

# Classification algorithm based on matrix kernels

Miaoyan Wang, Aug 20, 2020

## 1 Notation

1.  $\mathbb{O}(d, r) := \{\mathbf{P} \in \mathbb{R}^{d \times r} : \mathbf{P}^T \mathbf{P} = \mathbf{I}_r\}$ , the collection of  $d$ -by- $r$  matrices whose columns are orthonormal. When no confusion arises, I use the term “projection matrix” to denote either the matrix  $\mathbf{P}\mathbf{P}^T \in \mathbb{R}^{d \times d}$  or the matrix  $\mathbf{P} \in \mathbb{R}^{d \times r}$ .
2.  $\mathcal{K}^{\text{row}}(i, j, \mathbf{X}, \mathbf{X}') := \langle \Phi(\mathbf{X}_{i:}), \Phi(\mathbf{X}'_{j:}) \rangle$  denotes the value of row kernel evaluated at the vector pair, ( $i$ -th row of matrix  $\mathbf{X}$ ,  $j$ -th row of matrix  $\mathbf{X}'$ ).
3. I sometimes use the shorthand  $\mathcal{K}^{\text{row}}(i, j)$  to denote  $\mathcal{K}^{\text{row}}(i, j, \mathbf{X}, \mathbf{X}')$ , when the feature pair  $(\mathbf{X}, \mathbf{X}')$  is clear given the contexts. Note that  $\mathcal{K}^{\text{row}}(i, j)$  can be calculated without explicit feature mapping. Similar convention for  $\mathcal{K}^{\text{col}}(i, j, \mathbf{X}, \mathbf{X}')$ .
4. Let  $\mathbf{W}^{\text{row}} = \mathbf{P}_r \mathbf{P}_r^T = \llbracket w_{ij}^{\text{row}} \rrbracket \in \mathbb{R}^{d_1 \times d_1}$  and  $\mathbf{W}^{\text{col}} = \mathbf{P}_c \mathbf{P}_c^T = \llbracket w_{ij}^{\text{col}} \rrbracket \in \mathbb{R}^{d_2 \times d_2}$  denote the row- and column-wise projection matrices, respectively.

## 2 Algorithm based on bilinear maps

Consider the bilinear mapping,

$$\begin{aligned} \Phi: \mathbb{R}^{d_1 \times d_2} &\rightarrow (\mathcal{H}_r \times \mathcal{H}_c)^{d_1 \times d_2} \\ \mathbf{X} &\mapsto [\Phi(\mathbf{X})_{ij}], \quad \text{where } \Phi(\mathbf{X})_{ij} \stackrel{\text{def}}{=} (\phi_c(\mathbf{X}_{i:}), \phi_r(\mathbf{X}_{:j})). \end{aligned}$$

Primal problem:

$$\begin{aligned} \min_{\mathbf{P}_r, \mathbf{P}_c} \min_{\mathbf{C}} \quad & \frac{1}{2} \|\mathbf{C}\|_F^2 + c \sum_{i=1}^n \xi_i, \\ \text{subject to} \quad & y_i \langle \mathbf{P}_r \mathbf{C} \mathbf{P}_c^T, \Phi(\mathbf{X}_i) \rangle \leq 1 - \xi_i \text{ and } \xi_i \geq 0, \quad i = 1, \dots, n. \end{aligned} \tag{1}$$

Parameters in the primal problem:  $(\mathbf{P}_r, \mathbf{P}_c, \mathbf{C})$ , where  $\mathbf{P}_r \in \mathbb{O}(d_1, r_1)$ ,  $\mathbf{P}_c \in \mathbb{O}(d_2, r_2)$ , and  $\mathbf{C} = \llbracket (\mathbf{c}_i^{\text{row}}, \mathbf{c}_j^{\text{col}}) \rrbracket \in (\mathcal{H}_r \times \mathcal{H}_c)^{r_1 \times r_2}$  is the “core matrix” consisting of linear coefficients.

1. Update  $\mathbf{C}$ , given  $(\mathbf{P}_r, \mathbf{P}_c)$ .

$$\text{implicit update } \mathbf{C} \leftarrow \sum_i \alpha_i y_i \mathbf{P}_r^T \Phi(\mathbf{X}_i) \mathbf{P}_c.$$

- Saved quantities: dual variables  $\alpha \in \mathbb{R}^n$ .
- Auxiliary quantities: kernel  $\mathcal{K}(\mathbf{X}, \mathbf{X}')$  specified below.
- Objective value: objective value for the dual problem.

