Real World Data Machine Learning and Estimation of Graphical Models using Lasso-related Approaches

Candidate Numbers: 36878, 39295, 42471, 36792
London School of Economics and Political Science

December 12, 2019

Contents

Part I

Real World Data

1 Introduction

1.1 Data Description and Objectives

The original data is a marketing data file of a Portuguese banking institution containing 41188 records from 2008 to 2010[1]. It contains variables related to client information, contact information, previous contact history, social and economic context as well as the outcome which is whether the client subscribed to the marketing product or not. Institutions would gain more profit by employing precision marketing, which can be supported by a classification model. The goal of this part is to find out which model makes better predictions, how to improve the performance and how to implement the model in the context of telemarketing.

1.2 Performance Measure

In this part, three performance measures are used to compare models: accuracy, recall (true positive rate) and AUC. For bank telemarketing, the profit gained from one successful subscription is much higher than the cost of making a phone call. Statistically, true positives are more important than true negatives and therefore high recall is preferred.

The principle is to choose high-performance classifier in all three measures. When other performances are roughly equal, a classifier with the highest recall is preferred.

2 Data Preprocessing

The data is randomly split into training and validation data (2/3) and test data (1/3).

2.1 Variable Transformation and Standardisation

After the categorical variables are recoded into sets of dummy variables, two dummy variables `loanunknown` and `housingunknown` are perfectly collinear so one of them is removed. The variable `duration` is also removed because it is not known before a call is performed in reality. For the numerical variable `pdays`, value 999 means client was not previously contacted so the inverse of `pdays` + 1 is used and values are transformed into decimals between 0 and 1. Standardisation is applied to numerical variables based on the training data and particularly, count variables `campaign` and `previous` are centered at 1 and 0. There are 51 predictors after preprocessing.

2.2 Oversampling

The target `y` is imbalanced with only 11.3% of class `yes`, which might lead to useless classifier and therefore the minority class is oversampled to have balanced training data. Table 1 shows considerable improvement in recall after oversampling and therefore the high accuracy before oversampling might be misleading. So the oversampled training data is used in this part.

Table 1: Summary of results before and after oversampling

	Accuracy-before	**Accuracy-after**	**Recall-before**	**Recall-after**	**AUC-before**	**AUC-after**
LR	89.8%	83.6%	21.0%	**62.5%**	0.795	0.796
LDA	89.0%	83.5%	37.7%	**62.4%**	0.792	0.796

3 Experiments and Results

Four classification models are investigated including logistic regression (LR), linear discriminant analysis (LDA), K-Nearest Neighbors (KNN) and Random Forest (RF). The settings are `k` = 15 in KNN, default in RF and all 51 features are included in LR and LDA.

Based on the results from Table 2, LR performs slightly better than LDA in all measures. Compared with other models, KNN has much lower accuracy and RF has much lower recall. It is notable that KNN and RF are considerably more compute-intensive than LR and LDA.

Table 2: Summary of results

	Accuracy	**Recall**	**AUC**
LR	83.6%	62.5%	0.796
LDA	83.5%	62.4%	0.796
KNN	67.4%	68.1%	0.739
RF	87.4%	48.3%	0.789

	Accuracy	**Recall**	**AUC**
LR-FSS	83.6%	61.9%	0.796
LR-Lasso	83.4%	63.3%	0.794
LR-Poly-Lasso	83.1%	**64.0%**	0.800
Adaboost	**84.3%**	63.0%	**0.809**

4 Logistic Regression and Tuning

In view of the great performance and simplicity, LR is paid more attention to. Before running the model, outliers are first detected based on the cook's distance and 686 records are removed from training data.

4.1 Subset Selection

In order to implement subset selection, relaxed lasso as well as forward and backward stepwise selection (FSS and BSS) are applied. Relaxed lasso, FSS, BSS and `stepAIC` function select 45, 23, 24 and 25 features respectively. As shown in Table 2, the classifier, which was fitted with features selected by 5-fold cross-validation lasso based on misclassification error, shows more improvement in recall than other methods.

