Homework 3

Machine Learning

1 SVM vs. Neural Networks

In this part we perform both SVM and neural networks (MLP) methods for classification works on the selected datasets. Different experiment settings and configurations are tried for a comprehensive study on the performance of these two methods.

1.1 Datasets

Two datasets are selected in this part: `splice` and `satimage`. Both datasets are with relatively less features and a appropriate proportion of training and testing data. The details of these two datasets are shown in Tab. 1. All the data are first processed with the `scale` function provided in `sklearn`: `preprocessing.StandardScaler`.

Table 1: Details of the Datasets.

Datasets	Classes	Features	Data
`splice`	2	60	1,000 / 2,175 (testing)
`satimage`	6	36	4,435 / 2,000 (testing)

1.2 Experiment Settings

1.2.1 SVM

For SVM implementation, we use the function provided in `sklearn`: `sklearn.svm.SVC` and change the parameters for different model settings. Tab. 2 shows the descriptions of the related parameters. The two datasets vary in data dimensions as well as training data sample size and we will also try different kernels for SVM. Tab. 3 shows all the different configurations of the experiment related to SVM.

Table 2: Important parameters of SVM models.

Parameters	Description	Values
C	Penalty parameter of the error term.	float
kernel	Specifies the kernel type to be used in the algorithm.	linear, poly, rbf, sigmoid, precomputed or a callable.
decision_function_shape	Multi-class strategy: one-vs-one or one-vs-rest.	ovo, ovr

Table 3: SVM experiment configurations.

Datasets	Kernel	C
`splice`	linear, poly, rbf, sigmoid	$[0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 5, 10]$
`satimage`	linear, poly, rbf, sigmoid	$[0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 5, 10]$

1.2.2 Neural Networks

As to neural networks, we also use the function provided in `sklearn`: `MLPClassifier` and change the parameters (displayed in Tab. 4) for different experiment configurations. Tab. 5 demonstrates all that have been done with the experiment related to neural networks.

Table 4: Parameters for MLP classifier.

Parameters	Description	Values
hidden_layer_sizes	The ith element represents the number of neurons in the ith hidden layer.	tuple
activation	Activation function for the hidden layer.	identity, logistic, tanh, relu
solver	The solver for weight optimization.	lbfgs, sgd, adam
alpha	L2 penalty (regularization term) parameter.	float
learning_rate	Learning rate schedule for weight updates.	constant, invscaling, adaptive
learning_rate_init	The initial learning rate used.	double

Table 5: MLP experiment configurations.

Datasets	hidden_layer_sizes	activation	learning rate
`splice`	(10/20), (10/20, 10/20), (10/20, 10/20, 10/20)	identity, logistic, tanh, relu	0.0001, 0.001, 0.005, 0.01, 0.1
`satimage`	(10/20), (10/20, 10/20), (10/20, 10/20, 10/20)	identity, logistic, tanh, relu	0.0001, 0.001, 0.01, 0.1, 0.5

1.3 Experiment Results

In this part we demonstrate the results of SVM and MLP respectively. We will conduct an experiment to compare the performance of SVM and neural networks in next section.

1.3.1 SVM

Figure. 1 shows the performance difference of SVM with various kernels. As we can see from the picture, rbf performs best on Splice dataset. On the Satimage dataset, the four kernel functions perform relatively the same when the number of iterations is large, while the 2- or 3-degree polynomial kernel perform relatively poor when the number of iterations is small.

