

# SMMK modification

Chanwoo Lee, May 17, 2020

## 1 SMMK symmetric adjustment

The current SMM kernel treats  $X$  and  $X^T$  differently. To address this, we use

$$X^* = \begin{pmatrix} 0 & X^T \\ X & 0 \end{pmatrix} \quad \text{or} \quad \tilde{X}^* = \begin{pmatrix} 0 & X \\ X^T & 0 \end{pmatrix} \in \mathbb{R}^{(m+n) \times (m+n)},$$

as input data matrix. First, we show that the output boundary of the SMMK is invariant over whether we use  $X^*$  or  $\tilde{X}^*$ .

**Theorem 1.1.** *Suppose that for a given matrix kernel  $\mathbf{K}$  there exists vector valued kernel  $K$  that satisfies (the sufficient condition for valid kernel)*

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

*Also assume that the kernel  $K$  is symmetric in the sense that*

$$K(\mathbf{a}, \mathbf{b}) = K(\pi(\mathbf{a}), \pi(\mathbf{b})),$$

*where  $\mathbf{a}, \mathbf{b} \in \mathbb{R}^m$ , and  $\pi$  is an arbitrary permutation function. Then SMMK functions from input  $X^*$  and  $\tilde{X}^*$  have the same classification.*

*Proof.* Let  $\mathbf{h}$  be the corresponding feature map to the kernel  $\mathbf{K}$  such that

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

Consider the following SMMK primal problem.

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

It is enough to show that there exists permutation function  $\boldsymbol{\pi} : \mathbb{R}^{d \times (m+n)} \rightarrow \mathbb{R}^{d \times (m+n)}$  such that

$$\boldsymbol{\pi}(\mathbf{h}(X_i^*)) = \mathbf{h}(\tilde{X}_i^*) \text{ for all } i = 1, \dots, N. \tag{2}$$

Because if (2) holds, then the equation (1) can be expressed as

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

which is the primal problem from input  $\tilde{X}^*$ .

Notice that there exists vector valued feature mapping  $h$  such that

$$\mathbf{h}(X_i^*) = (h([X_i^*]_1), h([X_i^*]_2), \dots, h([X_i^*]_{m+n})),$$

where  $[\cdot]_i$  is the  $i$ -th column of the matrix. Let  $\pi_1$  be the column-wise permutation such that

$$\pi_1(\mathbf{h}(X_i^*)) = ((h([X_i^*]_{n+1}), \dots, h([X_i^*]_{n+m}), h([X_i^*]_1), \dots, h([X_i^*]_n))$$

By the symmetricity of the vector valued kernel  $K$ , the corresponding feature map  $h$  has the property that

$$\{h(\mathbf{a})_i : i = 1, \dots, d\} = \{h(\pi(\mathbf{a}))_i : i = 1, \dots, d\},$$

for any  $\mathbf{a} \in \mathbb{R}^{m+n}$  and permutation  $\pi : \mathbb{R}^{m+n} \rightarrow \mathbb{R}^{m+n}$ . To be specific, the permutation does not change components of the image of  $h$ . Therefore, there exist row-wise permutations  $\pi_{2,j}$  such that

$$[\pi_1(\mathbf{h}(X_i^*))]_j = \pi_{2,j}([\mathbf{h}(\tilde{X}_i^*)]_j).$$

By defining  $\pi_2 = (\pi_{2,1}, \dots, \pi_{2,m+n})$ , we have  $\pi = \pi_2 \circ \pi_1$  that satisfies (2).  $\square$

In addition, we can show that SMMK with  $X^*$  has the same classifier with  $X$  in the linear case.

