Lasso comparison on simulation

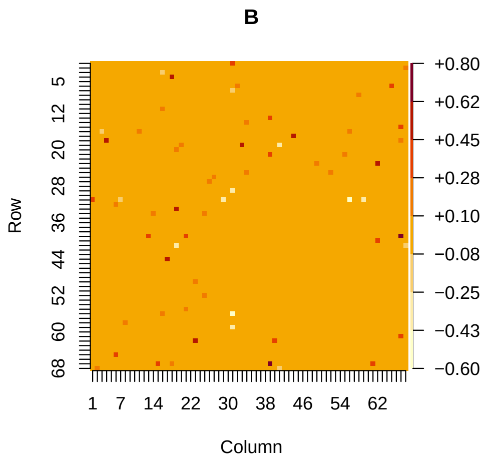
Chanwoo Lee, October 22, 2020

1 Lasso logistic matrix

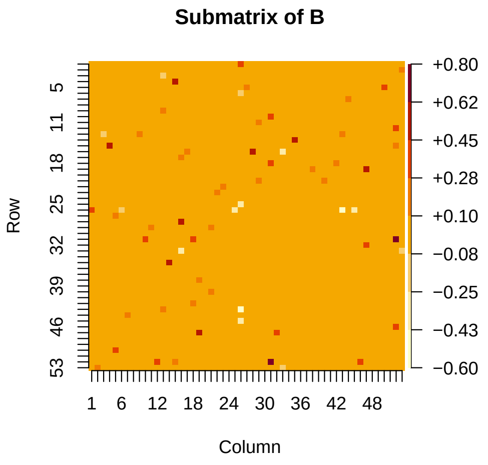
From lasso logistic regression method, I obtained matrix $\boldsymbol{B}_\ell$ such that

$$p(y_i = 1|\boldsymbol{X}_i) = \frac{\exp(b_0 + \langle \boldsymbol{B}_\ell, \boldsymbol{X}_i \rangle))}{1 + \exp(b_0 + \langle \boldsymbol{B}_\ell, \boldsymbol{X}_i \rangle)}, \quad i = 1, \ldots, n.$$

The coefficient matrix $\boldsymbol{B}_\ell$ has 96 non-zero entries. The sparsity of the matrix is 17 (has 52 non-zero columns) and the rank is 44. Figure 1 shows entries of the matrix.



(a) Full matrix $\boldsymbol{B}_\ell$.



(b) Submatrix of $\boldsymbol{B}_\ell$

Figure 1: (a) is the full coefficient matrix $\boldsymbol{B}_\ell$ while (b) is the submatrix of $\boldsymbol{B}_\ell$ where 0 columns and rows are deleted.

From this matrix, I run our algorithm with (rank, sparsity,weight) = (44,17,0.5) and compare with lasso logistic model in classification performance through cross validation. In addition, I decreased rank and compare with the lasso logistic model again. Table 1 shows the cross validation result on test datsets and training datasets. When the evaluation is based on misclassification rate, our model sometimes works better in prediction than lasso logistic model. The result shows that when we consider classification problem, we can find the hyper-parameters under which our model predicts better than the lasso model.

2 Improvement of SMMK algorithm

In SMMK algorithm, we consider sparsity structure in the last i.e. we calculate low rank matrix $\boldsymbol{B}$ first, and impose sparsity constraint choosing non zero columns and rows based on magnitude of $\text{diag}(\boldsymbol{B}^T\boldsymbol{B})$. Let $\boldsymbol{B}'$ be the matrix after imposed sparsity constraint, then we obtain approximated low rank matrix $\boldsymbol{B}''$ that close to $\boldsymbol{B}'$. However, the approximated $\boldsymbol{B}''$ is not the best low rank

	Lasso logistic	ADMM 1	SMMK 1	ADMM 2	SMMK 2	ADMM 3	SMMK 3
Test	0.377	0.340	0.302	0.363	0.321	0.395	0.392
Train	0.114	0	0	0	0	0.084	0.132