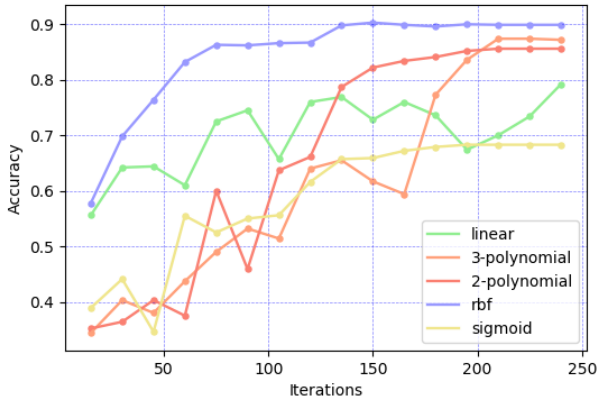
Fig. 2 demonstrates the performance difference of SVM with various penalty parameters. The penalty parameter is used to prevent the problem of over-fitting in the soft-margin technique: transforming the optimization object from

$$\frac{1}{2}w^T w$$

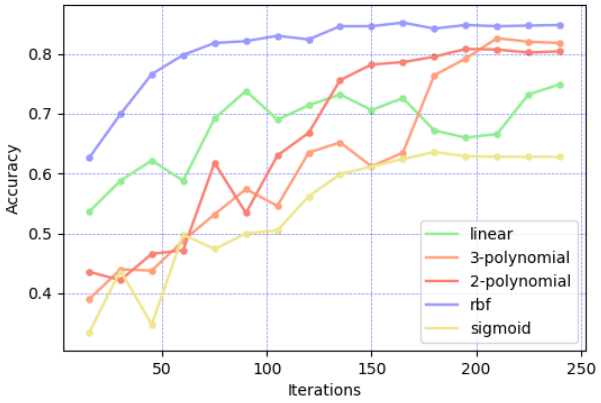
to

$$\frac{1}{2}w^T w + C\sum_{n=1}^{N}\xi_n,$$

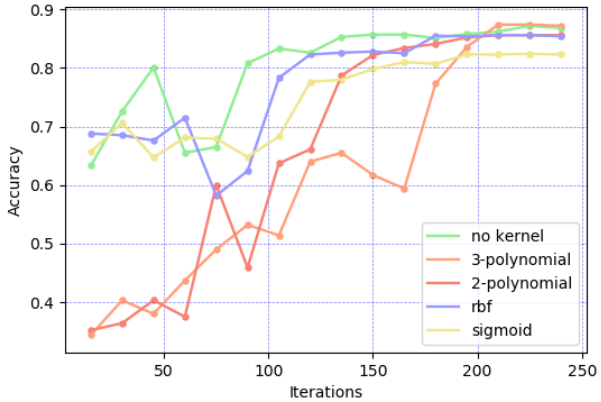
where the penalty parameter C is critical for classification performance and robustness. As can be seen from the results, the introduction of C is beneficial to the classification performance. More specifically, in the tested two datasets, with a proper C the accuracy of testing dataset is improved largely. A properly selected C indeed can increase the robustness of the classifier.



