Exploration of heterogeneous treatment effects under distributed storage

Abstract

Exploration of heterogeneous treatment effects under distributed storage

Keywords: subgroup analysis, two-step algorithm, distributed storaged, ADMM

1. Introduction

With different industry life cycle stages, the pharmaceutical industry has also produced different medical theories. At present, based on the individualized characteristics of different patients, individualized medical theory will be an inevitable trend of the medical industry. Complex analytical tools are needed in personalized treatment strategies. One of the key statistical challenges is to correctly identify subgroups from heterogeneous populations. To address this issue, a popular approach is to use a mixture model analysis Everitt & Hand (1981), treating data as coming from different subgroups, each with its own parameter values. Farewell (1982) analysis the survival data with long-term survivors by mixture model. All along, the mixture model is continuous improvement. Muthén & Shedden (1999) discussed the analysis of an extended finite mixture model where the latent classes corresponding to the mixture components for one set of observed variables influence a second set of observed variables. Pauler & Laird (2000) introduced a general finite mixture of nonlinear hierarchical models that allows estimates of component membership probabilities and random effect distributions for longitudinal data arising from multiple subpopulations. Rasmussen (2000) presented the infinite gaussian mixture model. Maugis et al. (2009) discussed variable selection for clustering with gaussian mixture models. Shen & He (2015) proposed a structured logistic-normal mixture model for

the purpose of identifying a subgroup that has an enhanced treatment effect as well as the variables that are predictive of the subgroup membership. The mixture model-based approach needs to specify an underlying distribution and the number of mixture components in the population which is often difficult to do in practice. To solve this problem, Ma & Huang (2017) proposed a concave pairwise fusion approach to subgroup analysis. They developed an alternating direction method of multipliers algorithm with concave penalties. With the rapid development of data storage and communication, the medical industry has also developed a new direction. The data exchange between hospitals has improved the accuracy of diagnosis and treatment. So far, statistical research on subgroup analysis has remained at the stand-alone level. There is few approach to do the subgroup analysis over the distributed storaged data.

For distributed storage data, we can abstract the computing structure into two parts, one is the master and the other is the workers. All data interactions only occur between the master and the worker and no data communication between workers. The assumption of such a computing environment fits in with the operational logic of the actual distributed computing platform, and on the other hand, the data security of each node can also be guaranteed. more specifically, all sample data only exists in each worker node. There is no sample data in the master, which only do the map-reduce operation, and some necessary calculations after reduce data from all workers.

In this paper, We propose a two-step approach that allows us to minimize the influence of the limitations of data space separation and to perform subgroup analysis from an overall perspective. Let y_{im} be the response variable for the i^{th} subject in m^{th} node. $X_{im} = (x_{im1}, ..., x_{imp})$ presents a set of covariates. We consider subgroup analysis of the heterogeneity driven by unknown or unobserved latent factors. Hence, we consider

$$y_{im} = \mu_{im} + x_{im}^T \beta + \epsilon_{im}, i = 1, ..., n_m; m = 1, ..., M, \tag{1}$$

where μ_{im} is unknown subject-specific intercepts, $\beta = (\beta_1, ...\beta_p)^T$ is the vector of unknown coefficients for x_{im}, and ϵ_{im} is the error term independent of x_{im} with

$E(\epsilon_{im}) = 0$ and $Var(\epsilon_{im}) = \sigma^2$. Here we assume that $y_i m$ are from K different groups with $K \geq 1$ and the data from the same group have the same intercept. In other words, let $\mathcal{G} = (\mathcal{G}_1, ..., \mathcal{G}_K)$ be a partition of {11,...,$n_1$1,...,$n_m m$}. We have $\mu_{im} = \alpha_k$ for all $im \in \mathcal{G}_k$, where α_k is the common value for the μ_{im}'s from group $\mathcal{G}_k$. Model 1 can be divided into two parts, combined with the actual situation of biomedicine, x_{im} represents some patient-related essential variables, such as age, gender, etc. μ_{im} represents the factors contributing to the heterogeneity, such as different treatments. Then μ_{im} can be written as $\mu_{im} = \mu + z_{im}^T \theta$, where z_{im} reprensents the different treatment effects. It is worth noting that we are talking about personalized medicine, which means different patients have different effects on the same treatment $\mu_{im} = \mu + z_{im}^T \theta_{im}$. Thus, model 1 becomes

$$y_{im} = \mu + z_{im}^T \theta_{im} + x_{im}^T \beta + \epsilon_{im}, i = 1, ..., n_m; m = 1, ..., M. \tag{2}$$