4.2 Polynomial Features

Polynomial terms of numerical variables are also taken into consideration. Quadratic terms of all numerical variables are first included in the formula and then 5-fold cross-validation lasso is implemented. Based on misclassification error, 25 features are selected and the new classifier shows great improvement in recall and AUC, as shown in Table 2.

5 Adaboost and Tuning

Adaboost is a boosting algorithm which can be applied to a classification problem. There are tree-specific parameters in the `gbm` package such as the maximum depth of each tree (`interaction.depth`) and the minimum number of observations in the terminal nodes (`n.minobsinnode`). For parameter tuning, lists of values of these two parameters are provided to implement the grid search algorithm with 5-fold cross-validation. The best combination based on accuracy is `interaction.depth=4, n.minobsinnode=10`. Lastly, Adaboost has an outstanding performance on test data in accuracy and AUC, which are shown in Table 2.

6 Summary and Conclusion

Although Adaboost has the best performance in accuracy and AUC, it is low in recall and difficult to interpret and implement in telemarketing. Logistic regression with 25 polynomial features selected by lasso is our final choice because of the high performance, simplicity and interpretability.

According to Table 3, marketing campaigns are more likely to succeed in a good economy and phone calls are more likely to be responded through mobile phones and in March and December. For the clients, the model shows that students and retired people could become target groups. It is also notable that clients who had previously subscribed a product through telemarketing or recently contacted are more likely to subscribe.

When it comes to implementing a classifier in telemarketing, a probabilistic outcome can be predicted for each client and then all clients can be ranked in descending order. The clients with the highest probability can be first contacted and it would make telemarketing not only more accurate but also more efficient and organised.

Table 3: Coefficient table of final model

significance code: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

client	**est**	
age^2	0.07	***
job-retired	0.21	**
job-student	0.51	***
mar-single	0.08	***
edu-univ	0.11	***
edu-unkn	0.19	***
def-unkn	-0.21	***

history	**est**	
campaign	0.00	
campaign^2	-0.04	***
pdays	0.27	***
pout-non	0.43	***
pout-success	1.45	***

contact	**est**	
con-tele	-0.31	***
mon-dec	2.49	***
mon-mar	2.02	***
mon-may	-0.70	***
mon-nov	-0.18	**
mon-oct	1.75	***
day-mon	-0.16	**
day-wed	0.09	***

social	**est**	
nr.employed	-0.58	***
nr.employed^2	0.05	**
cons.conf.idx^2	-0.23	***
emp.var.rate^2	0.19	***
euribor3m^2	0.23	**

Part II

Estimation of Graphical Models using Lasso-related Approaches

1 Introduction

Graphical modelling is widely used in the estimation of large covariance matrices and plays an important role in multiple disciplines ranging from economics and finances to engineering and computer science[2]. One of such approaches is node-wise lasso, which addresses the problem by, for each node, regressing on the remaining variables. For sparse high-dimensional graphs, this approach was investigated by Meinshausen and Buhlmann (2006)[3] and Peng et al. (2009)[4]. Alternatively, there is graphical lasso, which was developed by Friedman et al. (2007)[5] and estimates the covariance matrix relying on l_1 penalty term.

The main goal of this part is to study graphical and node-wise lasso approaches, develop parameter tuning methods, and evaluate their performance under different settings.

2 The Graphical and Node-wise Lasso Approaches

2.1 Problem set-up

The conditional dependence of p random variables $X = (X_1, \dots, X_p)^T$ can be modelled by an undirected graph $G = (V, E)$ with vertex set $1, \dots, p$ and edge set E defined in (2).

$$c_{jl} = Cov(X_j,\ X_l \mid X_k, 1 \le k \le p, k \ne j, l) \tag{1}$$

$$E = \{(j,l) : c_{lj} \ne 0,\ 1 \le j,\ l \le p,\ j \ne l\} \tag{2}$$

Under the assumption that X is multivariate Gaussian with covariance matrix Σ which is the inverse of Θ, it can be proved that $c_{jl} = 0$ if and only if the (j,l)-th entry of inverse covariance matrix $\Theta_{jl} = 0$. Therefore, E also can be written as:

$$E = \{(j,l) : \Theta_{jl} \ne 0,\ 1 \le j,\ l \le p,\ j \ne l\} \tag{3}$$

2.2 The Node-wise Lasso Approach

One can perform lasso approach to regress X_j on the rest of the variables $X_l, l \in V, l \ne j$, where b_{jl} is the coefficient of X_l to regress X_j. The node-wise lasso approach estimates E based on the lasso estimator for b_{ij}. Two rules can be considered as defined in (4) and (5), which are named node-wise lasso 1 and node-wise lasso 2 respectively.

