Support Matrix Machine Review

Chanwoo Lee, May 16, 2020

1 Linear SMM

1.1 Model

Assume that we have training data $\{(X_1, y_1)\}_{i=1}^N$ where $X_i \in \mathbb{R}^{m\times n}$ is matrix valued predictor and $y_i \in \{-1, +1\}$ is its corresponding class label. To make the model consider matrix structure, we think the following formulation:

$$
\begin{aligned}
(P) \qquad\qquad & \min_{B,b,\boldsymbol{\xi}} && \frac{1}{2}\|B\|^2 + C\sum_{i=1}^N \xi_i \\
& \text{subject to} && y_i(\langle B, X_i\rangle + b) \geq 1 - \xi_i, \\
& && \xi_i \geq 0, \quad i = 1, \cdots, N.
\end{aligned} \tag{1}
$$

Here we use $\|\cdot\|$ as Frobenius matrix norm. If the coefficient matrix $B \in \mathbb{R}^{m\times n}$ has full rank, (1) is reduced to regular SVM with predictor $\text{Vec}(X_i)$. To capture matrix structure, we assume that the matrix B has a low rank $r < \min(m, n)$ so that

$$
B = UV^T \quad \text{ where } U \in \mathbb{R}^{m\times r} \text{ and } V \in \mathbb{R}^{n\times r}.
$$

Then, the equation (1) becomes,

$$
\begin{aligned}
(P) \qquad\qquad & \min_{U,V,b,\boldsymbol{\xi}} && \frac{1}{2}\|UV^T\|^2 + C\sum_{i=1}^N \xi_i \\
& \text{subject to} && y_i(\langle UV^T, X_i\rangle + b) \geq 1 - \xi_i, \\
& && \xi_i \geq 0, \quad i = 1, \cdots, N.
\end{aligned} \tag{2}
$$

We see that the solution SMM function is

$$
\begin{aligned}
\boldsymbol{f}(X;\Theta) &= \langle UV^T, X\rangle + b \quad \text{ where } \Theta = (U, V, b) \\
&= \sum_{i=1}^N \alpha_i y_i \langle H_U X_i H_V, X\rangle + b \\
&= \sum_{i=1}^N \alpha_i y_i \langle H_U X_i, X\rangle + b \\
&= \sum_{i=1}^N \alpha_i y_i \langle X_i H_V, X\rangle + b,
\end{aligned} \tag{3}
$$

where $H_A = A(A^TA)^{-1}A^T$ is a projection matrix. The last three equalities can be obtained from dual solution in the next section. Therefore, we estimate our classifier as

$$
\text{sign}\left(\boldsymbol{f}(X;\hat{\Theta})\right), \quad \text{where } \hat{\Theta} = (\hat{U}, \hat{V}, \hat{b}) \text{ is an optimizer of (2).}
$$

1.2 Algorithm

We optimize (2) using coordinate descent algorithm that solves for one factor fixing the other factors. For each update of matrices U and V, we use quadratic programming with dual problem where strong duality holds. The following is the primal and dual problem in each update.

1. When fixing V,

$$
\begin{aligned}
(P_u) \qquad & \min_{U,b,\boldsymbol{\xi}} \quad \frac{1}{2}\|UV^T\|^2 + C\sum_{i=1}^{N}\xi_i \\
& \text{subject to} \quad y_i(\langle UV^T, X_i\rangle + b) \geq 1 - \xi_i, \\
& \qquad\qquad\quad \xi_i \geq 0, \quad i = 1, \cdots, N.
\end{aligned}
$$

$$
\begin{aligned}
(D_u) \qquad & \max_{\boldsymbol{\alpha}\in\mathbb{R}^m:\boldsymbol{\alpha}\geq 0} \left(\sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \langle X_i, X_j H_V\rangle\right) \\
& \text{subject to} \quad \sum_{i=1}^{N} y_i\alpha_i = 0, \\
& \qquad\qquad\quad 0 \leq \alpha_i \leq C, \quad i = 1.\cdots, N,
\end{aligned}
$$

   where $H_V = V(V^TV)^{-1}V^T$. We have the optimizer $U = \sum_{i=1}^{N}\alpha_i y_i X_i V(V^TV)^{-1}$.