Throughout this paper, we focus on model2 by considering that even the samples of one subgroup are distributed storaged, our estimation method can still correctly identify their subgroups. Several authors have studied the problem of exploring homogeneity effects of covariates over a single machine. Ma & Huang (2017) proposed a concave pairwise fusion penalized least squares approach for this purpose and derive an alternating direction method of multipliers algorithm Boyd et al. (2011) for implementing the following approach.

$$Q_n(\mu, \beta; \lambda) = \frac{1}{2} \sum_{i=1}^{n} (y_i - \mu_i - X_i^T \beta)^2 + \sum_{1 \leq i < j \leq n} P(|\mu_i - \mu_j|, \lambda), \tag{3}$$

where $P(\cdot, \lambda)$ is a concave penalty function with a tuning parameter $\lambda \geq 0$. However, when we introduce the node information, the objective function 3 is very difficult to solve. The objective function becomes as follow.

$$Q_n(\mu, \beta; \lambda) = \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{n_m} (y_{im} - \mu_{im} - X_{im}^T \beta)^2 + \sum_{1 \leq i < j \leq \sum_m n_m} P(|\mu_i - \mu_j|, \lambda). \tag{4}$$

Compared with function 3, function 4 brings a lot of problems that make the original estimation method malfunction. One problem is that The operation

on matrix X becomes unrealizable. Another problem is $|\mu_i - \mu_j|, 1 \leq i < j \leq \sum_m n_m$ needs a lot of data interaction, almost impossible in actual operation.

In large-scale data clustering, sampling is an efficient and most widely used approximation technique. Neyman (1934) elaborated on two sampling methods: the method of stratified sampling and the method of purposive selection. Recently, large-scale data analysis is a challenging and relevant task for present-day research and industry. Many statisticians use stratified sampling to subtly solve the problem of large data classification. Cervellera & Macciò (2018) analyzed the technique of stratified sampling from the point of view of distances between probabilities, and introduced an algorithm, based on recursive binary partition of the input space, aimed at obtaining samples that are distributed as much as possible as the original data. Zhao et al. (2019) proposed a stratified sampling based clustering algorithm for large-scale data. Therefore, we proposed a two-step fusion penalized algorithm based on ADMM algorithm and stratified sampling. Through the operation of map-reduce and two-step iteration, our algorithm can accurately identity the subgroups of data in each physical node with low computational cost.

The rest of this paper is organized as follows. In Section 2 we describe the two-step approach in detail. In Section 3 we solve the two-step approach by ADMM algorithm. In Section4, we do some numerical simulation in the spark cluster mode, compared with the calculation results when the data is stored in a single machine. In Section 5, the real data is distributed at different physical nodes, mimicking the hospital data interoperability.

2. Two-step Algorithm over distributed storaged data

For estimate model1, the objective function of the concave pairwise fusion penalized least squares approach is

$$Q_n(\mu, \beta; \lambda) = \frac{1}{2} \sum_{m=1}^{M} \sum_{i=1}^{n_m} (y_{im} - \mu_{im} - X_{im}^T \beta)^2 + \sum_{1 \leq i < j \leq \sum_m n_m} P(|\mu_i - \mu_j|, \lambda).$$

By the idea of ADMM algorithm, we can introduce a new set of parameters to separate the penalty function. But no matter how we introduce the parameters, an problem is that a matrix operation of $X(X^TX)^{-1}X^T$ can not be avoid in updating u_{im} and β. From a medical perspective, β presents the common coefficients for essential variables. This incidates that if the data of a single hospital database is sufficient, maybe we can estimate β accurately with X_m. However, the study of homogeneity, that is, the estimation of u_{im}, requires a comprehensive consideration of hospital data in various regions. Intuitively, the patient is rare for some hospital, but there may be sufficient sample references added up in all hospital data. For such patients, if the data of all hospitals can be comprehensively considered, the subgroup analysis will be more accurate. The essential reason why ADMM can not be used directly to solve the objective function is that it is difficult to calculate the oparation of super-large matrix X. So here we consider stratified sampling, which reduces the observation sample matrix, making the calculation possible. Now the problem is how stratified sampling to preserve homogeneous distribution.

