SMM Kernel Validity Check

Chanwoo Lee, April 26, 2020

1 Kernel validity check

1.1 Sufficient condition of the Kernel

Define feature mapping $\boldsymbol{h} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m'\times n}$ where $m < m'$. Kernels for matrix case are defined as

$$\boldsymbol{K}(X, X') = \boldsymbol{h}(X)^T \boldsymbol{h}(X') \in \mathbb{R}^{n\times n}.$$

For a given kernel $\boldsymbol{K}$, sufficient condition to guarantee that there exists a feature mapping $\boldsymbol{h}$ follows

Theorem 1.1 (Sufficient Condition)**.** *For a given matrix kernel $\boldsymbol{K}$, there exists feature mapping $\boldsymbol{h} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m'\times n}$ such that*

$$\boldsymbol{K}(X, X') = \boldsymbol{h}(X)^T \boldsymbol{h}(X') \in \mathbb{R}^{n\times n}.$$

if 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.

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

$$K(\boldsymbol{x}, \boldsymbol{x}') = \langle h(\boldsymbol{x}), h(\boldsymbol{x}')\rangle.$$

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

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

Then, we have the following equality.

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

Equation (1) implies $\boldsymbol{K}(X, X') = \boldsymbol{h}(X)^T \boldsymbol{h}(X')$ which proves the theorem. □

Remark 1. From Theorem 1.1, I replace polynomial kernel and exponential kernel so that satisfy the sufficient condition.

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

where $\circ$ is hadamard product.

1.2 Quadratic programming validity

In updating U and V, we are using quadratic programming such that

$$\min_{\boldsymbol{\alpha}} -\sum_{i=1}^{N}\alpha_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathrm{tr}(V^T\boldsymbol{h}(X_i)^T\boldsymbol{h}(X_j)V), \tag{2}$$

$$\min_{\boldsymbol{\beta}} -\sum_{i=1}^{N}\beta_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\beta_i\beta_j y_i y_j \mathrm{tr}(U^T\boldsymbol{h}(X_i)\boldsymbol{h}(X_j)^TU),$$

where U and V are orthonormalized. We define matrix Q as

$$Q_{i,j} = \mathrm{tr}(V^T\boldsymbol{h}(X_i)^T\boldsymbol{h}(X_j)V).$$

We prove Q is positive semi definite matrix which implies that quadratic programming is valid in the first equation in (2). All the procedure is the same in the case of (2).

Theorem 1.2 (Positive semi definiteness for QP)**.** *Matrix Q defined above is positive semi definite.*

Proof. It suffices to show that $\boldsymbol{x}^TQ\boldsymbol{x}$ is positive for any $\boldsymbol{x} \in \mathbb{R}^N$. Denote $A_i = h(X_i)^TU$, then $Q_{ij} = \mathrm{tr}(A_i^TA_j)$. We have the following equation.

$$\begin{aligned}
\boldsymbol{x}^TQ\boldsymbol{x} &= \sum_{i,j} x_ix_j\mathrm{tr}(A_i^TA_j) \\
&= \sum_{i,j} x_ix_j\langle \mathrm{Vec}(A_i), \mathrm{Vec}(A_j)\rangle \\
&= \left(\sum_i x_i\,\mathrm{Vec}(A_i)\right)\left(\sum_i x_i\,\mathrm{Vec}(A_i)\right) \\
&= \left(\sum_i x_i\,\mathrm{Vec}(A_i)\right)^2 \geq 0.
\end{aligned}$$

□

2 Polynomial kernel simulation

I generate data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{80} \in \mathbb{R}^{1\times 2}$ randomly. Define a feature mapping $h : \mathbb{R}^{1\times 2} \to \mathbb{R}^{3\times 2}$ as follows.

$$h\left((z_1, z_2)\right) = \begin{pmatrix} 1 & 1 \\ \sqrt{2}z_1 & \sqrt{2}z_2 \\ z_1^2 & z_2^2 \end{pmatrix}.$$

The classification rule from feature data $\boldsymbol{x}$ is

$$y = \mathrm{sign}\left(\langle B, h(\boldsymbol{x})\rangle\right),$$

where $B = u_1v_1^T$, $u_1 \in \mathbb{R}^{6\times 1}$, and $v \in \mathbb{R}^{2\times 1}$. Notice that for arbitrary $\boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^{2\times 1}$, we have,

$$h(\boldsymbol{x})^T, h(\boldsymbol{z}) = (\boldsymbol{x}^T\boldsymbol{z} + \mathbb{1}_2\mathbb{1}_2^T) \circ (\boldsymbol{x}^T\boldsymbol{z} + \mathbb{1}_2\mathbb{1}_2^T) = K(\boldsymbol{x}, \boldsymbol{z}),$$

where K is a polynomial kernel with degree two. The simulation result shows that SMM with polynomial kernel perfectly fit the model while linear SMM classifies all data points as label -1.