2. When fixing U,

$$
\begin{aligned}
(P_v) \qquad & \min_{V,b,\boldsymbol{\xi}} \quad \frac{1}{2}\|UV^T\|^2 + C\sum_{i=1}^{N}\xi_i \\
& \text{subject to} \quad y_i(\langle UV^T, X_i\rangle + b) \geq 1 - \xi_i, \\
& \qquad\qquad\quad \xi_i \geq 0, \quad i = 1, \cdots, N.
\end{aligned} \tag{4}
$$

$$
\begin{aligned}
(D_v) \qquad & \max_{\boldsymbol{\alpha}\in\mathbb{R}^m:\boldsymbol{\alpha}\geq 0} \left(\sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \langle X_i, H_U X_j\rangle\right) \\
& \text{subject to} \quad \sum_{i=1}^{N} y_i\alpha_i = 0, \\
& \qquad\qquad\quad 0 \leq \alpha_i \leq C, \quad i = 1.\cdots, N,
\end{aligned}
$$

   where $H_U = U(U^TU)^{-1}U^T$. We have the optimizer $V^T = \sum_{i=1}^{N}\alpha_i y_i (U^TU)^{-1}U^T X_i$.

Based on the above argument, we summarize the SMM algorithm in Algorithm 1.

Algorithm 1: SMM algorithm

Input: $(X_1, y_1), \cdots, (X_N, y_N)$, rank r

Parameter: U,V

Initizlize: $U^{(0)}, V^{(0)}$

Do until converges

Update U fixing V :

Solve $(D_u) : \max_{\boldsymbol{\alpha}} \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \langle X_i, X_j H_V \rangle$.

$U = \sum_{i=1}^N \alpha_i y_i X_i V (V^T V)^{-1}$.

Update V fixing U :

Solve $(D_v) : \max_{\boldsymbol{\alpha}} \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \langle X_i, H_U X_j \rangle$.

$V = \sum_{i=1}^N \alpha_i y_i X_i^T U (U^T U)^{-1}$.

Output: $B = UV^T$

1.3 SMM conditional probability calculation

To calculate conditional probability, we solve the regularization problem with weighted hinge loss.

$$\begin{aligned} \min_{U,V,b,\boldsymbol{\xi}} \frac{1}{2}\|UV^T\|^2 + C\left[(1-\pi)\sum_{y_i=1}\xi_i + \pi \sum_{y_i=-1}\xi_i\right] \\ \text{subject to } y_i\left(\langle UV^T, X_i\rangle + b\right) \geq 1 - \xi_i \\ \xi_i \geq 0, \quad i = 1, \ldots, N. \end{aligned} \tag{5}$$

We use coordinate descent approach for (5). For example, we update matrix U holding V fixed from the following dual problem.

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N\sum_{j=1}^N \alpha_i\alpha_j y_i y_j \langle X_i H_V, X_j H_V\rangle, \\ \text{subject to } 0 \leq \alpha_i \leq C(1-\pi) \text{ for } y_i = 1, \\ 0 \leq \alpha_i \leq C\pi \text{ for } y_i = -1, \\ \sum_{i=1}^N \alpha_i y_i = 0. \end{aligned}$$

Let $\hat{\Theta}_\pi = (\hat{U}_\pi, \hat{V}_\pi, \hat{b}_\pi)$ be the solution of (5). One can repeatedly solve (5) using different π, for example, $0 = \pi_1 < \cdots < \pi_{m+1} = 1$. We estimate conditional probability $\mathbb{P}(y = 1|X)$ as,

$$\mathbb{P}(y=1|X) = \frac{1}{2}\left(\arg\max_{\pi_j}\{\text{sign}(\boldsymbol{f}(X|\hat{\Theta}_{\pi_j})) = 1\} + \arg\max_{\pi_j}\{\text{sign}(\boldsymbol{f}(X|\hat{\Theta}_{\pi_j})) = -1\}\right).$$

2 Non linear SMM

2.1 Model

We fit the SM classifier using input feature $\{(\boldsymbol{h}(X_i), y_i)\}_{i=1}^N$ where $\boldsymbol{h} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m'\times n}$ $(m < m')$ and produce the (nonlinear) classifier $\hat{\boldsymbol{f}}(X) = \text{sign}\left(\langle \hat{U}\hat{V}^T, \boldsymbol{h}(X)\rangle + \hat{b}\right)$.