$$\widehat{E}_1 = \{(j,l) : b_{jl},\ b_{lj} \ne 0,\ 1 \le j,\ l \le p,\ j \ne l\} \tag{4}$$

$$\widehat{E}_2 = \{(j,l) : b_{jl} \text{ or } b_{lj} \ne 0,\ 1 \le j,\ l \le p,\ j \ne l\} \tag{5}$$

2.3 The Graphical Lasso Approach

The graphical lasso approach estimates E based on the estimator for Θ_{ij}. The rule can be considered as defined in (6), which is named graphical lasso.

$$\widehat{E}_3 = \{(j,l) : \widehat{\Theta}_{jl} \neq 0,\ 1 \leq j,\ l \leq p,\ j \neq l\} \tag{6}$$

3 Simulation and Application

3.1 Data Simulation

For general use of simulated data in this part, $p = 50$, $n = 500$, $prob = 0.1$ are set so that there are 50 random variables, 500 samples of each variable and each off-diagonal entry in the inverse covariance matrix is non-zero with probability of 10%. There are 1225 possible edges under these settings and the simulated E contains 118 edges.

To simplify the problem, a vector of a binary outcome in a certain order is introduced to represent the set E. The length of the vector is the total number of possible edges in the graph. Outcome 1 signifies that the edge is included in E and outcome 0 indicates that the edge is not in E.

3.2 Application of the Approaches with a Certain λ

All approaches are firstly applied using a certain λ, which equals 0.08.

Sets $\widehat{E}_1$, $\widehat{E}_2$ and $\widehat{E}_3$ are estimated from definitions (4), (5) and (6). Package `glmnet` is used to implement the node-wise lasso approach and package `glasso` is used to implement the graphical lasso approach. Once an estimated vector which represents the estimated edge set is derived, a confusion matrix and other measures such as false negative rate (FNR), false positive rate (FPR) and misclassification rate can be computed.

The results are shown in tables 4 and 5. Under these simulation settings and for this particular λ, the most accurate approach is node-wise lasso 2, while the least accurate one is graphical lasso.

Table 4: The confusion matrices for three approaches with $\lambda = 0.08$

node1	**F**	**T**
F	1012	2
T	95	116

node2	**F**	**T**
F	1041	2
T	66	116

graphical	**F**	**T**
F	940	0
T	167	118

Table 5: Summary of FPR, FNR and misclassification rate for three approaches

	FPR	**FNR**	**misclassification rate**
node-wise 1	8.58%	1.69%	7.92%
node-wise 2	5.96%	1.69%	5.55%
graphical	15.19%	0%	13.63%

3.3 ROC Curves and AUC

To produce ROC curves for all three approaches, a grid of 50 different λ values is provided to obtain FPRs and FNRs. Additionally, the AUC values can be deduced by estimating area under each curve.

From figure 1, it is clear that all three approaches produce a very good estimation under this settings because all AUCs are approaching 1 as shown in table 6. Furthermore, these values indicate that graphical lasso is slightly less accurate than both node-wise lasso.

