$

The quantity $\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y)$ is known as the generalized functional margin and can be reduced to $yf(\boldsymbol{x})$ in the binary case. For multiclass classification, we have the following regularization problem.

$$\min_{\boldsymbol{f}} \frac{1}{m} \sum_{i=1}^{m} \ell(\min(\boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i) + \lambda \sum_{k=1}^{K} J(f_k)$$
$$\text{subject to } \sum_{k=1}^{K} f_k(\boldsymbol{x}) = 0.$$

Define the unit K-cube hyperplane as

$$A_K = \{(\pi_1, \cdots, \pi_k) : \sum_{k=1}^{K} \pi_k = 1, \pi_k \geq 0, k - 1, 2, \cdots, K\}.$$

For any given $\boldsymbol{\pi} \in A_K$, one can solve the regularization problem based on the weighted loss

$$\min_{\boldsymbol{f}} \frac{1}{m} \sum_{i=1}^{m} \pi_{y_i} \ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)) + \lambda \sum_{k=1}^{K} J(f_k) \tag{11}$$
$$\text{subject to } \sum_{k=1}^{K} f_k(\boldsymbol{x}) = 0.$$

To construct a good probability estimate from the classification rules, we require that the loss function ℓ in Equation (11) is consistent in the following sense.

Definition 2. A functional margin base loss ℓ is called weighted Fisher-consistent for weighted classification problem if the minimizer $\boldsymbol{f}^*$ of $E[\pi_Y \ell(\min(\boldsymbol{g}(\boldsymbol{f}(\boldsymbol{X}, Y))|\boldsymbol{X} = \boldsymbol{x}]$ satisfies

$$\arg\max_{k=1,\cdots,K} f_k^*(\boldsymbol{x}) = \arg\max_{k=1,\cdots,K} \pi_k p_k(\boldsymbol{x}), \quad \forall \boldsymbol{x}, \forall \boldsymbol{\pi} \in A_k.$$

For any $\ell(\cdot)$, we define it's truncated loss at a location $s \leq 0$ by

$$l_{T_s}(\cdot) = \min(\ell(\cdot), \ell(s)).$$

The following theorem shows that the truncated loss ℓ_{T_s} is weighted Fisher-consistent.

Theorem 3.1. *Let $\ell(\cdot)$ be a nonincreasing loss function satisfying $\ell'(0^+) < \ell'(0^-) \leq 0$. Then a sufficient condition for the weighted truncated loss $\pi_y \ell_{T_s}(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y))$ with $K > 2$ and $s \leq 0$ to be weighted Fisher-consistent for estimating $\arg\max_j \pi_j p_j$ is that the truncation location s satisfies $\sup_{\{u:u \geq -s \geq 0\}} \frac{\dot{\ell}(0) - \dot{\ell}(u)}{\ell(s) - \ell(0)} \geq K - 1$. This condition is also necessary if $\ell(\cdot)$ is convex.*

Example 1. The hinge loss, $\ell(u) = [1-u]_+$: the condition becomes $s \in [-\frac{1}{K-1}, 0]$.

Example 2. The logistic loss, $\ell(u) = \log(1+e^u)$: the condition becomes $s \in [-\log(2^{K/(k-1)}-1), 0]$.

4 Review of Robust Model-Free Multiclass Probability Estimation

In this paper, linear SVM is used where $f(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{\beta} \rangle + \beta_0$.

4.1 Direct scheme for probability recovery

Define the truncated hinge loss as

$$H_{T_s}(u) = \min(H_1(s), H_1(u)),$$

where H_1 is hinge loss and $s = -\frac{1}{K-1}$ Denote $\hat{\boldsymbol{f}}^{\boldsymbol{\pi}}$ as the solution of the $\boldsymbol{\pi}$-weighted truncated hinge loss SVM, obtained from,

$$\min_{\boldsymbol{f}} \frac{1}{n} \sum_{i=1}^{n} \pi_{y_i} H_{T_s}(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)) + \lambda \sum_{k=1}^{K} J(f_k)$$
$$\text{subject to } \sum_{k=1}^{K} f_k(\boldsymbol{x}) = 0.$$

where $\boldsymbol{\pi} = (\pi_1, \cdots, \pi_K) \in A_K$. By Theorem 3.1, we have that $\arg\max_k \hat{f}_k^{\boldsymbol{\pi}}$ converges to $\max_k pi_k p_k$ as $n \to \infty$ and $\lambda \to 0$. The following proposition gives a key result for estimating the probabilities for each $\boldsymbol{x} \in S$.