In this paper, we propose a two-step fusion penalized algorithm. Firstly, we apply fusion penalized model in each node to get the hierarchy. The hierarchy obtained by this method can preserve the homogeneous distribution. The objective function in each node is

$$Q_{nm}(\mu_m, \beta_m; \lambda) = \frac{1}{2}\sum_{i=1}^{n_m}(y_{im} - \mu_{im} - X_{im}^T\beta_m)^2 + \sum_{1\le i<j\le n_m} P(|\mu_{im} - \mu_{jm}|, \lambda), \tag{5}$$

where $\mu = (\mu_{1m}, ..., \mu_{n_m m})$, m presents the index of node, and $P(\cdot, \lambda)$ is a concave penalty function with a tunning parameter $\lambda > 0$. In model 5, the penalty can shrinks some of $\mu_{im} - \mu_{jm}$ to zero, so that we can partition observations into subgroups. The tunning parameter λ needs to trades off the loss and penalty to get meaningful solution. Our approach is running the algorithm on a grid of λ_s in a decrease order with warm-start and choosing the one that minimizes certain selection rule, such as AIC, BIC, etc. As for the penalty function, we choose those who can produce unbiased estimates, which is SCAD proposed by

Fan & Li (2001) and MCP proposed by Zhang et al. (2010). These concave penalties enjoy the sparsity as the L1 penalty that it can automatically yield zero estimates.

Secondly, we run stratified sampling based on the subgroup analysis of each node, then reduce the samples to the master and apply the fusion penalty approach. We take proportional allocation strategy in stratified sampling, and the size of the sample in each stratum is taken in proportion to the size of the stratum. After sampling, we run the fusion penalty approach to the samples on master, the objective function is

$$Q_n(\mu, \beta; \lambda) = \frac{1}{2}\sum_{i=1}^{n}(y_i - \mu_i - X_i^T\beta)^2 + \sum_{1\leq i<j\leq n} P(|\mu_i - \mu_j|, \lambda). \qquad (6)$$

Overall, we state the details of two-step fusion penalty in algorithm 1

Algorithm 1 Two-step fusion penalty

Input:

The set of observation covariates X_m at each node. The set of response variable y_m for the subjects at each node.

Output:

Subject-specific intercepts μ_{im} of each observation in model 1.

Common coefficients β for X in model 1.

1: Computing $\mu_m, \beta_m : \arg\min\{Q_{nm}(\mu_m, \beta_m; \lambda)\}$ at each node.

2: Reducing α_k and $\mathcal{G}$ from each node and the sampling number n_k of layer k is $\frac{|\mathcal{G}_k|}{\sum_{k=1}^{K}\mathcal{G}_{\|}}$.

3: Extracting n_k samples X'_k in $\mathcal{G}_k$ at each node for $k = 1, ..., K$ and reducing to master.

4: Computing $\mu', \beta' : \arg\min\{Q_n(\mu, \beta; \lambda)\}$ at master based on X'.

5: Let $\beta = \beta'$ for each X_{im} and y_{im} at m^{th} node, $\mu_{im} = \arg\min_{\mu_{im}\in\mu'}(y_{im} - \mu_{im} - X_{im}^T\beta)^2$.

6: **return** μ_{im} and β.

3. Computation

The estimation tast in minimization 5 is basically equal to minimization 6. The only difference is that 5 applies to local data of each node and 6 applies to the reduced data at master. So we only discuss how to get the solution of minimization 6, which is

$$\mu, \beta = \arg\min_{\mu,\beta} \left\{ \frac{1}{2} \sum_{i=1}^{n} (y_i - \mu_i - X_i^T \beta)^2 + \sum_{1 \le i < j \le n} P(|\mu_i - \mu_j|, \lambda) \right\} \quad (7)$$

The unconstraint problem 7 can be recast as

$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2} \textstyle\sum_{i=1}^{n} (y_i - \mu_i - X_i^T \beta)^2 + \sum_{1 \le i < j \le n} P(|\eta_{ij}|, \lambda) \\ \text{subject to} \quad & \eta_{ij} = \mu_i - \mu_j \end{aligned} \quad (8)$$