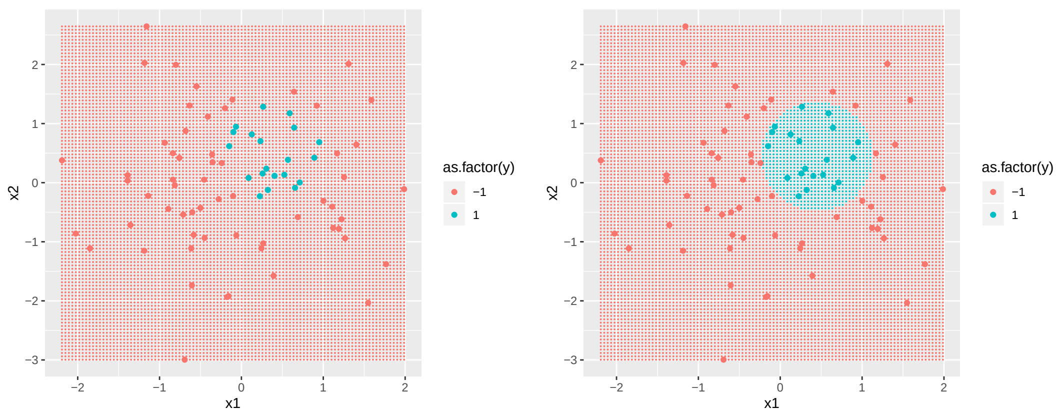

Figure 1: The left figure shows the linear SMM classification result which assign all points as label 1. The right figure shows classification boundary with SMM with polynomial kernel. Our algorithm succeeds to find ground truth boundary when used with right kernel.

3 Comparison simulation

3.1 Linearly separable data case

I generate random matrix $X_1, \dots, X_{100} \in \mathbb{R}^{10\times 8}$ whose entries are from i.i.d.Unif$(-1,1)$. I set ground truth matrix $B = UV^T \in \mathbb{R}^{10\times 8}$ such that $U \in \mathbb{R}^{10\times 3}, V \in \mathbb{R}^{8\times 3}$ whose entries are from i.i.d. Unif$(-1,1)$. Our classification rule is

$$y = \text{sign}(\langle B, X\rangle + 0.1).$$

Table 1 and 2 show linear methods outperform nonlinear one. In addition, SMM linear method outperforms SVM showing that SVM might have overfitting problem.

	1st	2nd	3rd	4th	5th	average
SVM	0.85	0.70	0.75	0.70	0.50	0.70
SMM	0.90	0.90	0.75	0.80	0.85	0.84
SMM(polynomial)	0.75	0.55	0.85	0.55	0.75	0.69
SMM(gaussian)	0.65	0.50	0.85	0.70	0.80	0.70

Table 1: Miss Classification Rate (MCR) on 5 folded Cross validation(CV)

	1st	2nd	3rd	4th	5th	average
SVM	1	1	1	1	1	1
SMM	1	1	1	1	1	1
SMM(polynomial)	1	1	1	1	1	1
SMM(gaussian)	1	1	1	1	1	1

Table 2: Training error on 5 folded CV

3.2 Linearly inseparable data case

I generate feature data matrix diversifying the variance and the number of data set. The detailed rules are as follow.

1. Sim 2.1: $N = 50$

$$\begin{aligned} \mathbb{P}\left((X_{\cdot 1}^T, X_{\cdot 2}^T)^T | y = 1\right) &\sim N((1,1,-1,-1)^T, 4I_4), \\ \mathbb{P}\left((X_{\cdot 1}^T, X_{\cdot 2}^T)^T | y = -1\right) &\sim N((0,0,0,0)^T, I_4). \end{aligned}$$

2. Sim 2.2: $N = 100$

$$\begin{aligned} \mathbb{P}\left((X_{\cdot 1}^T, X_{\cdot 2}^T)^T | y = 1\right) &\sim N((1,1,-1,-1)^T, 4I_4), \\ \mathbb{P}\left((X_{\cdot 1}^T, X_{\cdot 2}^T)^T | y = -1\right) &\sim N((0,0,0,0)^T, I_4). \end{aligned}$$

