**CHAPTER Ensemble Based Causal Discovery**

**Introduction ----------------------------------------------------------------------------**

Our causal discovery analysis is based on a dataset of 78 records, the statistical units of our analysis. Each unit consists of to a ground truth (GT) causal model represented by a Direct Acyclic Graph (DAG) and of the 5 DAGs reconstructed by as many causal discovery algorithms: GES, GOLEM, LINGAM. Notears and PC.

The data had been obtained by generating a large number of examples using a GT DAG topology with different numbers of nodes (from 9 to 30) and edges (from 20 to 450), with various functional dependencies between parent and child nodes (including Linear, and Quadratic), and with various probability distributions of the exogenous, i.e. parent-less, variables (such as Gaussian, Exponential, Uniform, and Gumbel). More details on the data generation are in the Appendix.

Individually, the algorithms are not very efficient in reconstructing the GT: the efficiency runs from the about 10% of GES to the about 47% of GOLEM. However at least one of the five algorithms reconstructs perfectly the GT about 64% of the times (for each GT case, the average number of correct predictions is 1.76).

Another way of comparing the results with the GT is through measuring the distance between DAGs defined as the Hamming distance between the respective adjacency matrices (i.e. in terms of oriented edges): the average distance between the algorithms and the ground truth varies from the 25.62 edges of PC to the 10.59 edges of LINGAM. However, if we look at the performances case by case, and take the DAG closest to the ground truth, we find that the average distance of the best performer is shrank by a factor two (it is 5.39).

The improvement is even more dramatic if we normalize the distance with respect to the number of actual edges of the GT DAG: while the best performing individual algorithm is LINGAM with a standardized distance of 0.024, looking at the performances case by case and picking the best we get an average standardized distance an order of magnitude smaller (it is 0.004).

Thus there are potential improvements in trying to select case by case the best performing algorithm(s) of the ensemble (notice that there could be more than one reaching the same distance from the GT DAG).

We want to learn from the data a criterion able to extract this knowledge from the ensemble.

Intuitively a good amount of information on the correctness of the respective guesses could come from the degree of agreement among the DAGS yielded by the different algorithms. The DAGs relationships can be captured by the collection of their pairwise distances (according to a suitable distance definition specified below, relevant in the causal context), but also, almost equivalently by their individual distances with respect to a centroid DAG (defined as specified below) together with the total distance with all the others:

the dispersion of the set of five DAGs with respect to the centroid can inform on the level of agreement among algorithms, while the total distances can inform on the outlying predictions.

With GT DAGs available this information can be used to train a ML predictor model  
trying to tell which algorithms (if any) have correctly predicted the GT DAG, or which are the closest to it.

**A causality oriented definition of distance**

Since we are investigating the performance of causal discovery algorithms a more appropriate definition of agreement between two DAGs should be based on causal concepts, not only on adjacency matrix relationships: we defined a distance based on node pair d-separation: given a DAG each pair of nodes has been assessed for d-separation, and each DAG with $n$ nodes has been associated to a bitset of $n(n-1)/2$ bits, where each bit was set to one in case the two nodes of that pair were d-separated. The bitset were the used as signatures of the DAG for pairwise comparing the DAGs. The distance $d\_ij = d(DAG\_i,DAG\_j)$ between two DAGs $i$ and $j$ has been defined as the Hamming distance between their d-separation bitsets.

The distance based on the bit-set of the di-separated pairs is much more meaningful that the Hamming distance between the adjacency matrices. Indeed, reversing a single edge in one of two identical dags creates a difference of exactly two elements of the adjacency matrix, but in terms of d-separation can completely change the causal structure of the graph, affecting the d-separation of many pairs (as an illustrative example, consider two large subgraphs connected by a sequence pattern: reversing the second edge of the sequence transforms the sequence into a collider and d-separates all the pairs with one note in the first subgraph and one node in the second).

Thus, in the input data later used for training, we took the adjacency matrices information, transformed it into d-separation information for all the node pairs of a DAG and computed the Hamming distance between bitsets. However, notice that when comparing the predictions of the DAGs to the GT we used the Hamming distance between adjacency matrices, which informs directly about the topology of the graphs (i.e. measured the difference in terms of edge difference count).