The augmented Lagrangian function Boyd et al. (2011) of 8 is

$$\begin{aligned} L_{a_{11},\ldots,a_{1n},\ldots,a_{nn}} = & \frac{1}{2} \sum_{i=1}^{n} (y_i - \mu_i - X_i^T \beta)^2 + \sum_{1 \le i < j \le n} P(|\eta_{ij}|, \lambda) \\ & + \sum_{1 \le i < j \le n} (\frac{a_{ij}}{2} \|\mu_i - \mu_j - \eta_{ij}\| + < \mu_i - \mu_j - \eta_{ij}, v_{ij} >), \end{aligned} \quad (9)$$

where v_{ij} are the Lagrangian multipliers related to equality constraint $\eta_{ij} = \mu_i - \mu_j$ for $1 \le i < j \le n$, and a_{ij} are the corresponding positive penalties parameters. The algorithm for solving 9 updates the primal variables μ, β, η via block coordinate descent in a closed form and updates dual variables v_{ij} via one step gradient ascent. we state the details of this algorithm in algorithm **??**.

Algorithm 2 Minimize $Q_n(\mu, \beta; \lambda)$

Input: Fix the values of λ, a_{ij}, maximum iteration number $\mathrm{It}_{\max}$, and initial values of μ_{ij}^0,β^0,η_{ij}^0 and v_{ij}^0.

Output: β and μ

1: **for** $k = 0, 1..., \mathrm{It}_{\max}$. **do**

2: For given $\eta^{(k)}$ and $v^{(k)}$, updating $\mu^{(k+1)}$ and $\beta^{(k+1)}$

$$\mu^{(k+1)} = \arg\min_{\mu} L(\mu, \beta, \eta^{(k)}, v^{(k)}) \tag{10}$$

and

$$\beta^{(k+1)} = \arg\min_{\beta} L(\mu^{(k+1)}, \beta, \eta^{(k)}, v^{(k)}) \tag{11}$$

3: Updating $\eta^{(k+1)}$ based on different penalty function

$$\eta^{(k+1)} = \arg\min_{\eta} L(\mu^{(k+1)}, \beta^{(k+1)}, \eta, v^{(k)}) \tag{12}$$

4: Updating $v^{(k+1)}$

$$v_{ij}^{(k+1)} = v_{ij}^{(m)} + a_{ij}(\mu_i^{(k+1)} - \mu_j^{(k+1)} - \eta_{ij}^{(k+1)}) \tag{13}$$

5: Check stop condition.

6: **end for**

The subproblems 10 to 13 are given in closed form solutions with a broad class of penalties. The detailed derivation is shown in the Appendix A.

4. Simulation studies

In this paper, the simulation is based on Spark platform. Apache Spark is a fast universal computing engine designed for large-scale data processing. Spark is a general parallel framework of Hadoop MapReduce-like open source in UC Berkeley AMP lab (AMP Lab, University of California, Berkeley). Spark has

the advantages of Hadoop MapReduce. But unlike MapReduce, the output of Job can be saved in memory so that it no longer needs to read and write HDFS. So Spark is better suited for data mining and machine learning, the algorithm of MapReduce which needs to be iterated.

We generate data from the following model

$$y_i = \mu_i + x_i^T \beta + \epsilon_i, i = 1, ..., n, \tag{14}$$

where $x_i = (x_{i1}, ..., x_{i5})^T$ are generated from the multivariate normal distribution $N(0, \Sigma)$, $\Sigma = (\sigma_{kj})$ and $\sigma_{kj} = 0.5^{|k-j|}$. We simulate $\beta = (\beta_1, ..., \beta_5)^T$ from independent Uniform $[0.5, 1]$. We generate μ_i from two different values $-\alpha$ and α with equal probabilities. More specific, we generate μ_i from the distribution: $p(\mu_i = -\alpha) = p(\mu_i = \alpha) = 1/2$. In our analysis, we compare the performance of the estimators by using MCP, SCAD and weighted L_1 penalty