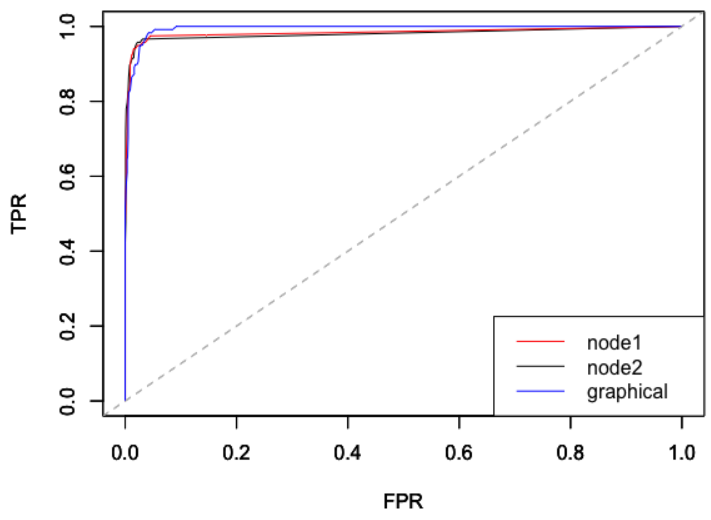

Figure 1: ROC curves for three approaches

Table 6: AUC values for node-wise and graphical lasso

	AUC
node-wise 1	0.9941819
node-wise 2	0.9956785
graphical	0.9936843

3.4 Time Performance

The time taken for each approaches to run 50 times was measured. For node-wise approaches jointly, it took 10.3 second while for the graphical lasso, 0.9 seconds, which is 10 times faster than node-wise lasso. Hence, graphical lasso is much more computationally efficient than node-wise combined.

4 Tuning Parameters λ

4.1 Parameter Tuning Method

To find the optimal λ for each method, the misclassification rate is used as the performance metric. Inspired by K-fold cross-validation, K misclassification rates on each λ are generated using K different parts of data. Despite the same test data being used in this context, the intuition behind cross-validation can be applied to generate K different training data. Then, it is possible to find the λ which produces the lowest mean misclassification rate. Finally, one-standard-deviation rule can be applied to find the optimal λ.

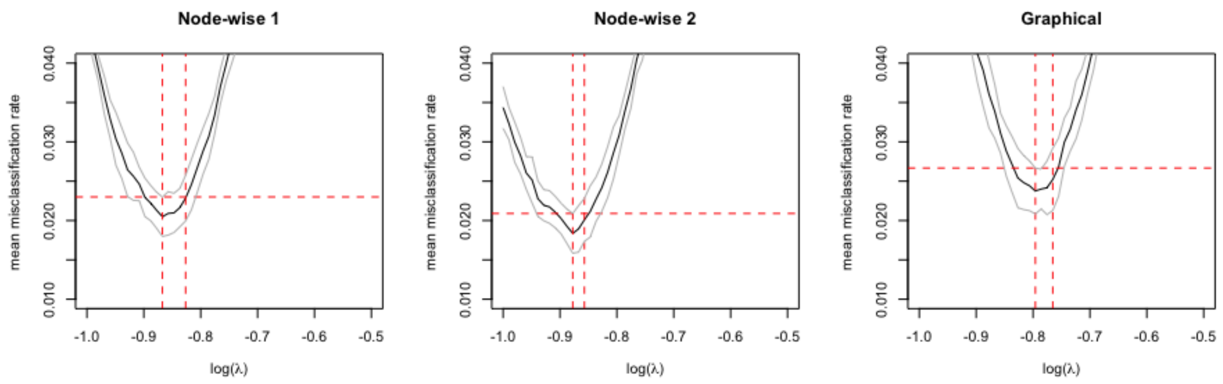

Figure 2: Mean misclassification rate against $\log(\lambda)$ for three lasso approaches.

The results shown above in figure 2 are produced using 10-fold cross-validation. The grey lines represent the one standard error bands. λ which generates the lowest mean misclassification rate and λ which is selected by one-standard-deviation rule are highlighted with red dashed lines. The optimal λ in these settings are shown in table 7.

Table 7: Minimum misclassification rates for each lasso

	$\hat{\lambda}$
node-wise 1	0.1490972
node-wise 2	0.1389495
graphical	0.1716698

Based on the selected optimal parameters, sets $\widehat{E}_1$, $\widehat{E}_2$ and $\widehat{E}_3$ can be estimated respectively. Confusion matrices and corresponding performance measures computed using all the simulated samples are shown in tables 8 and 9.