**A definition of centroid DAG**

For each DAG i we computed the sum of the d-seperation bitset distances between it and the reminder DAGs, $s\_i=sum\_j d\_ij$. The centroid DAG has been defined as the one with the lowest sum of the distances $min\_i s\_i$. The DAGs corresponding to the centroid could be one or more.

The dispersion of the collection of DAGs has been assessed using both the maximum of $s\_i$ and the total sum of distances $t=sum\_i s\_i$.

**Learning form the DAGs relationships to to select the best performing DAG**

In this work, we contribute a causal discovery method that takes advantage the synthesis of the topology-related information about the relationships among graphs of an ensemble as captured by a suitably defined centroid graph to identify which topology is closest to the ground truth.

We will focus on answering the following questions.

**QUESTION 1**

Which algorithm or which algorithms have correctly predicted the GT DAG?

**QUESTION 2**

Which algorithm is the closest to the GT DAG?

We will find a method for addressing Question 2, and then look at the thus selected candidate performance in answering to Question 1.

To this purpose we use a supervised learning method.

Notice that the DAGs generated by the discovery algorithms can sometimes coincide with one another, whether they correspond to the ground truth or not: thus, there is not necessarily a single correct answer to the question of which is the ground truth or which is the algorithm closest to the ground truth. As a consequence, we formulate the problem as a multi-label classification task (each algorithm plays the role of a label, there are also $6$ labels, counting the CENTROID graph). Then we use two common approaches to address this type of classification: the Binary Relevance (BR) approach and the Classifier Chain (CC) approach. After choosing a representative from the set of predicted labels, using a classifier-precision-based criterion, we used its distance from the GT (defined as the Hamming distance between adjacency matrices, i.e. in terms of edges).

**VARIABLES ----------------------------------------------------------------------------------------**

**Dataset generation related variables**

The variables characterizing each dataset were those described in the following, only the ones with a name not ending by an underscore were available to the training algorithms.

Number of nodes (NNODES, integer), Number of possible pairs (NPOSSIBLE\_PAIRS, integer), Number of actual edges (NACTUAL\_EDGES, integer), Linear or non-linear functional dependence (Linear\_, possible values “linear” and “nonlinear”), Statistical distribution used for the extrinsic variables (Distribution)