Details: We use kernel trick to solve for  $\alpha$  without explicit feature mapping. Given the projections  $(\mathbf{P}_r, \mathbf{P}_c)$ , the optimization (1) is a standard SVM with kernel

$$\begin{aligned}\mathcal{K}(\mathbf{X}, \mathbf{X}') &= \langle \mathbf{P}_r^T \Phi(\mathbf{X}) \mathbf{P}_c, \mathbf{P}_r^T \Phi(\mathbf{X}') \mathbf{P}_c \rangle \\ &= \left( \sum_{i,j} w_{ij}^{\text{col}} \right) \left( \sum_{i,j} w_{ij}^{\text{row}} K^{\text{row}}(i, j) \right) + \left( \sum_{i,j} w_{ij}^{\text{row}} \right) \left( \sum_{i,j} w_{ij}^{\text{col}} K^{\text{col}}(i, j) \right).\end{aligned}\quad (2)$$

for all feature pairs  $(\mathbf{X}, \mathbf{X}')$ . Here I have used the shorthand  $K^{\text{row}}(i, j)$  to denote the value of row kernel evaluated on the  $i$ -th row of  $\mathbf{X}$  and  $j$ -th row of  $\mathbf{X}'$ .

**Remark 1** (Computational consideration). We can compute the summations in (2) without explicit loop. In particular, both identities hold:  $\sum_{i,j} w_{ij}^{\text{col}} = \|\mathbf{1}^T \mathbf{P}_c\|_2^2$  and  $\sum_{i,j} w_{ij}^{\text{row}} K^{\text{row}}(i, j) = \text{trace}(\mathbf{W}^T \mathbf{K})$ , where  $\mathbf{K} \leftarrow \llbracket K^{\text{row}}(i, j, \mathbf{X}, \mathbf{X}') \rrbracket$  is a pre-stored matrix (or array, if we go through all possible feature pairs  $(\mathbf{X}, \mathbf{X}')$ ).

2. Update  $\mathbf{P}_r^{\text{new}}$ , given  $(\mathbf{C}, \mathbf{P}_c)$ .

explicit update  $\tilde{\mathbf{P}}_r^{\text{new}} \leftarrow \sum_i \beta_i y_i \Phi(\mathbf{X}_i) \mathbf{P}_c \mathbf{C}^T = \sum_{i,j} \beta_i \alpha_j y_i y_j \underbrace{\Phi(\mathbf{X}_i) \mathbf{P}_c \mathbf{P}_c^T \Phi^T(\mathbf{X}_j)}_{d_1\text{-by-}d_1 \text{ matrix over } \mathbb{R}} \mathbf{P}_r$

normalize  $\mathbf{P}_r^{\text{new}} \leftarrow \text{QR decomposition of } \tilde{\mathbf{P}}_r^{\text{new}}$ .

- Saved quantities:  $\mathbf{P}^{\text{new}} \in \mathbb{O}(d_1, r_1)$ .
- Auxiliary quantities: matrix  $\Phi(\mathbf{X}_i) \mathbf{P}_c \mathbf{P}_c^T \Phi^T(\mathbf{X}_j)$ , dual variables  $\beta \in \mathbb{R}^n$ .
- Objective value: objective value for the dual problem.

Details: for each feature pair  $(i, j) \in [n]^2$ , we compute the matrix  $\Phi(\mathbf{X}_i) \mathbf{P}_c \mathbf{P}_c^T \Phi^T(\mathbf{X}_j)$  without explicit feature mapping,

$$\begin{aligned}\underbrace{\Phi(\mathbf{X}_i) \mathbf{P}_c \mathbf{P}_c^T \Phi^T(\mathbf{X}_j)}_{=: \mathbf{M}} &= \left( \sum_{s,s'} w_{ss'}^{\text{col}} \right) \begin{bmatrix} K^{\text{row}}(1, 1, \mathbf{X}_i, \mathbf{X}_j) & \cdots & K^{\text{row}}(1, d_1, \mathbf{X}_i, \mathbf{X}_j) \\ \vdots & \vdots & \vdots \\ K^{\text{row}}(d_1, 1, \mathbf{X}_i, \mathbf{X}_j) & \cdots & K^{\text{row}}(d_1, d_1, \mathbf{X}_i, \mathbf{X}_j) \end{bmatrix} + \\ &\quad \left( \sum_{s,s'} w_{ss'}^{\text{col}} K^{\text{col}}(s, s', \mathbf{X}_i, \mathbf{X}_j) \right) \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix},\end{aligned}\quad (3)$$