Table 8: The confusion matrices for three approaches with $\lambda = \hat{\lambda}$

node1	**F**	**T**
F	1105	21
T	2	97

node2	**F**	**T**
F	1105	16
T	2	102

graphical	**F**	**T**
F	1103	25
T	4	93

Table 9: Summary of performance measures for three approaches for different λ

	measure	$\lambda = \hat{\lambda}$	$\lambda = 0.08$
	FPR	0.18%	8.58%
node-wise 1	FNR	17.80%	1.69%
	misclassification rate	1.88%	7.92%
	FPR	0.18%	5.96%
node-wise 2	FNR	13.56%	1.69%
	misclassification rate	1.47%	4.82%
	FPR	0.36%	15.19%
graphical	FNR	21.19%	0%
	misclassification rate	2.37%	13.63%

The results from Table 9 approach conclusions from two different directions. Firstly, as λ goes to the optimal λ, sample performance becomes better. It is notable that FNRs become higher. This happens because the misclassification rate is the metric which is optimised and it balances between FPR and FNR. The imbalanced classes would exaggerate the change of FNR. Then, comparing different approaches, the performance of node-wise 2 lasso is similar to that one of node-wise 1 but is slightly better. This is consistent with the definition of stricter condition on node-wise 2. Lastly, graphical approach has the worst performance.

4.2 Replicating and Testing

To test the out-of-sample performance of selected optimal λ, different sets of data samples are simulated using the same Θ and E.

The results shown in figure 3 are generated from 50 different simulations. For all three measures, the mean of node-wise 2 is the smallest and the mean of graphical approach is the largest, which is consistent with the in-sample performance. In terms of standard error as shown in table 10, graphical approach is the most variant while both node-wise approaches are less variant. This behaviour also complies with the bias-variance trade-off property.

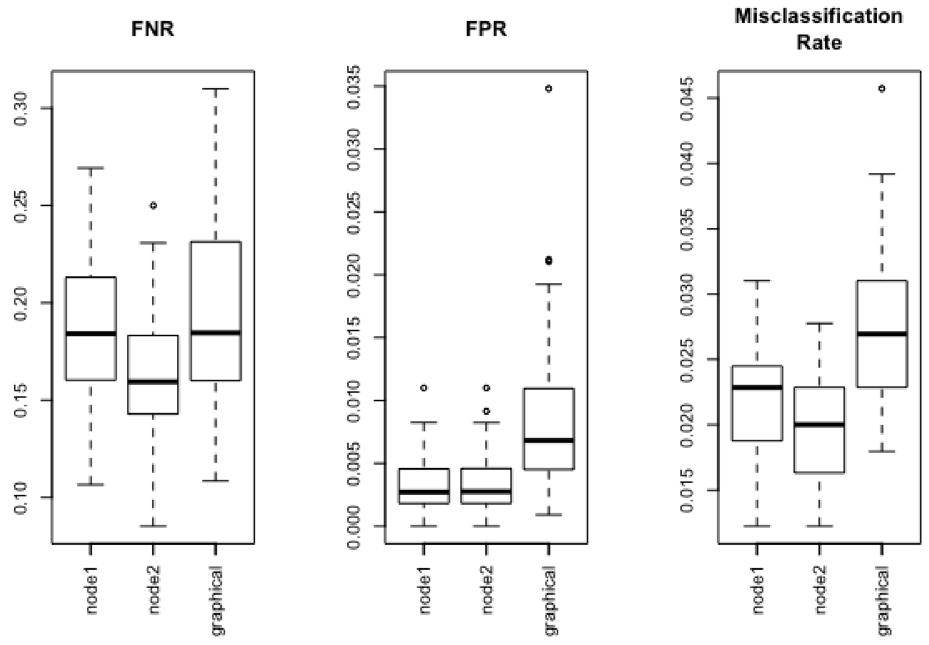

Figure 3: Boxplot of FPR, FNR and misclassification rate from 50 simulations

Table 10: Summary of standard error of FPR, FNR and misclassification rate (in 1e-4)

	FNR	FPR	misclassification rate
node-wise 1	2.42	37.03	4.30
node-wise 2	2.33	35.04	4.03
graphical	6.28	46.22	5.80

4.3 Replication

The whole procedure — including generating data to computing AUC and minimum misclassification rate over a grid of λ values — is repeated for each method 50 times under the original settings. Additionally, the mean and standard error of each measure is computed.