3. Sim 2.3: $N = 100$

$$\begin{aligned} \mathbb{P}\left((X_{\cdot 1}^T, X_{\cdot 2}^T)^T | y = 1\right) &\sim N((1,1,-1,-1)^T, I_4), \\ \mathbb{P}\left((X_{\cdot 1}^T, X_{\cdot 2}^T)^T | y = -1\right) &\sim N((0,0,0,0)^T, I_4). \end{aligned}$$

Figure 2 shows the generated data sets. Sim 2.3 is relatively easier to classify with linear function. Table 3 shows the averaged MCR on 5 folded CV. It shows that SMM with nonlinear kernels outperform when the data is hard to classify by linear function. Table 4 shows averaged MCR on each training set. As expected, the higher dimension has the better fitting on training data.

	sim 2.1	sim 2.2	sim 2.3
SVM	0.76	0.735	0.860
SMM	0.75	0.740	0.865
SMM(polynomial)	0.80	0.750	0.785
SMM(gaussian)	0.71	0.745	0.830

Table 3: This table shows the averaged MCR of 5 folded CV according to different methods.

	sim 2.1	sim 2.2	sim 2.3
SVM	0.80	0.780	0.868
SMM	0.77	0.763	0.865
SMM(polynomial)	0.87	0.786	0.848
SMM(gaussian)	0.92	0.876	0.903

Table 4: This table shows the averaged MCR on 5 training data sets according to different methods.

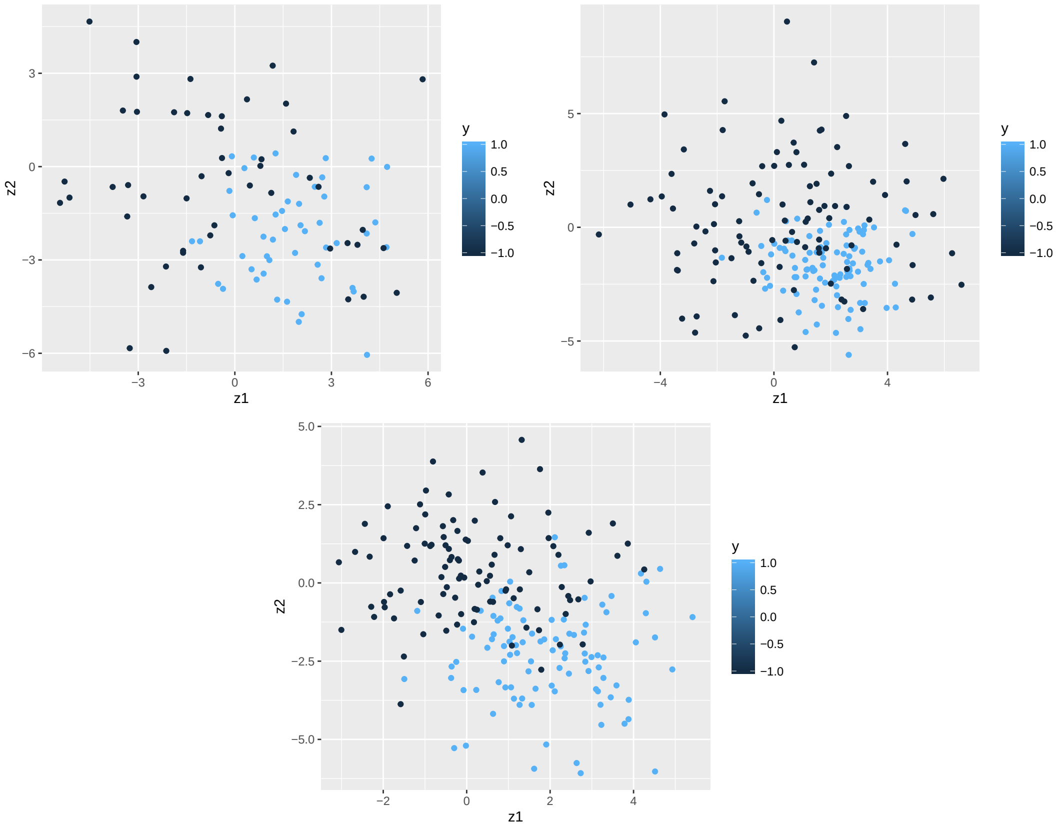

Figure 2: The figures represent from Sim 2.1 to Sim 2.3 in sequence.