Our objective primal problem for nonlinear case is

$$\min_{U\in\mathbb{R}^{m'\times r},V\in\mathbb{R}^{n\times r},\boldsymbol{\xi}} \frac{1}{2}\|UV^T\|^2 + c\sum_{i=1}^{N}\xi_i \tag{6}$$
$$\text{subject to } y_i(\langle UV^T, \boldsymbol{h}(X_i)\rangle + b) \leq 1-\xi_i$$
$$\xi_i \geq 0, \quad i=1,\dots,N.$$

We define kernel matrix $\boldsymbol{K} : \mathbb{R}^{m\times n}\times\mathbb{R}^{m\times n}\to\mathbb{R}^{n\times n}$ as

$$\boldsymbol{K}(X,X') = \boldsymbol{h}(X)^T\boldsymbol{h}(X').$$

From (3), we see that the solution function $\boldsymbol{f}(X)$ can be written

$$\begin{aligned}\boldsymbol{f}(X) &= \langle UV^T, \boldsymbol{h}(X)\rangle + b\\ &= \sum_{i=1}^{N}\alpha_i y_i \mathrm{tr}\left(H_V\boldsymbol{h}(X)^T\boldsymbol{h}(X_i)\right) + b\\ &= \sum_{i=1}^{N}\alpha_i y_i \mathrm{tr}\left(H_V\boldsymbol{K}(X,X_i)\right) + b.\end{aligned}$$

We estimate the classifier as

$$\mathrm{sign}\left(\sum_{i=1}^{N}\hat{\alpha}_i y_i \mathrm{tr}\left(H_{\hat{V}}\boldsymbol{K}(X,X_i)\right) + \hat{b}\right)$$

We will show that optimizing (6) requires only knowledge of the kernel function $\boldsymbol{K}$ in the next section.

2.2 Algorithm

We take coordinate descent approach to optimize (6).

1. When V fixed, we have the following dual problem from (6).

$$\min_{\boldsymbol{\alpha}} -\sum_{i=1}^{N}\alpha_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathrm{tr}\left(H_V\boldsymbol{K}(X_i,X_j)\right) \tag{7}$$
$$\text{subject to } \sum_{i=1}^{N} y_i\alpha_i = 0$$
$$0\leq\alpha_i\leq C, \quad i=1,\dots,N.$$

We obtain updated U formula with the optimizer $\boldsymbol{\alpha}$ in (7) as

$$U = \sum_{i=1}^{N}\alpha_i y_i \boldsymbol{h}(X_i)V(V^TV)^{-1} \tag{8}$$

where $\boldsymbol{h}$ function is not known. We borrow this formula to update V in the next step.

2. When U fixed, we have the following dual problem from (6).

$$\min_{\boldsymbol{\beta}} -\sum_{i=1}^{N}\beta_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\beta_i\beta_j y_i y_j \langle H_U \boldsymbol{h}(X_i), H_U \boldsymbol{h}(X_j)\rangle \tag{9}$$
$$\text{subject to } \sum_{i=1}^{N} y_i\beta_i = 0$$
$$0 \le \beta_i \le C, \quad i = 1, \dots, N.$$