Based on the results demonstrated in figure 4, graphical lasso consistently produces the lowest AUC with the largest standard error. Node-wise 2 has the highest AUC and the smallest standard error. The performance in misclassification rate follows the same pattern. This is also consistent with the bias-variance trade-off property.

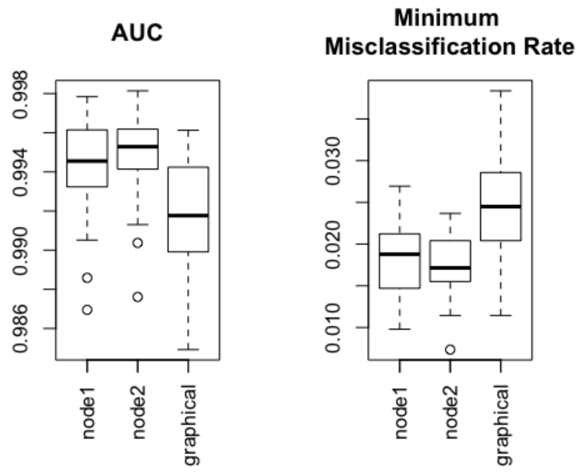

Figure 4: AUC and minimum misclassification rates for all lasso approaches

5 Simulation Settings

To explore other simulation settings, initially the number of samples (n) is varied while the number of variables (p) and ($prob$), probability to have non-zero entry in the inverse covariance matrix, are fixed. After that, $prob$ is changed as the other two parameters are fixed.