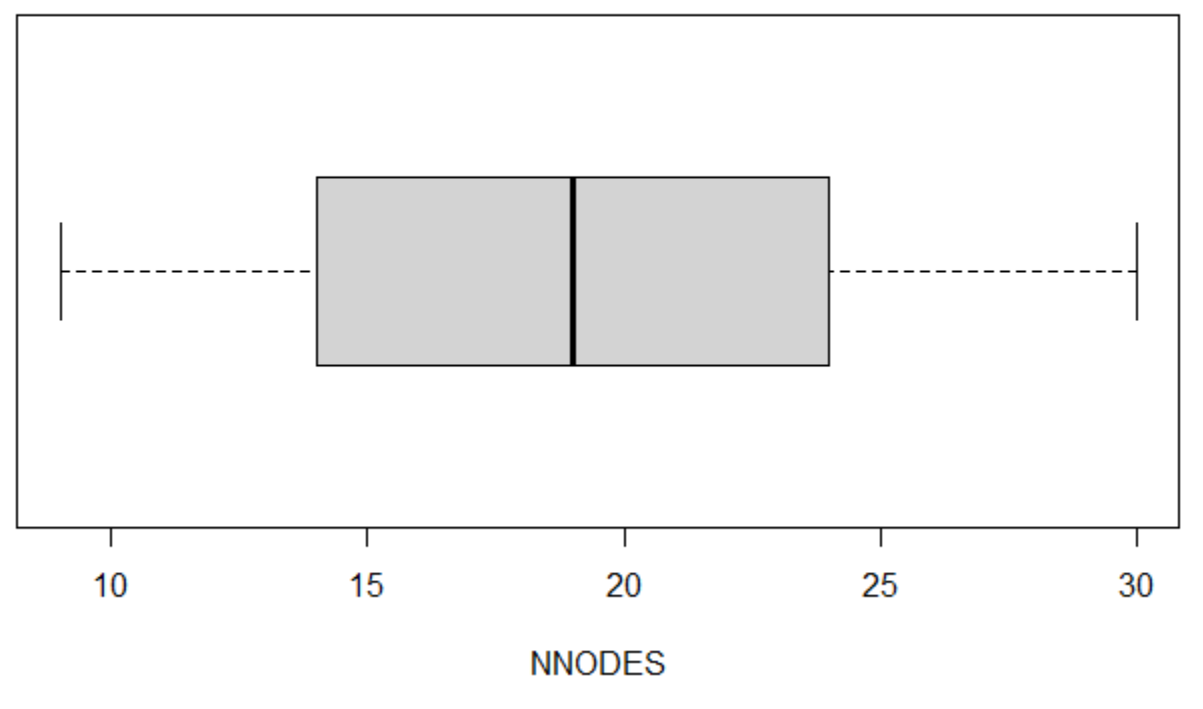


Figure: Boxplot of the number of nodes NNODES.

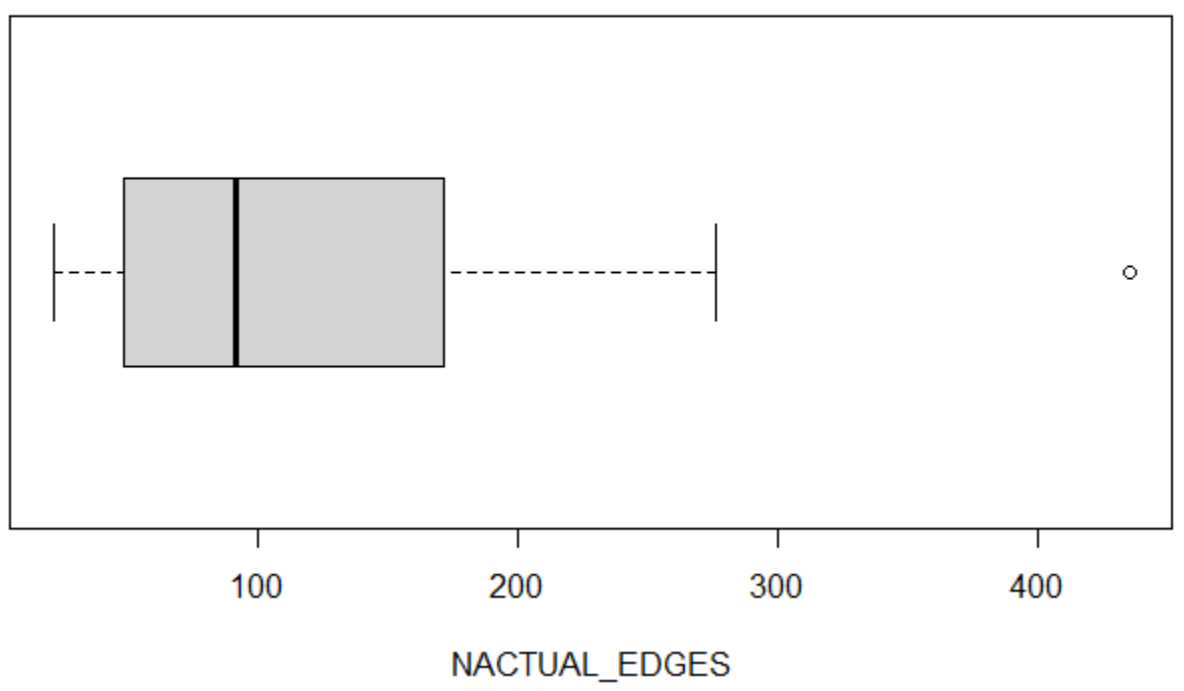


Figure: Boxplot of the number of nodes NACTUAL\_EDGES.

**Unlabeled part of the dataset,**

This part of the dataset contains, for each statistical unit, those variables describing the relationships among the DAGS as reconstructed by the five causal discovery algorithms (at least one of which is identified as centroid).