To get an optimal $\boldsymbol{\beta}$ in (9), we need the information of $\langle H_U h(X_i), H_U h(X_j)\rangle$. Notice

$$\begin{aligned}\langle H_U \boldsymbol{h}(X_i), H_U \boldsymbol{h}(X_j)\rangle &= \operatorname{tr}\left(H_U \boldsymbol{h}(X_j)\boldsymbol{h}(X_i)^T\right) = \operatorname{tr}\left(U(U^TU)^{-1}U^T \boldsymbol{h}(X_j)\boldsymbol{h}(X_i)^T\right) \\ &= \operatorname{tr}\left((U^TU)^{-1}U^T \boldsymbol{h}(X_j)\boldsymbol{h}(X_i)^T U\right)\end{aligned} \tag{10}$$

Using the (8), we have the following expressions for the components in (10), which only related to the kernel $\boldsymbol{K}$.

$$\begin{aligned} U^TU &= \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (V^TV)^{-1}V^T \boldsymbol{h}(X_i)^T \boldsymbol{h}(X_j) V (V^TV)^{-1} \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (V^TV)^{-1}V^T \boldsymbol{K}(X_i, X_j) V (V^TV)^{-1} \\ &= (V^TV)^{-1}V^T \left(\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \boldsymbol{K}(X_i, X_j)\right) V (V^TV)^{-1}, \end{aligned} \tag{11}$$

$$\begin{aligned} U^T\boldsymbol{h}(X_j) &= \sum_{l=1}^{N}\alpha_l y_l (V^TV)^{-1}V^T \boldsymbol{h}(X_l)^T \boldsymbol{h}(X_j) \\ &= \sum_{l=1}^{N}\alpha_l y_l (V^TV)^{-1}V^T \boldsymbol{K}(X_l, X_j) \\ &= (V^TV)^{-1}V^T \sum_{l=1}^{N}\alpha_l y_l \boldsymbol{K}(X_l, X_j). \end{aligned}$$

Therefore, we can find the optimizer $\boldsymbol{\beta}$ using only the knowledge of the kernel $\boldsymbol{K}$ and the last update of V. We update new V as

$$V^T = \sum_{i=1}^{N}\beta_i y_i (U^TU)^{-1}U^T \boldsymbol{h}(X_i).$$

by plugging the formula (11).

2.3 Kernel matrix validity check

Definition 1. We call the matrix kerenl $\boldsymbol{K} : \mathbb{R}^{m\times n}\times\mathbb{R}^{m\times n} \to \mathbb{R}^{n\times n}$ valid if there exists a feature mapping $\boldsymbol{h} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m'\times n}$ such that

$$\boldsymbol{K}(X, X') = \boldsymbol{h}(X)^T\boldsymbol{h}(X') \in \mathbb{R}^{n\times n} \quad \text{for any matrices } X, X' \in \mathbb{R}^{m\times n}.$$

Here is a sufficient condition for the valid matrix kernel.

Theorem 2.1 (Sufficient Condition)**.** *For a given matrix kernel $\boldsymbol{K}$,suppose that there exists vector kernel K such that*

$$[\boldsymbol{K}(X, X')]_{i,j} = K(X_{\cdot i}, X'_{\cdot j}),$$

where $X_{\cdot i}$ is i-th column of the matrix X. Then, the kernel is valid.

Proof. Let $h : \mathbb{R}^m \to \mathbb{R}^{m'}$ be a feature mapping corresponding to vector kernel K such that

$$K(\boldsymbol{x}, \boldsymbol{x'}) = \langle h(\boldsymbol{x}), h(\boldsymbol{x'})\rangle \quad \text{for any } \boldsymbol{x}, \boldsymbol{x'} \in \mathbb{R}^m.$$

Define a matrix feature mapping $\boldsymbol{h} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m'\times n}$ as

$$\boldsymbol{h}(X) = (h(X_{\cdot 1}), \cdots , h(X_{\cdot n})) .$$

Then, we have the following equality.

$$\begin{aligned}
[\boldsymbol{h}(X)^T\boldsymbol{h}(X')]_{ij} &= \left[(h(X_{\cdot 1}), \cdots , h(X_{\cdot n}))^T \left(h(X'_{\cdot 1}), \cdots , h(X'_{\cdot n})\right)\right]_{ij} \\
&= h(X_{\cdot i})^T h(X'_{\cdot j}) = \langle h(X_{\cdot i}), h(X'_{\cdot j})\rangle \\
&= K(X_{\cdot i}, X'_{\cdot j}).
\end{aligned} \tag{12}$$

Equation (12) implies $\boldsymbol{K}(X, X') = \boldsymbol{h}(X)^T\boldsymbol{h}(X')$ which proves the theorem. □

Necessary condition for the valid kernel is as follows.