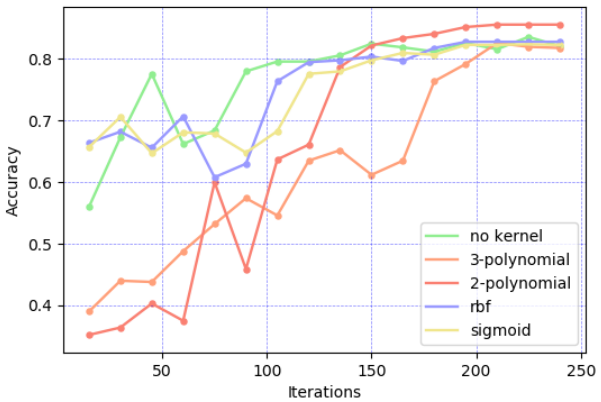
(a) Splice Training Dataset



(b) Splice Testing Dataset

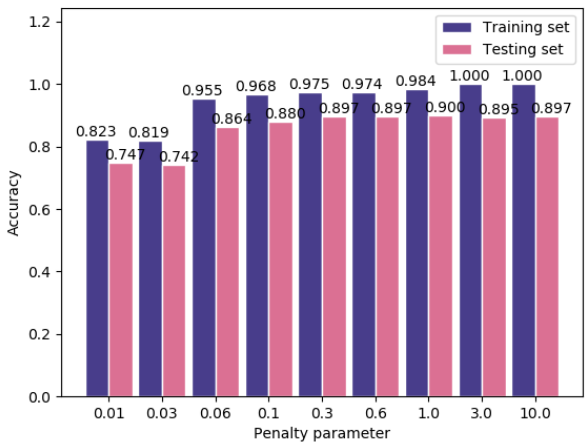


(c) Satimage Training Dataset

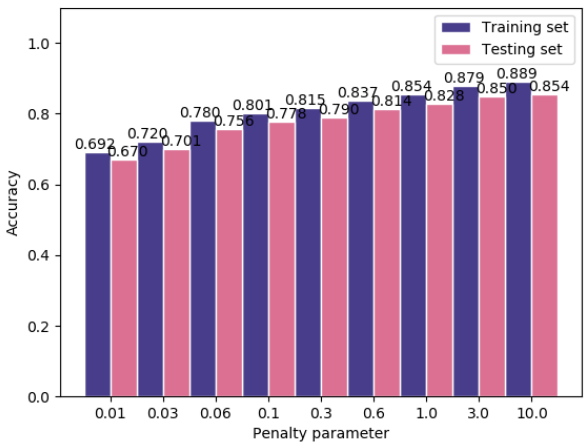


(d) Satimage Testing Dataset

Figure 1: Performance of SVM with different kernels.