**Theorem 1.2.** *In the linear SMM case, the classifier  $\mathbf{f}(X; \hat{B}) = \langle \hat{B}, X \rangle + \hat{b}$  form the cost  $C$  in the primal problem is the same as the classifier of  $\mathbf{f}(X^*; \hat{B}^*) = \langle \hat{B}^*, X^* \rangle + \hat{b}$  with the half cost  $\frac{1}{2}C$ . To*

*be specific,  $\hat{B}^* = \begin{pmatrix} 0 & \hat{B}^T \\ \hat{B} & 0 \end{pmatrix}$*

**Conclusion also holds under rank-constraints:  
rank  $r$  with  $X \iff$  rank  $2r$  with  $X^*$**

*Proof.* The primal problem of linear SMM is

$$(P) \quad \min_{B, b, \xi} \quad \frac{1}{2} \|B^*\|^2 + C \sum_{i=1}^N \xi_i \quad (3)$$

subject to  $y_i(\langle B^*, X_i^* \rangle + b) \geq 1 - \xi_i,$   
 $\xi_i \geq 0, \quad i = 1, \dots, N.$

Consider the following equality

$$\begin{aligned} \langle B^*, X^* \rangle &= \left\langle \begin{pmatrix} B_1 & B_2 \\ B_3 & B_4 \end{pmatrix}, \begin{pmatrix} 0 & X^T \\ X & 0 \end{pmatrix} \right\rangle \\ &\stackrel{\text{rank}}{=} \left\langle \begin{pmatrix} 0 & B_2 \\ B_3 & 0 \end{pmatrix}, \begin{pmatrix} 0 & X^T \\ X & 0 \end{pmatrix} \right\rangle \\ &= \langle B_2^T + B_3, X \rangle \quad \leftarrow \text{rank (B2) + rank (B3)} \\ &= \langle B, X \rangle \quad \text{where } B = B_2^T + B_3. \end{aligned}$$

The second equality holds because  $B_1 = B_4 = 0$ . since  $B_1$  and  $B_4$  do not affect the constraint. Then, (3) becomes

$$(P) \quad \min_{B, b, \xi} \quad \frac{1}{4} \|B\|^2 + C \sum_{i=1}^N \xi_i \quad (4)$$

$$\begin{aligned} \text{subject to } & y_i(\langle B, X_i \rangle + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, \dots, N. \end{aligned}$$

In (4), we use the inequality that  $\|B^*\|^2 = \|B_2\|^2 + \|B_3\|^2 \geq \frac{1}{2}\|B_2 + B_3\|^2 = \frac{1}{2}\|B\|^2$  where equality holds when  $B_2 = B_3^T$ . Assume the images  $h(X)$  and  $h(X^*)$  span the same space  $\square$

this should be a numerical issue, not a statistical fact...

In the nonlinear case, we can not guarantee that the output boundaries from  $X$  and  $X^*$  are the same. The following is simple illustration of different boundary in the vector case SMMK. We have training data  $(X_1, y_1), \dots, (X_{40}, y_{40})$  where  $X_i \in \mathbb{R}^2$  and  $y_i \in \{-1, +1\}$ . I trained the data with input  $X \in \mathbb{R}^{2 \times 1}, X^T \in \mathbb{R}^{1 \times 2}, X^* \in \mathbb{R}^{4 \times 4}$  and  $\tilde{X}^* \in \mathbb{R}^{4 \times 4}$ . Figure 1 shows the SMMK boundary with polynomial kernel on each input. We can notice that the case of  $X^*$  and  $\tilde{X}^*$  have the same output, which is verified in Theorem 1.1. In addition, the symmetric adjustment boundary is in the middle between boundaries with  $X$  and  $X^T$ . This makes the boundary fluctuate less and stable. We can think symmetric adjustment as improvement from vector case kernel method which has steep fluctuation in subfigure A.