where  $K^{\text{row}}(s, s', \mathbf{X}_i, \mathbf{X}_j)$  denotes the value of row kernel value evaluated on the  $s$ -th row of  $\mathbf{X}_i$  and  $s'$ -th row of  $\mathbf{X}_j$ , and likewise for  $K^{\text{col}}(s, s', \mathbf{X}_i, \mathbf{X}_j)$ .

The coefficient  $\beta$  is obtained from a standard SVM with kernel

$$\mathcal{K}(\mathbf{X}, \mathbf{X}') = \text{trace}(\underbrace{\Phi(\mathbf{X})\mathbf{P}_c\mathbf{C}^T}_{=: \mathbf{A}} \underbrace{(\mathbf{C}\mathbf{C}^T)^{-1}}_{=: \mathbf{B}} \underbrace{\mathbf{C}\mathbf{P}_c^T\Phi^T(\mathbf{X}')}_{=: \mathbf{A}^T}),$$

for feature pair  $(\mathbf{X}, \mathbf{X}')$ , where

$$\begin{aligned} \mathbf{B} &= \mathbf{C}\mathbf{C}^T = \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{P}_r^T \Phi(\mathbf{X}_i) \mathbf{P}_c \mathbf{P}_c^T \Phi^T(\mathbf{X}_j) \mathbf{P}_r, \\ \mathbf{A} &= \Phi(\mathbf{X}) \mathbf{P}_c \mathbf{C}^T = \sum_i \alpha_i y_i \Phi(\mathbf{X}) \mathbf{P}_c \mathbf{P}_c^T \Phi^T(\mathbf{X}_i) \mathbf{P}_r. \end{aligned}$$

are calculated using (3).

3. Update  $\mathbf{C}^{\text{new}}$ , given  $(\mathbf{P}_r^{\text{new}}, \mathbf{P}_c)$ .
4. Update  $\mathbf{P}_c$ , given  $(\mathbf{C}^{\text{new}}, \mathbf{P}_r^{\text{new}})$ ; denoted by  $(\mathbf{C}, \mathbf{P}_r)$  in the following formula.

explicit update  $\mathbf{P}_c^{\text{new}} \leftarrow \sum_i \gamma_i y_i \Phi^T(\mathbf{X}_i) \mathbf{P}_r \mathbf{C} = \sum_{i,j} \gamma_i \alpha_j y_i y_j \underbrace{\Phi^T(\mathbf{X}_i) \mathbf{P}_r \mathbf{P}_r^T \Phi(\mathbf{X}_j)}_{d_2\text{-by-}d_2 \text{ matrix over } \mathbb{R}} \mathbf{P}_c$

normalize  $\mathbf{P}_c^{\text{new}} \leftarrow$  QR decomposition of  $\tilde{\mathbf{P}}_c^{\text{new}}$ .

The auxiliary quantities,  $\Phi^T(\mathbf{X}_i) \mathbf{P}_r \mathbf{P}_r^T \Phi(\mathbf{X}_j)$  and  $\gamma$ , are calculated similarly as in step 2.

### 3 Outputs

1. Convergence criterum? Objective value in the dual problem.
2. How to read off the decision function from the algorithm?

$$\begin{aligned} f(\mathbf{X}_{\text{new}}) &= \langle \mathbf{P}_r^T \Phi(\mathbf{X}_{\text{new}}) \mathbf{P}_c, \sum_i \alpha_i y_i \mathbf{P}_r^T \Phi(\mathbf{X}_i) \mathbf{P}_c \rangle \\ &= \sum_i \alpha_i y_i \left\{ \left( \sum_{s,s'} w_{ss'}^{\text{col}} \right) \left( \sum_{s,s'} w_{ss'}^{\text{row}} K^{\text{row}}(s, s', \mathbf{X}_i, \mathbf{X}_{\text{new}}) \right) + \right. \\ &\quad \left. \left( \sum_{s,s'} w_{ss'}^{\text{row}} \right) \left( \sum_{s,s'} w_{ss'}^{\text{col}} K^{\text{col}}(s, s', \mathbf{X}_i, \mathbf{X}_{\text{new}}) \right) \right\}. \end{aligned}$$