(a) Splice



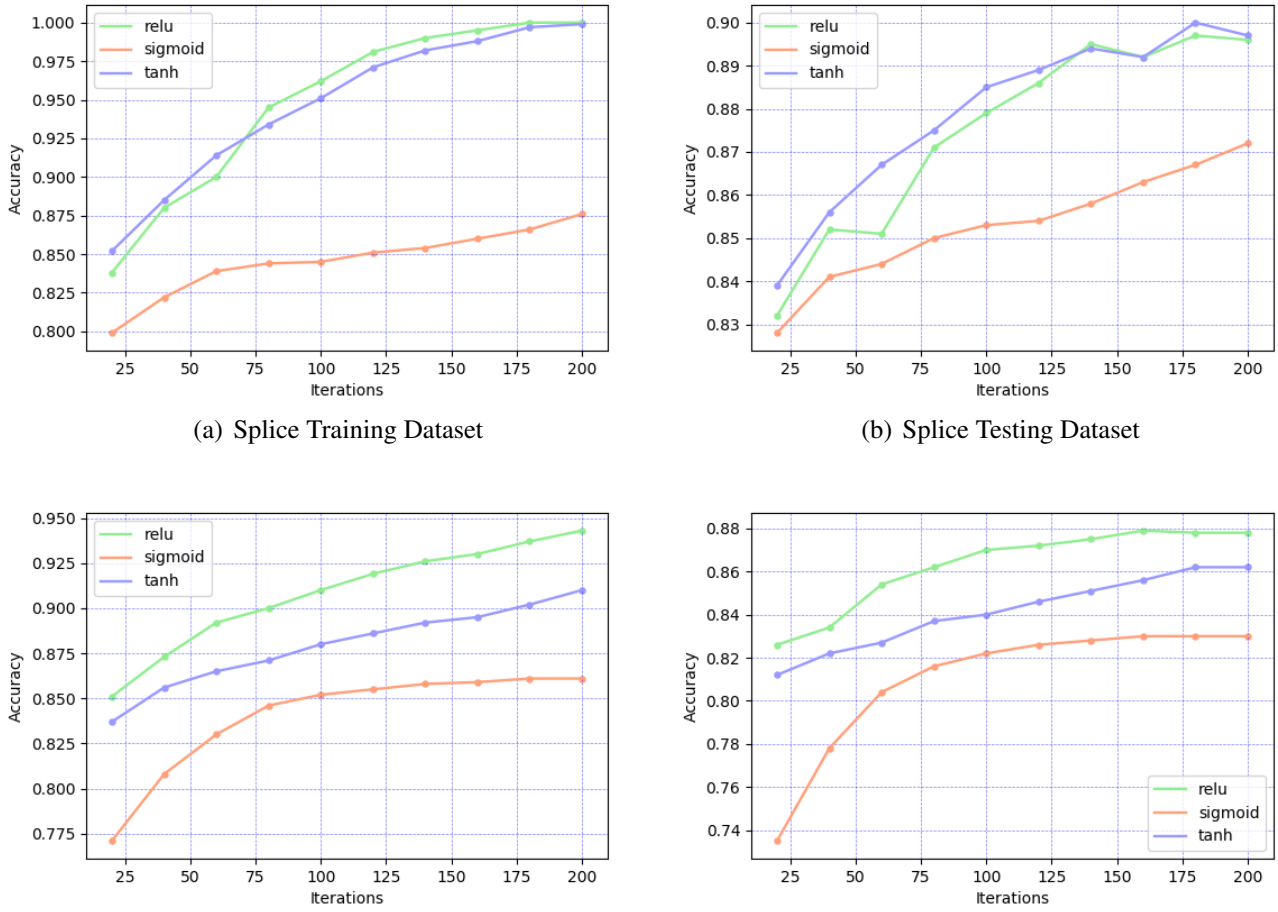
(b) Satimage

Figure 2: Performance of SVM with different penalty parameters.

1.3.2 MLP

Figure. 3 shows the performance difference of MLP with various activation functions. Fig. 4 illustrates the influence of learning rate on neural network models. Fig. 5 demonstrates the performance difference of MLP with different network structures.

As shown in Fig. 3, the effects of different activation function are widely divergent, especially on `satimage`. In most cases, "relu" is undoubtedly the best choice. "tanh" has a close performance to "relu" on `Splice` but performs worse on `Satimage`. "sigmoid" has poor performance on either datasets.



(a) Splice Training Dataset

(b) Splice Testing Dataset

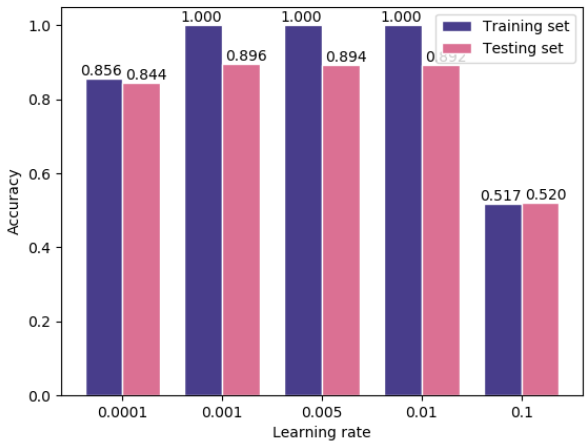
(c) Satimage Testing Dataset

(d) Satimage Testing Dataset

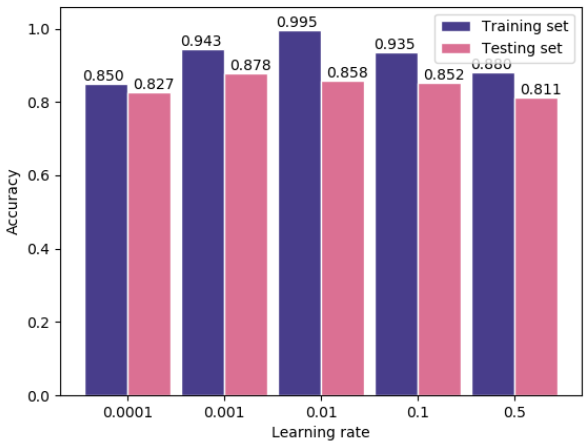
Figure 3: Performance of MLP with different activation functions.

In training process, learning rate controls the step size of parameters update. Adjusting learning rate is always time-consuming and critical due to its large impact on the final performance and sometimes the ability of convergence. The results show that 0.001 can provide good performance for both `Splice` and `Satimage`. When the learning rate reaches a certain point, the performance can drop dramatically as the learning rate increases.