Table 1: Misclassification ratse on lasso logistic model and our model according to different algorithms. The numbering of each algorithm means different combinations of (rank, sparsity). 1: (rank, sparsity) = (44,17), 2: (rank, sparsity) = (10,17), and 3: (rank, sparsity) = (1,16)

matrix which has the sparse structure that minimizes the objective loss. From this perspective, we can re-run the algorithm after we choose which columns and rows are zero. To be specific, suppose that we have choosed the non-zero columns and rows and denote I as such index set. For the simplicity I assume that indices of non zero columns and rows are the same. Notice that

$$\langle \boldsymbol{B}, \boldsymbol{X}_i \rangle = \langle \boldsymbol{B}_I, [\boldsymbol{X}_i]_I \rangle,$$

where $\boldsymbol{M}_I$ is defined to be a sub matrix of $\boldsymbol{M}$. Therefore, we can find the best low rank r matrix $[\boldsymbol{B}'']_I$ solving the algorithm with reduced feature data input $([\boldsymbol{X}_1]_I, y_1), \cdots, ([\boldsymbol{X}_n]_I, y_n)$. Then, we can back original sized matrix $\boldsymbol{B}''$ adding zero columns and rows. One shortcoming of this approach is that it is time consuming because we run the algorithm two times: the first algorithm is to choose the non zero columns and rows while second algorithm is to find the best low rank matrix under the sparsity constraint. If we use the full rank in the first algorithm, then we can reduce the time because the first algorithm becomes quadratic programming. But the accuracy of choosing non-zero columns and rows are reduced. So it is good to use when the matrix is dense. I have checked that when (rank,sparsity) = (1,16), newly modified algorithm has 0.37 misclassification rate on test datasets while 0 misclassification on training datasets. Considering both SMMK and ADMM do not have perfect classification on training sets as one can see in Table 1 (combination 3), current algorithms can be improved more.

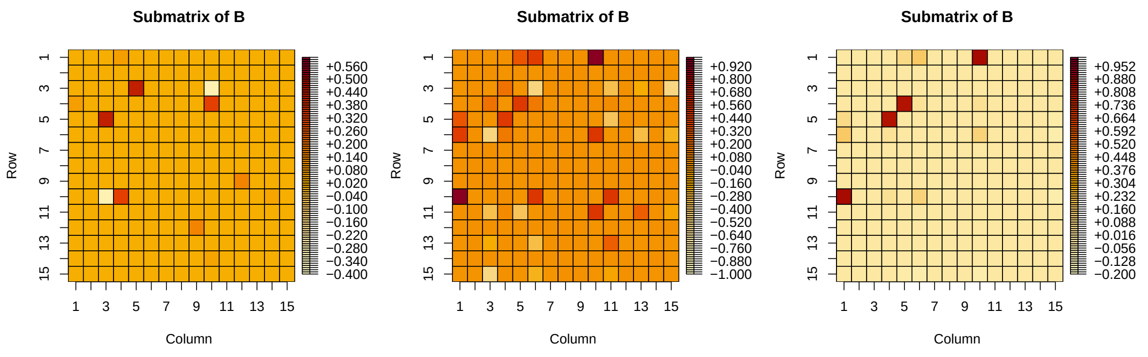
3 Comparison on reduced feature space

Here, I use reduced feature matrices $\boldsymbol{X}_i \in \mathbb{R}^{15\times 15}$ from VSPLOT brain datasets and choose the number of sample size as $n = 140$ to obtain fast results.

3.1 Classification

First, I obtain lasso coefficient $\boldsymbol{B}_\ell$ from 140 data points in VSPLOT brain dataset with reduced feature matrices. It turns out that the rank is 8 and the sparsity is 7. The number of non zero elements of $\boldsymbol{B}_\ell$ is 16. From this information, I obtain the coefficient matrix $\boldsymbol{B}$ from our algorithm with (rank,sparsity,weight) = (8,7,0.5). However, obtained $\boldsymbol{B}$ has much more dense than $\boldsymbol{B}_\ell$ so I increased the sparsity and assign full rank on given sparsity i.e. (rank, sparsity,weight) = (10,5,0.5). Figure 3 shows the obtained coefficient matrices.

The following table shows the averaged misclassification rates on test datasets in cross validation.



(a) (b) (c)

Figure 2: (a) is the full coefficient matrix $\boldsymbol{B}_\ell$. (b) is the full matrix of $\boldsymbol{B}$ when rank = 8, sparsity = 7, and weight = 0.5. (c) is the full matrix of $\boldsymbol{B}$ when rank = 5, sparsity = 10, and weight = 0.5

	Lasso logistic	ADMM (rank, sparsity, weight) = (8,7,0.5)	ADMM (rank, sparsity, weight) = (5,10,0.5)
Test dataset	0.44	0.45	0.37

Table 2: Misclassification rates in crosss validation on each method.