Theorem 4.1. *For any given $\boldsymbol{x} \in S$ satisfying $\min_k p_k(\boldsymbol{x}) > 0$, there exists a unique weight vector $\tilde{\boldsymbol{\pi}}(\boldsymbol{x}) = (\tilde{\pi}_1(\boldsymbol{x}), \cdots, \tilde{\pi}_K(\boldsymbol{x})) \in A_K$ such that*

$$\tilde{\pi}_1(\boldsymbol{x}) p_1(\boldsymbol{x}) = \tilde{\pi}_2(\boldsymbol{x}) p_2(\boldsymbol{x}) = \cdots = \tilde{\pi}_K(\boldsymbol{x}) p_k(\boldsymbol{x}).$$

We can estimate $p_j(\boldsymbol{x})$ from the estimation of $\tilde{\boldsymbol{\pi}}(\boldsymbol{x})$.

Theorem 4.2. *For any given $\boldsymbol{x} \in S$, we assume that its associated border weight is estimated as $\hat{\boldsymbol{\pi}}(\boldsymbol{x})$. Then its class probability estimated as*

$$\hat{p}_k(\boldsymbol{x}) = \frac{\hat{\pi}_k(\boldsymbol{x})^{-1}}{\sum_{i=1}^{K} \pi_i(\boldsymbol{x})^{-1}}, \quad k = 1, \cdots, K$$

This estimation of probability is consistent.

Theorem 4.3. *For any non-increasing lossw function $\ell(\cdot)$ with $\ell'(0) < 0$, if the truncation location s is chosen such that $\sup_{\{u: u \geq -s \geq 0\}} \frac{\ell(0)-\ell(u)}{\ell(s)-\ell(0)} \geq K-1$. When $\lambda \to 0$ and the grid size $d_\pi \to 0$ as $n \to \infty$, estimate $\hat{p}_k(\boldsymbol{x})$ based on the truncated loss ℓ_{T_s} is asymptotically consistent, i.e.,*

$$\hat{p}_k(\boldsymbol{x}) \to p_k(\boldsymbol{x}) \text{ for } k = 1, \cdots, K \text{ as } n \to \infty.$$

5 Matrix predictor SVM

Suppose we have train data set such as $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_N, y_N)$ where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$ for all i. We have generalized optimization problem with $f(x) = \langle \boldsymbol{x}, \boldsymbol{\beta} \rangle + \beta_0$,

$$\min_{\beta_0, \boldsymbol{\beta}} \sum_{i=1}^{N} L(y_i, f(\boldsymbol{x}_i)) + \frac{\lambda}{2} \|\boldsymbol{\beta}\|^2,$$

where $L(y, f(x))$ is a loss function. We can characterize several loss functions in terms of what they are estimating at the population level. Table 1 shows several loss functions and corresponding minimization functions on population level.

Loss Function	$L(y, f(x))$	Minimization Function
Binomial Deviance	$\log(1 + e^{-yf(x)})$	$f(x) = \log \frac{P(Y=+1\|x)}{P(Y=-1\|x)}$
SVM Hinge Loss	$[1 - yf(x)]_+$	$f(x) = \text{sign}[P(Y = +1\|x) - \frac{1}{2}]$
Squared Error	$[1 - yf(x)]^2$	$f(x) = 2P(Y = +1\|x) - 1$

Table 1: The population minimizers for the different loss functions.

Consider we have a test data set such as $(X_1, y_1), \cdots, (X_N, y_N)$ where $X_i \in \mathbb{R}^{m \times n}$ and $y_i \in \{-1, 1\}$ for all i. My proposal to train this kind of matrix predictors dataset is to define

$$f(x) = \langle X, B \rangle + \beta_0 \quad \text{where } B \in \mathcal{B} = \{B : \text{rank}(B) \leq r\}.$$

and solve the optimization problem,

$$\min_{\beta_0, B} \sum_{i=1}^{N} L(y_i, f(\boldsymbol{x}_i)) + \frac{\lambda}{2} \|B\|_F^2 .$$

The rank r can be chosen as $\max_{i=1,\cdots,N} \text{rank}(X_i)$. This rank constrained space implies the belief that predictor space has low rank structure. This constraint also functions as reducing the number of parameters to estimate.

6 Discussion and to do list

- Comparison between kernel SVM and new matrix preserving linear SVM (benefit of using linear SVM with matrix predictor?)
- Algorithm derivation in matrix case.
- Extension to multiclass case (similar to Section 4).

References

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Yichao Wu, Hao Helen Zhang, and Yufeng Liu. Robust model-free multiclass probability estimation. *Journal of the American Statistical Association*, 105(489):424–436, 2010.