Network structure is always an import part for the overall model performance. However, in my experiment, the performance are almost the same after about 25 iterations. One possible reason is that the dataset is relatively small and the number of neurons is not a dominant factor in determining the model performance.

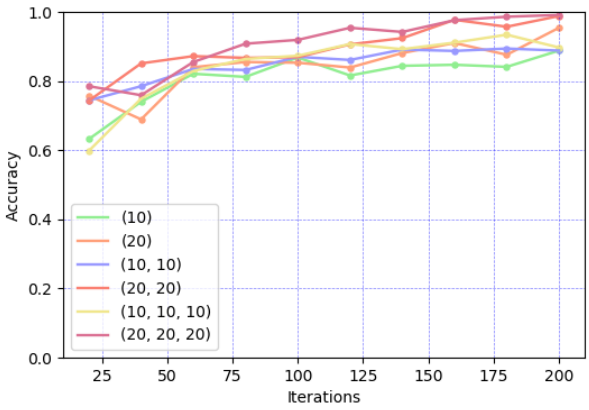


(a) Splice

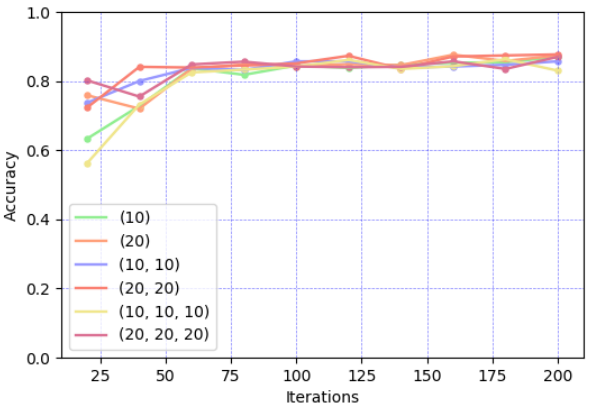


(b) Satimage

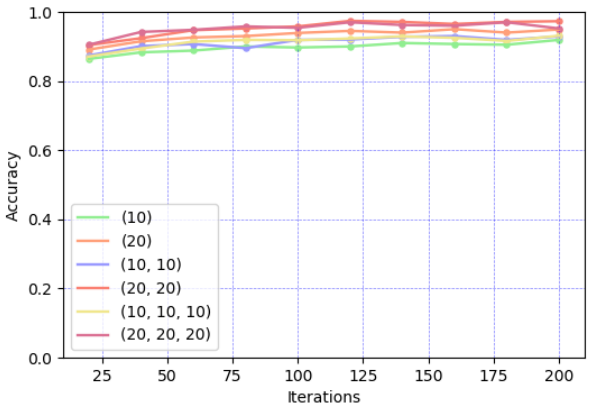
Figure 4: Performance of MLP with different learning rate.



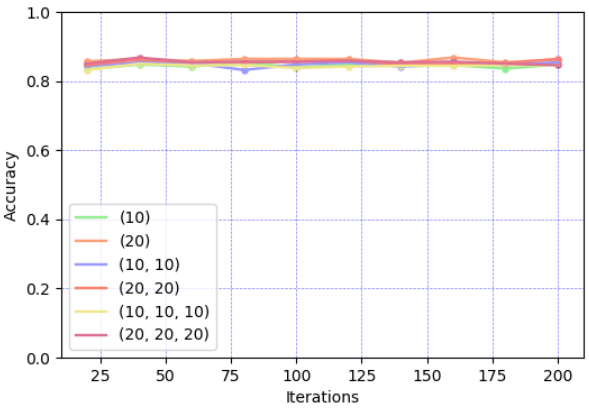
(a) Splice Training Dataset



(b) Splice Testing Dataset



(c) Satimage Testing Dataset



(d) Satimage Testing Dataset

Figure 5: Performance of MLP with different network architectures.

2 SVM vs. Deep Learning

2.1 Dataset

In this section we will carry out experiment with a large dataset. Here we choose the famous MNIST dataset.

2.2 Experiment

To reduce computation complexity, we simply try the SVM model without kernel. Fig. 6 illustrates part of the predictions made by the trained Linear SVM model. Tab. 6 gives an overall comparison between linear SVM.