$$P(|\mu_i - \mu_j|, \lambda) = \lambda w_{ij} |\mu_i - \mu_j|, \tag{15}$$

where w_{ij} presents the weights. For the L_1 penalty weights, Ma & Huang (2017) proposed a Gaussian kernel defined on the distance of two points $\exp(-\phi(y_i - y_j)^2)$, where the constant ϕ is nonnegative. When $\phi = 0$, 15 turns to be the Lasso penalty. Here we consider 100 realizations with $n = 100$, $\alpha = 1, 1.5, 2$ and $\phi = 0, 0.5, 1, 2$. We select λ by minimizing the modified BIC

$$BIC = \log \left[\sum_{i=1}^{n} (y_i - \hat{\mu}_i - x_i^T \hat{\beta})^2 / n \right] + C_n \frac{\log n}{n} (\hat{K} + p), \tag{16}$$

where $C_n = c \log(\log(n + p))$, $n + p$ is the number of components in μ and β, and c is a positive constant. We evaluate the algorithm by the model clustering ability, the estimation accuracy and the computational speed. Therefore, we consider the following criterias:

1. The average value and the standard error of the square root of the mean squared errors(MSE) for the estimated value of μ and β

$$MSE(\hat{\mu}) = \|\hat{\mu} - \mu\| / \sqrt{n}, \tag{17}$$

$$MSE(\hat{\beta}) = \|\hat{\beta} - \beta\| / \sqrt{p}. \tag{18}$$

2. The mean, median and standard error(s.e.) of $\hat{K}$.

3. The Rand Index measure Rand (1971) by:

$$RI = \frac{TP + TN}{TP + FP + FN + TN},$$

where a true positive (TP) decision assigns two observations from the same ground truth group to the same cluster, a true negative (TN) decision assigns two observations from different groups to different clusters, a false positive (FP) decision assigns two observations from different groups to the same cluster, and a false negative (FN) decision assigns two observations from the same group to different clusters.

4. the running times(CPU time) for the whole process of calculating the estimate $\hat{\beta}$, $\hat{\mu}$.

Figure 1 shows the solution paths for μ against λ by using MCP, SCAD and weighted L_1 penalties with $\phi = 0, 0.5, 1, 2$, based on 100 realizations with $n = 100$ and $\alpha = 2$.

From Table 1, we compare the results of two algorithms in different computing environments. Our two-step fusion penalty distributed algorithm applies in data distributed storage computing environment, and the concave pairwise fusion approach applies in stand-alone environment. In order to study the influence of the space separation data storage on estimation, we set the same sample size. The difference in comparison is that in distributed environment, data is stored distributed in M nodes, and data of different nodes can not interact. In the stand-alone environment, all data are stored on a single computer. The results in Table 1 show that our algorithm can basically solve the problem of data space separation. The accuracy of the results obtained by our algorithm in distributed environment is comparable to that obtained by the concave pairwise fusion approach in single computer.

Figure 1: Solution paths for $(\mu_1, ..., \mu_n)$ against λ values by using MCP, SCAD and weighted L1 penalties