The variables describing those DAG distances and their quality of centroid are the following (notice that [ALGO] can take the values GES, GOLEM, LINGAM, Notears, PC):

**[ALGO]\_SUM\_DIST (integer): sum of the hamming distances between the DAG yielded by the [ALGO] algorithm and the other DAGs**

**[ALGO]\_SUM\_DIST\_STAND (real): previous variable divided by NPOSSIBLE\_PAIRS**

**[ALGO]\_IS\_CENTROID (Boolean): whether the DAG is the centroid**

Then we had variables describing the ensemble of causal discovery DAGS

**CENTOIDS\_COUNT (positive integer): number of algorithms taking the centroid role (from 1 to 5)**

**MAX\_SUM\_DIST (integer): maximum of the sums of distances across DAGs**

**MAX\_SUM\_DIST\_STAND (real): previous variable divided by NPOSSIBLE\_PAIRS**

**CENTROID\_SUM\_DIST (integer): sums of distances for the centroid**

**CENTROID\_SUM\_DIST\_STAND (real): previous variable divided by NPOSSIBLE\_PAIRS**

**Labels and GT related variables**

A number of variables were used to describe the relationship of the individual algorithms DAGs to the GT DAG. We recall that the distances were computed as Hamming distances between the adjacency matrices respectively of [ALGO] and GT. Notice that her the standardization is performed using the actual number of edges present in the GT DAG, no longer using the number of possible node pairs.

Here the values taken by [ALGO] were the 5 algorithms in the set {GES, GOLEM, LINGAM, Notears, PC} plus CENTROID. Typically, one or more of these variables, which not available to the learned predictor, are used as predictor targets.

**[ALGO]\_GT\_DIST\_ (integer): see above**

**[ALGO]\_GT\_DIST\_STAND\_ (real): previous variable divided by NACTUAL\_EDGES**

**[ALGO]\_IS\_GT\_ (Boolean): whether [ALGO] has made a correct prediction**

Then we had variables describing the ensemble of causal discovery DAGS

**NUM\_OF\_CORRECT\_ (integer): n. of correct predictions by the set of 5 algorithms**

**MIN\_DIST\_TO\_GT\_ (integer): minimum distance, when 0 the prediction was correct**

**MIN\_DIST\_TO\_GT\_STAND\_ (real): previous variable divided by NACTUAL\_EDGES**

**[ALGO]\_IS\_MIN (Boolean): whether [ALGO]’s prediction corresponds to the minimum distance from GT**

**MAX\_DIST\_TO\_GT\_ (integer): maximum distance**

**MAX\_DIST\_TO\_GT\_STAND\_ (real): previous variable divided by NACTUAL\_EDGES**

We want to find whether the knowledge from the ensemble can be extracted so as to obtain ensemble predictions which are better than the individual algorithms’ predictions, and in affirmative case we want to find out what level of improvement can be achieved.

We will check in a preliminary analysis that at least an algorithm (out of the 5 considered) made a DAG prediction coinciding with the GT: the answer will turn out to be positive in about two thirds of the times.