Figure 6: Predictions made on the first 30 testing samples by the Linear SVM model.

2.3 Strengths and Weaknesses of SVM on Big Dataset

Below are several of the strengths and weaknesses of SVM on big dataset.

Table 6: Performance Comparisons for MNIST Dataset.

Algorithms	Test Error (Training Time)
Linear SVM	0.0897, 11 min
2-layer NN, 300 hidden units, mean square error	0.047
3-layer NN, 500+300 HU, softmax, cross entropy, weight decay	0.0153
6-layer NN 784-2500-2000-1500-1000-500-10 (on GPU) [elastic distortions]	0.0035
Convolutional net LeNet-4	0.011
committee of 35 conv. net, 1-20-P-40-P-150-10 [elastic distortions]	0.0023

1. The support vectors are solved via quadratic programming, which is highly time-consuming and memory-consuming when the dataset is large (more than 10000 samples with high dimensional features). Suppose the number of samples and features are n and m respectively, the complexity of SVM can be $O(mn^2)$ when using kernel methods.

2. The classical SVM mainly deals with binary classification problems, although it is able to act on multi-class classification problems with the help of one-vs-one or one-vs-rest method, the performance is relatively poor.

3 Causal Discovery Algorithm

3.1 Problem Background

It is an important problem in bank risk management to predict the customers' default payment. Usually, the probability of a default payment is influenced by many factors such as age, education, previous bill statement, previous paid bill and credit. It is helpful to figure out some causalities among these factors and we hence carry out causal discovery on these factors.

3.2 Dataset

In this experiment we use the dataset "Default of Credit Card Clients" from the UCI Machine Learning Respository*, which is with mixed data type of "discrete" and "continuous". To make computation easier, 3000 items are sampled from the previous dataset and 17 variables are selected. Details on the variables are shown in Tab. 7.

3.3 Methods

We use the tool package provide in the Tetrad project† and try several different algorithms: FCI, GFCI, RFCI and variants of PC.

3.4 Experiment Result

Fig. 8 shows the causal graph generated by these four algorithms. The meanings of the edge types are shown in Fig. 7.

*https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

†http://www.phil.cmu.edu/ tetrad/

Table 7: Details on Dataset Variables.

Name	Description
ID	The id of the client.
Default Payment Next Month	Yes = 1, No = 0
SEX	1 = male; 2 = female
Education	1 = graduate school; 2 = university; 3 = high school; 4 = others.
Marriage	1 = married; 2 = single; 3 = others
Age	Year
PAY_0-6	History of past payment. -1 = pay duly; n = payment delay for n month.
LIMIT_BAL	Amount of the given credit (NT dollar).
BILL_AMTn	Amount of bill statement in Sep, 2005, Aug, 2005...
PAY_AMTn	Amount paid in Sep, 2005, Aug, 2005...