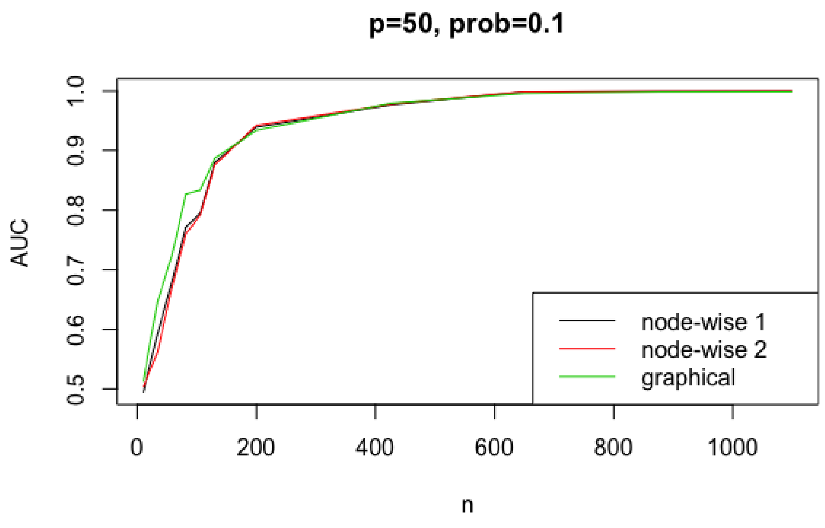

Figure 5: AUC against n

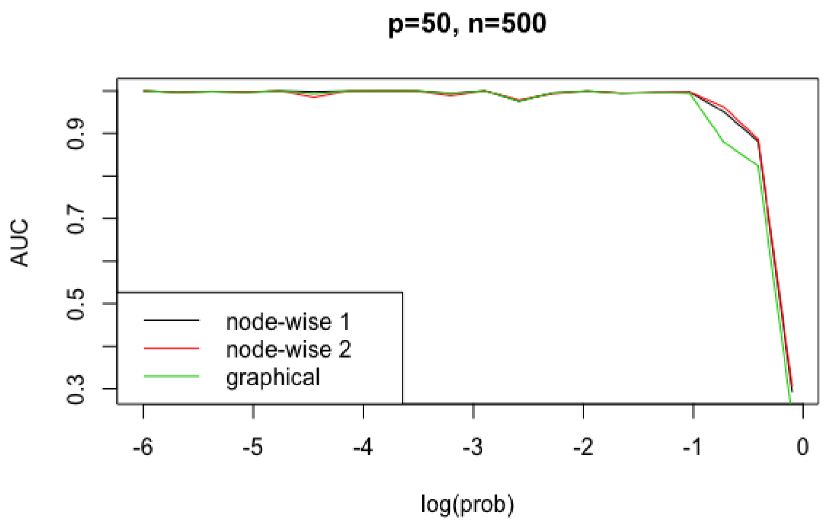

Figure 6: AUC against $prob$

Firstly, the values of AUC are plotted against various values of n in range from 0 to 1000 in figure 5. When $n < p$, the model is not reliable and produces an AUC value below 0.6. Once n increases past the value of p, the AUC increases rapidly for $n < 200$ as expected. For $n \geq 200$, the performance improves but at a much slower pace.

Overall, all three approaches perform very similarly. However, there is a slight difference for smaller values of n: graphical lasso marginally outperforms both node-wise approaches. For very large n, graphical lasso is slightly worse, but the difference in the AUC values from this method and node-wise lasso is negligible.

Secondly, figure 6 visualises how AUC is stable and high for low *prob* and drops rapidly as soon as *prob* increases past a certain point. This behaviour is originated from the sparsity structure and the imbalanced classification problem. For small *prob*, Θ is sparse and there are a few actual positive outcomes in the vector representing E. This leads to TPR approaching 1 while FPR is nearly 0. As a result, AUC is very close to 1. However, as *prob* increases past 0.2, Θ becomes less sparse, and, therefore, there is a sharp decrease in TPR and AUC.

Comparing three approaches, it is possible to deduce that graphical lasso performs worse than both node-wise approaches when Θ is not sparse: there is a significant deviation of AUC for graphical lasso from the other two lines around $prob = 10^{-0.5}$. On the other hand, both node-wise approaches produce very similar AUC values. For a larger probability, $prob > 0.3$, node-wise 1 tends to underperform in comparison to node-wise 2, with the difference between their AUC values growing as *prob* increases.

6 Summary

In this part, graphical lasso approach and node-wise lasso approach are introduced to estimate a graphical model and a conditional dependence structure. Package glmnet and glasso are used to implement these approaches in R. The estimation performance of an approach can be evaluated by FPR, FNR, misclassification rate and AUC.

Besides, a parameter-tuning method which uses cross-validation based on misclassification rate is developed. The method is capable of improving the performance for both in-sample and out-of-sample data. Based on the optimal tuning parameter, it is shown that the graphical lasso approach is slightly less accurate and more variant than node-wise lasso approach is. However, it is also worth noting that graphical lasso approach is significantly computationally faster than node-wise lasso approaches.

Moreover, experiments are also conducted under different simulation settings. The conditional dependence structure is more difficult to estimate when there are more variables than the samples. Graphical lasso approach outperforms other lasso when the sample size is not large enough. In terms of sparsity, the estimation of a graphical model is more accurate when the structure is sparser. Both node-wise lasso have better performance for a less sparse structure and a more balanced classification problem.

A Appendix 1

A.1 Attribute Information

Table 11: Summary of Attribute Information[6]

Factor	**Attributes**	**Type**	**Description**
Bank client	age	Numeric	
	job	Categorical	Type of job: admin, blue-collar, entrepreneur, housemaid, management, retired, self-employed, services, student, technician, unemployed, unknown.
	marital	Categorical	Marital status: divorced, married, single, unknown; note: divorced means divorced or widowed.
	education	Categorical	basic.4y, basic.6y, basic.9y, high.school, illiterate, professional.course, university.degree, unknown.
	default	Categorical	Has credit in default? no, yes, unknown.
	housing	Categorical	Has housing loan? no, yes, unknown.
	loan	Categorical	Has personal loan? no, yes, unknown.
Telemarketing	contact	Categorical	Contact communication type: cellular, telephone
	month	Categorical	Last contact month of year: jan, feb, mar, ..., nov, dec
	day_of_week	Categorical	Last contact day of the week: mon, tue, wed, thu, fri
	duration	Numeric	Last contact duration, in seconds (numeric). Important note: this attribute highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model.
	campaign	Numeric	Number of contacts performed during this campaign and for this client (includes last contact)
	pdays	Numeric	Number of days that passed by after the client was last contacted from a previous campaign (999 means client was not previously contacted)
	previous	Numeric	Number of contacts performed before this campaign and for this client
	poutcome	Categorical	Outcome of the previous marketing campaign (failure, nonexistent, success)
Social and economic indictors	emp.var.rate	Numeric	Employment variation rate - quarterly indicator
	cons.price.idx	Numeric	Consumer price index - monthly indicator
	cons.conf.idx	Numeric	Consumer confidence index - monthly indicator
	euribor3m	Numeric	Euribor 3 month rate - daily indicator
	nr.employed	Numeric	Number of employees - quarterly indicator