**DESCRIPTIVE ANALYTICS ----------------------------------------------------------------------------**

**Relationships among DAGs, not considering the Ground Truth**

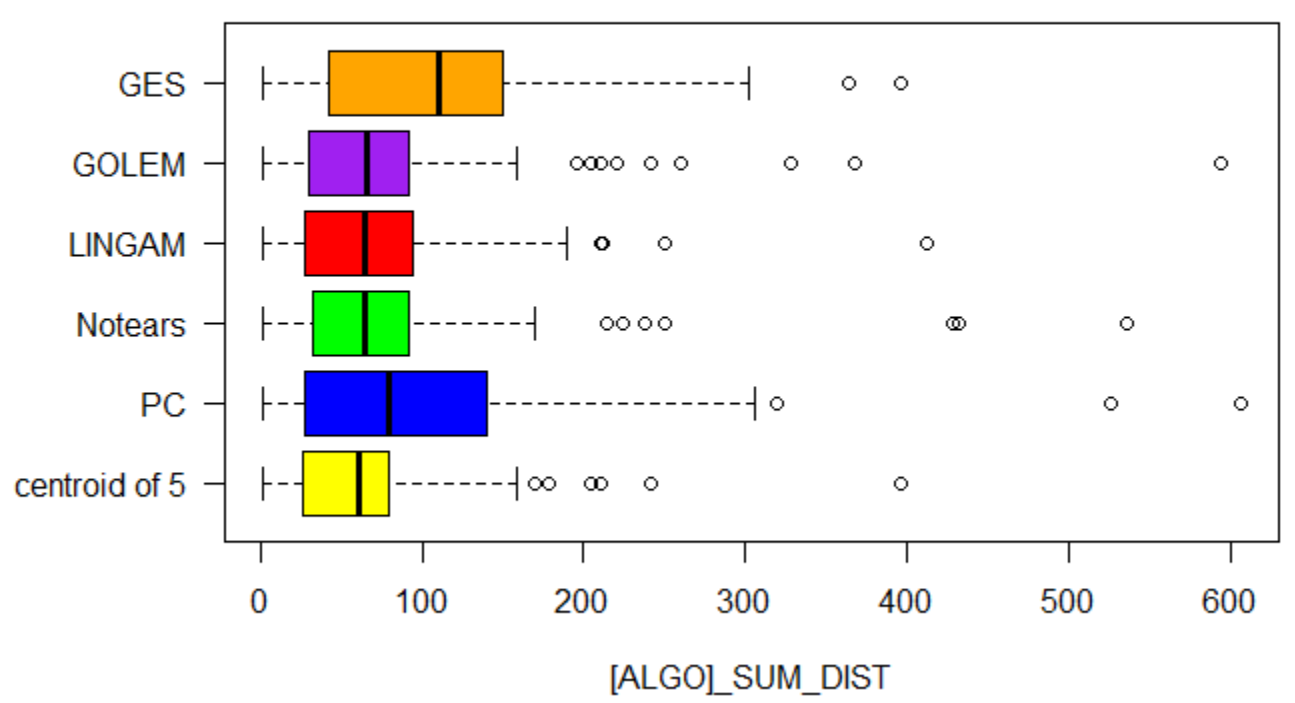


Figure: Boxplot of [ALGO]\_SUM\_DIST.