## 2 Simulation code

```

1 makesym = function(mat){
2   m = nrow(mat); n = ncol(mat)
3   nmat = rbind(cbind(matrix(0,n,n),t(mat)),cbind(mat,matrix(0,m,m)))
4   return(nmat)
5 }
6
7
8 x <- matrix(rnorm(30*2), ncol = 2)
9 y <- c(rep(-1,15), rep(1,15))
10 x[y==1,] <- x[y==1,] + 3/2
11 dat <- data.frame(x=x, y=as.factor(y))
12 colnames(dat) = c("x1", "x2", "y")
13 ggplot(data = dat, aes(x1, x2, colour = y)) + geom_point()
14 X = lapply(seq_len(nrow(x)), function(i) matrix(x[i,,drop = F], 2, 1))
15 nX = lapply(X, makesym)
16 tX = lapply(X, t)
17 ntX = lapply(tX, makesym)
18
19 lresult1 = smmk(X, y, 1, polykernel)
20 lresult2 = smmk(nX, y, 3, polykernel) rank for expanded predictor: at most 2
21 lresult3 = smmk(tX, y, 1, polykernel)
22 lresult4 = smmk(ntX, y, 3, polykernel) Adjust C?
23
24 lpredict1 = lresult1$predict
25 lpredict2 = lresult2$predict
26 lpredict3 = lresult3$predict
27 lpredict4 = lresult4$predict
28
29 x1_seq = seq(min(x[,1]), max(x[,1]), length.out = 100)
30 x2_seq = seq(min(x[,2]), max(x[,2]), length.out = 100)
31 xgrid = expand.grid(x1_seq, x2_seq)
32 ylgrid1 = apply(xgrid, 1, function(x) lpredict1(matrix(x, nrow = 2)))
33 grid1 = as.data.frame(cbind(xgrid, ylgrid1))
34 colnames(grid1) = c("x1", "x2", "y")
35 plt1 = ggplot(data = grid1, aes(x = x1, y = x2, colour = as.factor(y))) + geom_point(
36   size = 0.01) +
  
```

Q: which one has better cost? A or C/2?  
A: Almost similar after adjusting C and rank.