3. How to estimate the intercept in the primal problem?

$$\hat{b}_0 = \arg \min_{b_0 \in \mathbb{R}} \left\{ \sum_{i=1}^n (1 - y_i f(\mathbf{X}_i) - y_i b_0)_+ \right\}.$$

4. Penalty in the primal problem?

$$\|C\|_F^2 = \sum_{i,j} \alpha_i \alpha_j y_i y_j \text{trace}(\Phi(\mathbf{X}_i) \mathbf{P}_c \mathbf{P}_c^T \Phi^T(\mathbf{X}_j) \mathbf{P}_r \mathbf{P}_r^T) = (\boldsymbol{\alpha} \circ \mathbf{y})^T \llbracket \mathcal{K}(\mathbf{X}_i, \mathbf{X}_j) \rrbracket (\boldsymbol{\alpha} \circ \mathbf{y}),$$

where  $\llbracket \mathcal{K}(\mathbf{X}_i, \mathbf{X}_j) \rrbracket \in \mathbb{R}^{n \times n}$  is defined in (2) and  $\boldsymbol{\alpha} \in \mathbb{R}^n$  is the dual parameter in step 1 or 3. We omit the explicit expression of  $\|C\|_F^2$  because it is not needed in our algorithm.

## 4 Sanity check

1. Monotonic decreasing in objective value? Yes.
2. Consistent with earlier linear kernel algorithm, smmk(...)? Yes.
3. Consistent with one-way linear mapping? Yes.

```
## generate data with linear kernel
source("SMMfunctions.R")
set.seed(1818)
m = 3; n = 3; r = 1; N = 100; b0 = 0.1
result = gendat(m,n,r,N,b0)
X = result$X; y = result$y; dat = result$dat; B = result$B
X_t=X ## transform
y_true=rep(0,N)
for(i in 1:N){
  X_t[[i]]=t(X[[i]])
  y_true[i]=sum(B*X[[i]])
}
svd(B)$v[,1] ## true col projection
svd(B)$u[,1] ## true row projection

## Algorithm 1: earlier algorithm
source("SMMKfunctions.R")
old= smmk(X,y,r=1,kernel = "linear");
old$V ## est col projection
old_t= smmk(X_t,y,r=1,kernel = "linear");
old_t$V ## est row projection

## Algorithm 2: one-way map. Separate row/column projection.
source("SMMKfunctions_miaoyan.R")
new = smmk_new(X,y,kernel_row="const",kernel_col="linear",
r=c(1,1),cost=10/3);
new$P_col ## est col projection; similar to earlier old$V
new = smmk_new(X,y,kernel_row="linear",kernel_col="const",
```

```

r=c(1,1),cost=10/3);
new$P_row ## est row projection; similar to earlier old_t$V

### Algorithm 2: two-way map. Simultaneous row/column projections.
new_double = smmk_new(X,y,kernel_row="linear",kernel_col="linear",
r=c(1,1),cost=10,rep=1);
plot(new_double$obj)
new_double$P_col ## est col projection
new_double$P_row ## est row projection
plot(new_double$fitted,y_true)

### Algorithm 2: two-way map + transform. Results should be consistent.
new_double = smmk_new(X_t,y,kernel_row="linear",kernel_col="linear",
r=c(1,1),cost=10,rep=1);
new_double$P_row ## est col projection
new_double$P_col ## est row projeciotn
plot(new_double$fitted,y_true)

```

Sanity check under linear kernel,  $d = 3$ ,  $r = 1$ ,  $N = 100$ .

Parameters	Truth	Symmetric trick (SMMK)	Bilinear map
col projection	(0.63, 0.33, 0.70)	(0.66, 0.33, 0.67)	(0.66, 0.33, 0.67)
			(0.70, 0.39, 0.58)
row projection	(-0.54, 0.83, 0.08)	(-0.55, 0.83, 0.06)	(-0.55, 0.82, 0.06)
			(-0.48, 0.87, 0.09)