A.2 Detailed Stepwise Selection Methods

Table 12: Summary of Stepwise Selections Performances

	Accuracy	Recall	AUC	Size
LR-FSS	83.6%	61.9%	0.796	23
LR-BSS	83.6%	61.9%	0.796	24
LR-stepAIC	83.7%	61.9%	0.798	25

To treat the marketing classification problem as practical as possible, even we are aware that the stepwise selection function `regsubset` is applied for linear regression instead of logistics regression we used, we still make a try to see whether it can give a surprisingly good result for this particular dataset as it is much more computationally efficient than other approaches. Also, to overcome the inconsistency problem, we further introduce `stepAIC` function for backward selection with model size decided by choosing the lowest Bayesian information criterion, i.e. BIC, which is a criterion that can select smaller models and improve interpretability. The performances of these three selection approaches are included in Table 11.

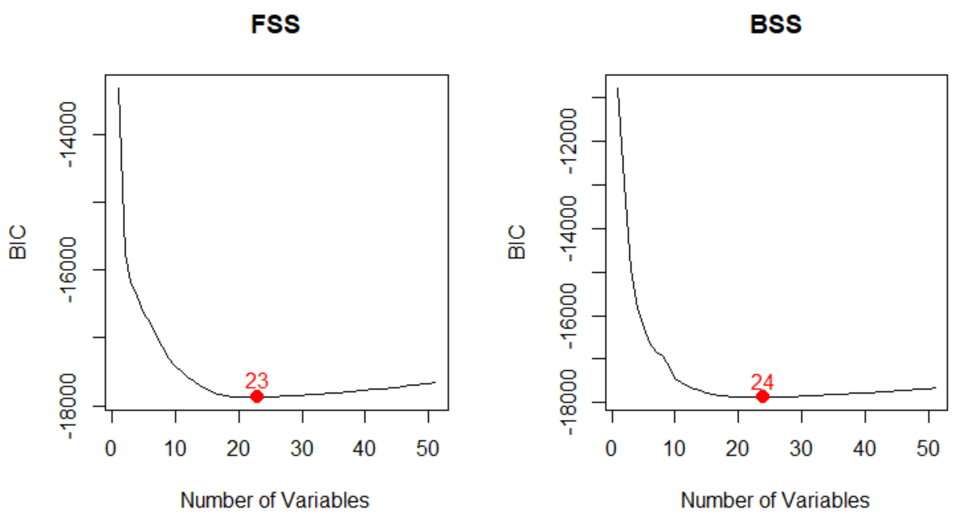

Figure 7: Optimal model sizes determined by BIC using `regsubset` function. Left: Forward Stepwise Selection, Right: Backward Stepwise Selection.

References

[1] S. Moro, P. Cortez, and P. Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems, Elsevier, 62:22-31*, June 2014.

[2] Moscone F. Tosetti E. and Vinciotti V. Sparse estimation of huge networks with a block-wise structure. *The Econometrics Journal, 20(3):S61–S85*, 2017.

[3] Meinshausen N. and Bühlmann P. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics, 34(3):1436-1462*, 2006.

[4] Zhou N., Peng J., Wang P., and Zhu J. Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association: 104(486):735-746*, 2009.

[5] Friedman J. Hastie T. and Tibshirani R. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics, 9(3):432–441*, 2007.

[6] Bank marketing data set. https://archive.ics.uci.edu/ml/datasets/bank+marketing#.