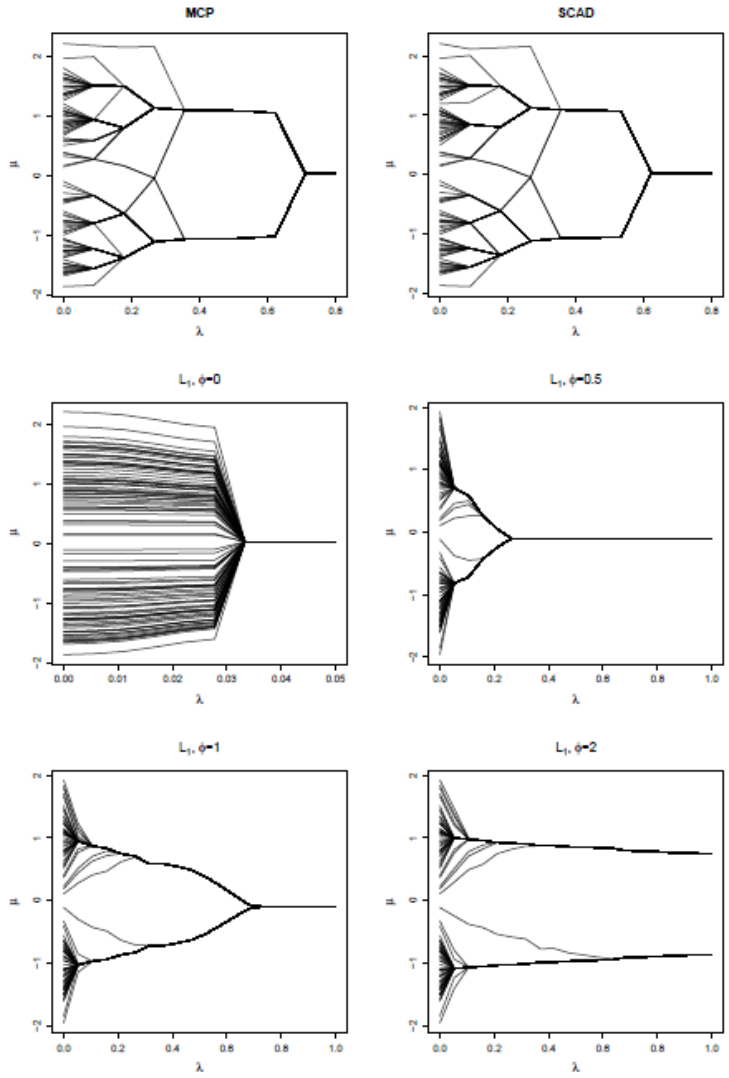

Table 1: The mean, median and standard error (s.e.) of $\hat{K}$ by the MCP, SCAD and weighted L_1 based on 100 realizations with n = 100 in distributed and stand-alone environment

c	Method	$\alpha = 1$			$\alpha = 1.5$			$\alpha = 2$		
		mean	median	s.e.	mean	median	s.e.	mean	median	s.e.
5	MCP	41.1%	100%	11.34	0.42	87.4%	100%	10.31	0.45	0.45
	DisMCP	41.1%	100%	11.34	0.42	87.4%	100%	10.31	0.45	0.45
	SCAD	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
	DisSCAD	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
	$L_1(\phi = 1)$	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
	Dis$L_1(\phi = 1)$	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
	$L_1(\phi = 2)$	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
	Dis$L_1(\phi = 2)$	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
10	MCP	41.1%	100%	11.34	0.42	87.4%	100%	10.31	0.45	0.45
	DisMCP	41.1%	100%	11.34	0.42	87.4%	100%	10.31	0.45	0.45
	SCAD	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
	DisSCAD	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
	$L_1(\phi = 1)$	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
	Dis$L_1(\phi = 1)$	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
	$L_1(\phi = 2)$	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45
	Dis$L_1(\phi = 2)$	20.2%	100%	17.73	2.30	88.2%	100%	13.71	2.21	0.45

Table 2: The mean and standard error (s.e.) shown in parentheses of the square root of the MSE for the estimated values of μ and β by the MCP, SCAD and L_1 penalty based on 100 realizations with n=100 and M=5

c	Method	μ			β		
		$\alpha=1$	$\alpha=1.5$	$\alpha=2$	$\alpha=1$	$\alpha=1.5$	$\alpha=2$
5	MCP	41.1%	100%	11.34	0.42	87.4%	100%
		(41.1%)	(100%)	(11.34)	(0.42)	(87.4%)	(100%)
	SCAD	41.1%	100%	11.34	0.42	87.4%	100%
		(41.1%)	(100%)	(11.34)	(0.42)	(87.4%)	(100%)
	$L_1(\phi=1)$	41.1%	100%	11.34	0.42	87.4%	100%
		(41.1%)	(100%)	(11.34)	(0.42)	(87.4%)	(100%)
	$L_1(\phi=2)$	41.1%	100%	11.34	0.42	87.4%	100%
		(41.1%)	(100%)	(11.34)	(0.42)	(87.4%)	(100%)
10	MCP	41.1%	100%	11.34	0.42	87.4%	100%
		(41.1%)	(100%)	(11.34)	(0.42)	(87.4%)	(100%)
	SCAD	41.1%	100%	11.34	0.42	87.4%	100%
		(41.1%)	(100%)	(11.34)	(0.42)	(87.4%)	(100%)
	$L_1(\phi=1)$	41.1%	100%	11.34	0.42	87.4%	100%
		(41.1%)	(100%)	(11.34)	(0.42)	(87.4%)	(100%)
	$L_1(\phi=2)$	41.1%	100%	11.34	0.42	87.4%	100%
		(41.1%)	(100%)	(11.34)	(0.42)	(87.4%)	(100%)