3.2 Probability estimation

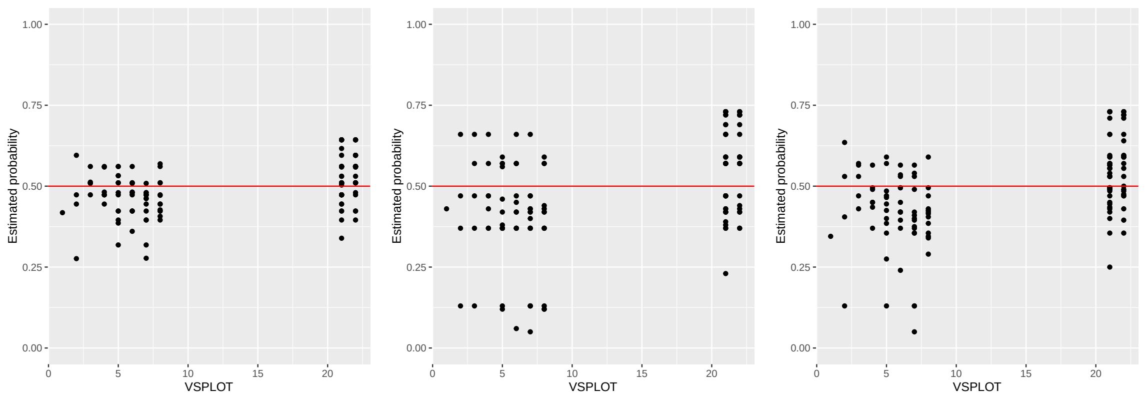
Here, I use ADMM algorithm and take two different approaches to estimate probability.

1. Option 1 is based on maximum cumulative probability
2. Option 2 is based on proposal in Biometrika (2008) Wang, shen & Liu.

First, the following figure is probability estimation results based on lasso logistic model and our model with different options and (rank,sparsity) = (5,10). Our model with Option 2 seems to outpeform others.

$$\begin{aligned}
\text{Logistic error (LE)} &: \sum_{y_i=1} \log\left(p(y_i=1|\boldsymbol{X}_i)\right) + \sum_{y_i=-1} \log\left(1-p(y_i=1|\boldsymbol{X}_i)\right) \\
\text{Squared error (SE)} &: \sum_i \left(y_i - p(y_i=1|\boldsymbol{X}_i)\right)^2 \\
\text{0-1 error (ZE)} &: \sum_{y_i=1} \mathbb{1}_{\{p(y_i=1|\boldsymbol{X}_i)<0.5\}} + \sum_{y_i=-1} \mathbb{1}_{\{p(y_i=1|\boldsymbol{X}_i)>0.5\}}
\end{aligned}$$

I evaluate the performance of each method using three different loss functions on test datsets. On our method, two combinations of hyper-parameters are used (rank,sparsity) = (8,7),(5,10). Option 1 and Option 2 are utilized to estimate the probabilities. Table 3 summarizes the cross validation results. In the perspectives of LE and SE, our method with Option 1 performs worse than random guess in all cases while Option 2 performs better when (rank, sparsity) = (5,10). Our approach outperforms random guess and Lasso logistic method based on ZE. So probability estimation based



(a) (b) (c)

Figure 3: The probability estimation result based on 140 samples with reduced feature matrices. (a) is based on lasso logistic regression. (b) and (c) are based on our model with (rank,sparsity) = (5,10). (b) uses Option 1 while (c) uses Option 2.

on Option 2 might lead to better hyper-parameter choices. I requested cross validation jobs on server to see the result on full feature matrices and sample sizes.

	Random guess	Lasso	Ours (8, 7, 1)	Ours (8, 7, 2)	Ours (5, 10, 1)	Ours (5, 10, 2)
LE	-19.40 (-78)	-19.31 (-74)	-22.29 (-71)	-21.39 (-71)	-19.92 (-72)	-19.08 (-71)
SE	7.00 (28)	6.95 (26)	7.84 (24)	7.46 (14)	7.13 (25)	6.79 (25)
ZE	0.5	0.44	0.43	0.42	0.4	0.41

Table 3: Cross validation results in probability estimation. ZE is scaled by the number of the test sets so that it is the same as misclassification rate. Our method has three different combinations (rank, sparsity, option). Parenthesis is averaged loss values on test datsets.