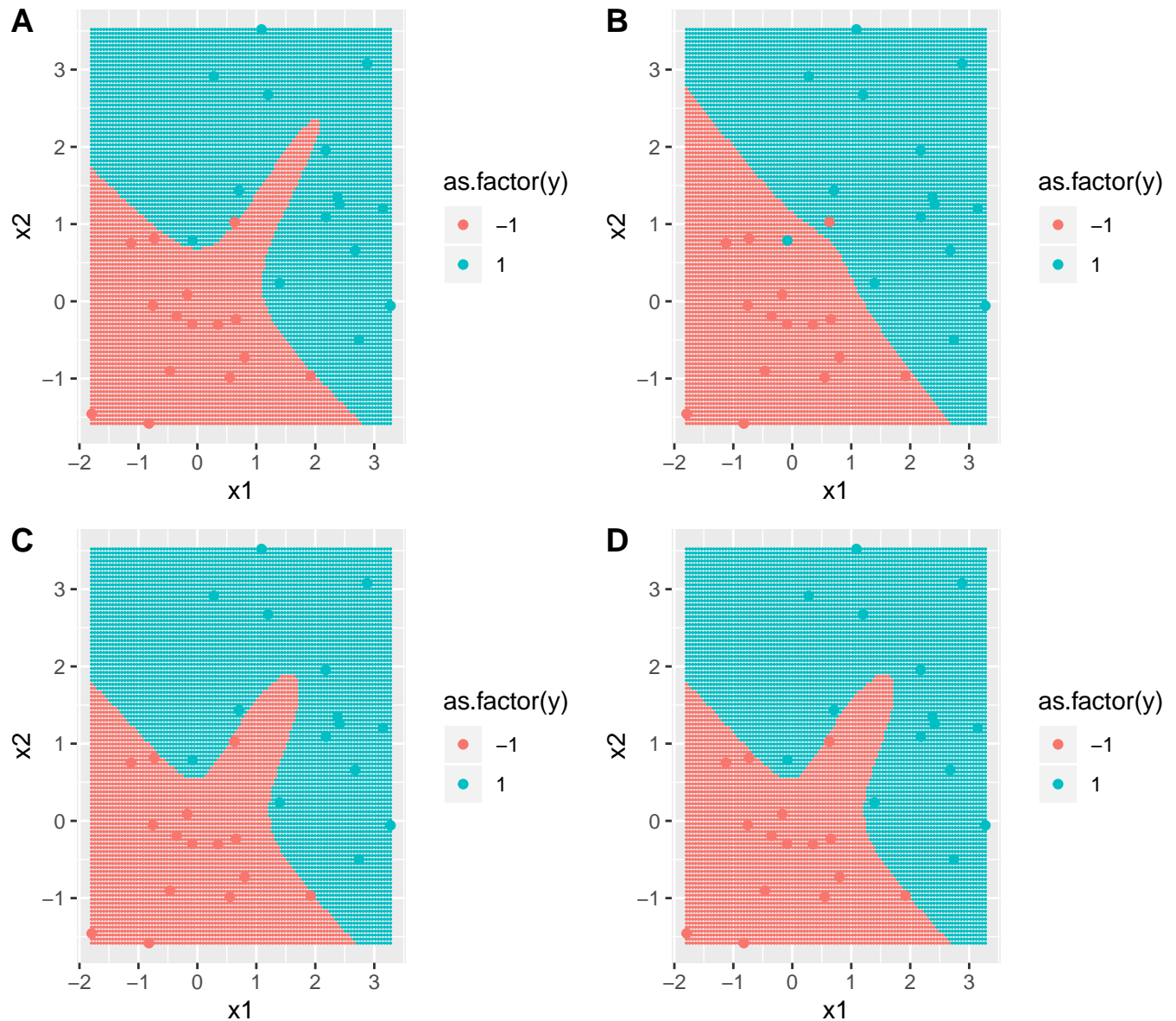


Figure 1: Subfigure A shows the SMMK boundary with input  $X \in \mathbb{R}^{2 \times 1}$ , B with input  $X^T \in \mathbb{R}^{1 \times 2}$ , C and D with  $X^*, \tilde{X}^* \in \mathbb{R}^{4 \times 4}$  respectively. Big dots in each figure is from the training data.

```

37
38 ylgrid2 = apply(xgrid,1,function(x) lpredict2(makesym(matrix(x,nrow = 2))))
39 grid2 = as.data.frame(cbind(xgrid,ylgrid2))
40 colnames(grid2) = c("x1","x2","y")
41 plt2 = ggplot(data = grid2, aes(x = x1,y=x2,colour = as.factor(y)))+geom_point(
42     size = 0.01)+
43     geom_point(data = dat,aes(x = x1,y = x2,colour = as.factor(y)))
44
45 ylgrid3 = apply(xgrid,1,function(x) lpredict3(matrix(x,ncol = 2)))
46 grid3 = as.data.frame(cbind(xgrid,ylgrid3))
47 colnames(grid3) = c("x1","x2","y")

```

```

48 plt3 = ggplot(data = grid3, aes(x = x1,y=x2,colour = as.factor(y)))+geom_point(
    size =0.01)+
49   geom_point(data =dat,aes(x = x1,y = x2,colour = as.factor(y)))
50
51 ylgrid4 = apply(xgrid,1,function(x) lpredict4(makesym(matrix(x,ncol = 2))))
52 grid4 = as.data.frame(cbind(xgrid,ylgrid4))
53 colnames(grid4) = c("x1","x2","y")
54 plt4 = ggplot(data = grid4, aes(x = x1,y=x2,colour = as.factor(y)))+geom_point(
    size =0.01)+
55   geom_point(data =dat,aes(x = x1,y = x2,colour = as.factor(y)))
56
57 length(which(ylgrid4==ylgrid2))
58
59 library(ggpubr)
60 ggarrange(plt1, plt3, plt2,plt4,labels = c("A", "B","C","D"),ncol = 2, nrow = 2)
61
62
63 length(which(ylgrid1==ylgrid2))

```