5. Real data example

In this section, we use the Cleveland Heart Disease Dataset to illustrate our method. The Cleveland Clinic Foundation heart disease dataset, contributed to the respository by Robert Detrano, contains 303 observations, 165 of which describe healthy people and 138 sick ones; 7 observations are incomplete, and 2 of the observations of healthy people have identical attribute values. We take the complete observation in our analysis. Each observation is described by 13 attributes, including 3 Boolean(e.g. sex), 4 nominal(e.g. type of chest pain), and 6 numerical(e.g. age). The output indicates the angiographic status of the disease, i.e. whether the narrowing of the vessel diameter is above or below

50%. Our analysis is to conduct subgroup for the fitted value of thalach as the response y after adjusting for the effects of the covariates: x_1=age in year; x_2=gender; x_3=resting blood pressure; x_4=serum cholesterol; x_5=fasting blood suger indicator; and x_6=resting electrocardiographic result;

In order to simulate hospital data sharing, we artificially distribute the samples in 5 nodes. We fit the heterogeneous model 1, and we identify subgroups by our proposed two-step fusion penalty approach. We select the tuning parameter by minimizing the modified BIC. As a result, two major groups are identified by both of the MCP and SCAD methods.

Table 3: The estimated values (est) for the coefficients β and μ by the OLS, MCP and SCAD under standalone and distributed environment

Method	β_1	β_2	β_3	β_4	β_5	β_6	μ_1	μ_2
OLS	1	1	1	1	1	1	-	-
MCP	1	1	1	1	1	1	1	1
DisMCP	1	1	1	1	1	1	1	1
SCAD	1	1	1	1	1	1	1	1
DisSCAD	1	1	1	1	1	1	1	1

We also calculate the coefficient of determination R^2, and obtain $R^2 = 0.667$, 0.704, 0.704, 0.667 and 0.109 for MCP, DisMCP, SCAD, DisSCAD and OLS methods. We see that taking into account the subgroup structure leads to a significant improvement of the model fitting. We find that for the actual data, even if we artificially distribute them in different nodes, the results obtained by our approach are almost the same as those obtained by concave pairwise fusion approach in a single computer environment.

6. Conclusion

In this paper, we propose a distributed algorithm named two-step fusion penalty approach to estimate the heterogeneous treatment effects under distributed storage. As an application, we analysis the Cleveland heart disease

dataset and we see that our approach leads to a significant improvement of the model fitting.

References

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J. et al. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, *3*, 1–122.

Cervellera, C., & Macciò, D. (2018). Distribution-preserving stratified sampling for learning problems. *IEEE transactions on neural networks and learning systems*, *29*, 2886–2895.

Everitt, B., & Hand, D. (1981). Finite mixture distributions chapman and hall. *New York*, .

Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, *96*, 1348–1360.

Farewell, V. T. (1982). The use of mixture models for the analysis of survival data with long-term survivors. *Biometrics*, (pp. 1041–1046).

Ma, S., & Huang, J. (2017). A concave pairwise fusion approach to subgroup analysis. *Journal of the American Statistical Association*, *112*, 410–423.

Maugis, C., Celeux, G., & Martin-Magniette, M.-L. (2009). Variable selection for clustering with gaussian mixture models. *Biometrics*, *65*, 701–709.

Muthén, B., & Shedden, K. (1999). Finite mixture modeling with mixture outcomes using the em algorithm. *Biometrics*, *55*, 463–469.

Neyman, J. (1934). On the two different aspects of the representative method: the method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, *97*, 558–625.

Pauler, D. K., & Laird, N. M. (2000). A mixture model for longitudinal data with application to assessment of noncompliance. *Biometrics*, *56*, 464–472.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, *66*, 846–850.

Rasmussen, C. E. (2000). The infinite gaussian mixture model. In *Advances in neural information processing systems* (pp. 554–560).

Shen, J., & He, X. (2015). Inference for subgroup analysis with a structured logistic-normal mixture model. *Journal of the American Statistical Association*, *110*, 303–312.

Zhang, C.-H. et al. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, *38*, 894–942.

Zhao, X., Liang, J., & Dang, C. (2019). A stratified sampling based clustering algorithm for large-scale data. *Knowledge-Based Systems*, *163*, 416–428.