Graph Edge Types

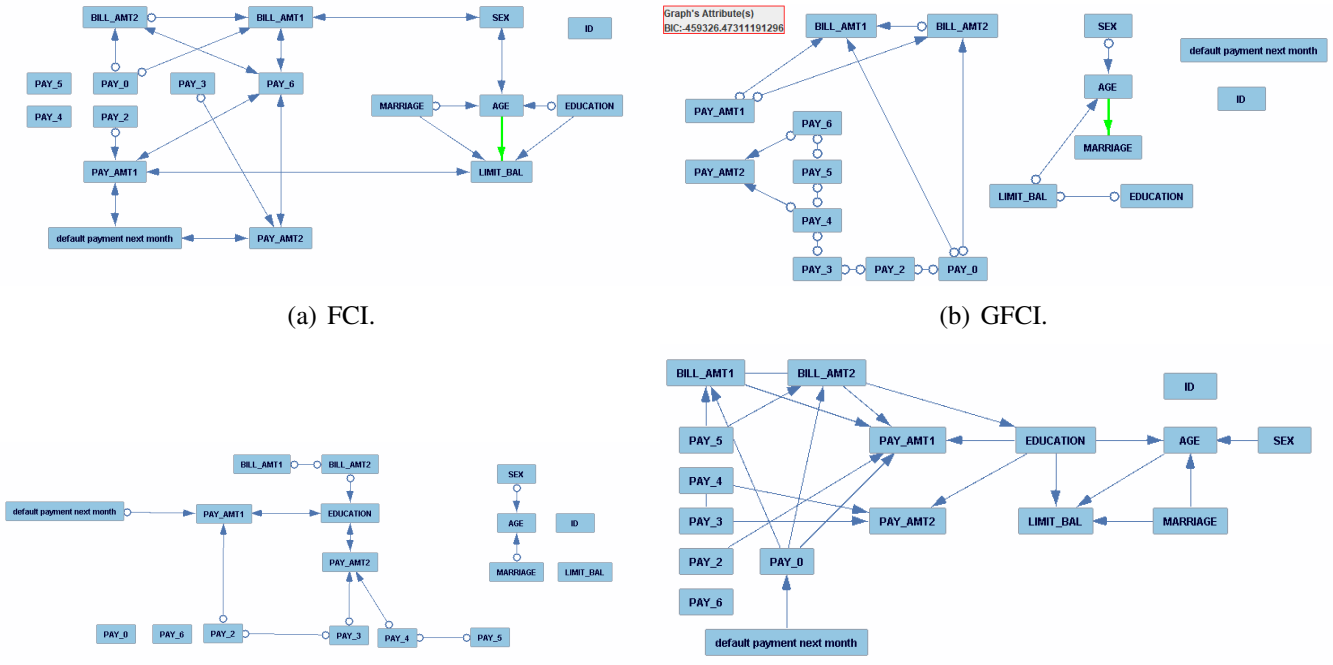
Edge Types	Present Relationships	Absent Relationships
A --> B	A is a cause of B. It may be a direct or indirect cause that may include other measured variables. Also, there may be an unmeasured confounder of A and B.	B is not a cause of A
A <-> B	There is an unmeasured confounder (call it L) of A and B. There may be measured variables along the causal pathway from L to A or from L to B.	A is not a cause of B. B is not a cause of A.
A o-> B	Either A is a cause of B (i.e, A --> B) or there is an unmeasured confounder of A and B (i.e, A <-> B) or both.	B is not a cause of A.
A o-o B	Exactly one of the following holds: 1. A is a cause of B 2. B is a cause of A 3. there is an unmeasured confounder of A and B 4. both a and c 5. both b and c	

If an edge is green that means there is no latent confounder. Otherwise, there is possibly latent confounder.

If an edge is **bold (thickened)** that means it is definitely direct. Otherwise, it is possibly direct.

Figure 7: The meanings of the edge types in the graph.

From Fig. (b) and Fig. 8(c), we can see that the generated graphs are separated by the data type of the nodes: most of the discrete variables are connected to the continuous variables. In Fig. 8(b) especially, we can see that the default payment next month is not connected to any other nodes in the graph, which is against our common sense and it is not reasonable.



(a) FCI. (b) GFCI.

(c) RFCI. (d) Variants of PC.

Figure 8: Causal Graphs Generated by Tested Algorithms.

In all, FCI has generated a relatively more reasonable result: the default payment next month is mutually influenced with the amount of bill paid in previous months: the more the amount is paid, the larger the probability of default payment this month is. Moreover, as we can see from Fig. (a), the client credit (LIMIT_BAL) is the end product of marriage, education and age, which is in line with our common sense. At last, FCI also performs well in determining the causality among PAY_n, PAY_AMTn and BILL_AMTn: if a client failed to pay duly the previous month, the less the amount paid this month will be and the larger the bill statement this month will be. An interesting finding is that there is an unmeasured confounder of SEX and BILL_AMT, which we may simply explain as stronger desire to purchase of women compared with men. However, there is inevitably some problematic causality discovered, such as the confounder relationship between sex and age.