Theorem 2.2. *Supple $\boldsymbol{K} : \mathbb{R}^{m\times n} \times \mathbb{R}^{m\times n} \to \mathbb{R}^{n\times n}$ is a function that takes as input a pair of matrices and produces a matrix. Let $\{X_i : \mathbb{R}^{m\times n} : i \in [N]\}$ denote a set of input matrices, and let $\mathcal{K}$ denote an order-4, (N, N, n, n)-dimensional tensor,*

$$\mathcal{K} = [\![\mathcal{K}(i, i', p, p')]\!], \quad \textit{where } \mathcal{K}(i, i', p, p') \textit{ is the } (p, p') - th \textit{ entry of the matrix } \boldsymbol{K}(X_i, X_{i'}).$$

Then, the factorization $\boldsymbol{K}(X_i, X_{i'}) = \boldsymbol{h}(X_i)^T\boldsymbol{h}(X_{i'})$ exists for some mapping $\boldsymbol{h}$, only if both of the following condition hold:

1. *For every index $i \in [N]$, the matrix $\boldsymbol{K}(i, i, :, :) \in \mathbb{R}^{n\times n}$ is positive semidefinite.*
2. *For every index $p \in [n]$, the matrix $\boldsymbol{K}(:, :, p, p) \in \mathbb{R}^{N\times N}$ is positive semidefinite.*

Proof. 1. Let $i \in [N]$ be a fixed index. For any vector $\boldsymbol{a} \in \mathbb{R}^d$,

$$\boldsymbol{a}^T\mathcal{K}(i, i, :, :)\boldsymbol{a} = \boldsymbol{a}^T\boldsymbol{h}(X_i)^T\boldsymbol{h}(X_i)\boldsymbol{a} = \langle \boldsymbol{h}(X_i)\boldsymbol{a}, \boldsymbol{h}(X_i)\boldsymbol{a}\rangle = \|\boldsymbol{h}(X_i)\boldsymbol{a}\|^2 \geq 0.$$

2. Let $p \in [n]$ be a fixed index. We use $[\cdot]_{(k,p)}-$th entry of the matrix. For any vector $\boldsymbol{b} \in \mathbb{R}^n$,

$$\begin{aligned}
\boldsymbol{b}^T\mathcal{K}(:, :, p, p)\boldsymbol{b} &= \sum_{ij} b_i b_j [\boldsymbol{h}(X_i)^T\boldsymbol{h}(X_j)]_{(p,p)} \\
&= \sum_{ij} b_i b_j \sum_k [\boldsymbol{h}(X_i)]_{(k,p)}[\boldsymbol{h}(X_j)]_{(k,p)}
\end{aligned}$$

$$= \sum_k \left(\sum_i [\boldsymbol{h}(X_i)]_{(k,p)} b_i \right)^2 \geq 0.$$

□

Remark 1. There are some matrix valued kernel that satisfies sufficient conditions.

$$\begin{aligned}
\text{Linear:} \quad & K(X, X') = X^T X' \\
\text{Polynomial:} \quad & K(X, X') = \underbrace{(X^T X' + \mathbb{1}_n \mathbb{1}_n^T) \circ \cdots \circ (X^T X' + \mathbb{1}_n \mathbb{1}_n^T)}_{d-\text{times}} \\
\text{Radial:} \quad & [K(X, X')]_{ij} = \exp\left(-\|X_{\cdot i} - X_{\cdot j}\|^2/\sigma\right),
\end{aligned}$$

where $\circ$ is hadamard product.