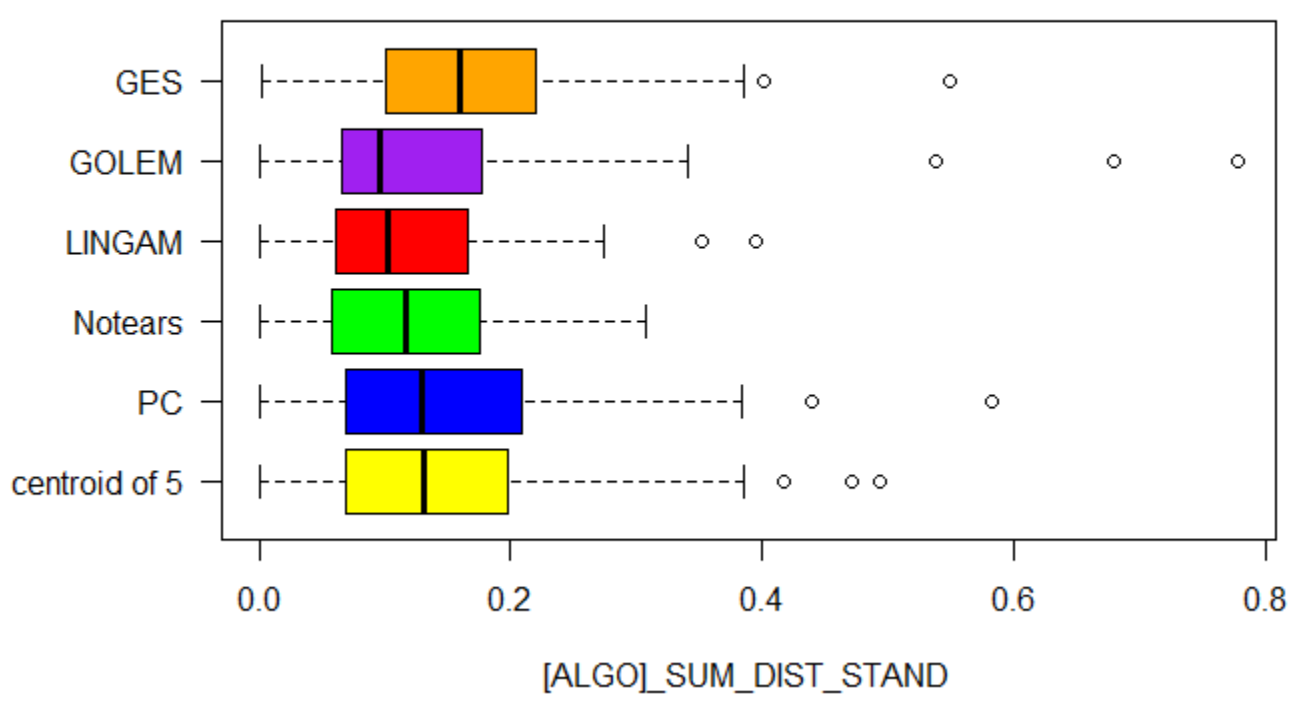


Figure: Boxplot of [ALGO]\_SUM\_DIST\_STAND.

Here are the percentages of the datasets for which the algorithm is the CENTROID:

GES 15%, GOLEM 63%, LINGAM 63%, Notears 45%, PC 28%.

There is often more than one centroid.

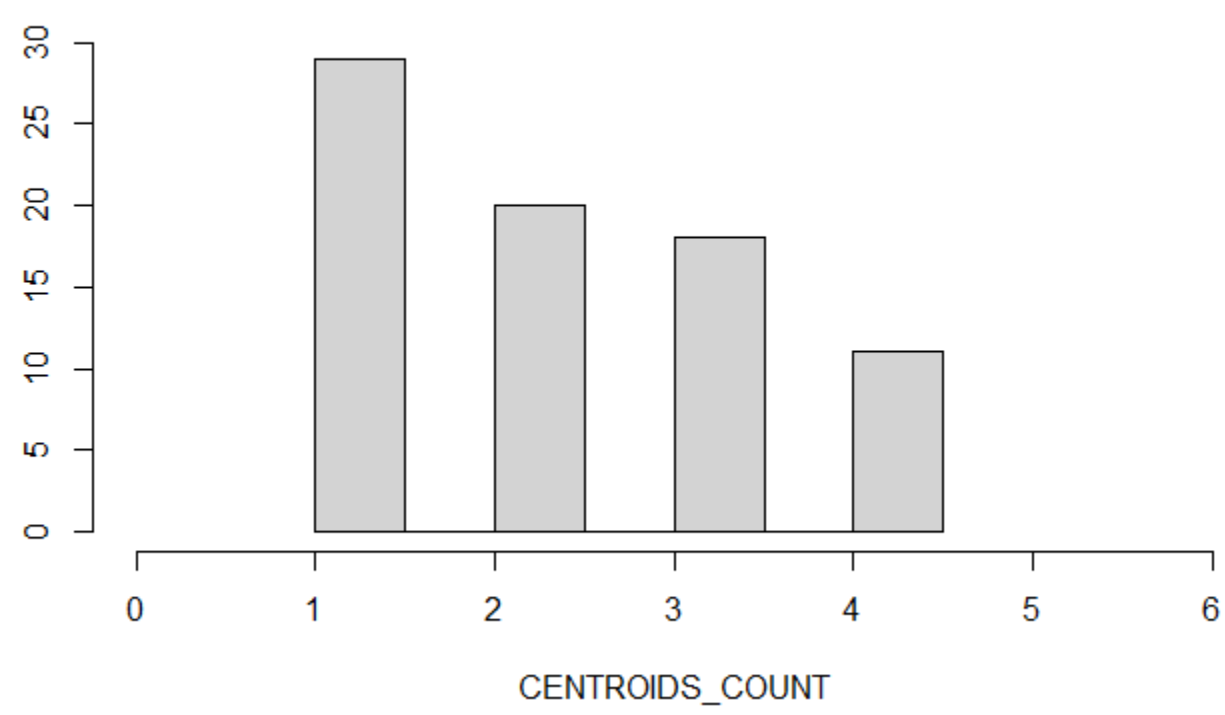


Figure: Histogram of CENTOID\_COUNT

**Descriptive Data Analytics – Features of the Ensemble relative to the Ground Truth**

Here we report about the relationship between the DAGs issued by the different algorithms and the GT DAGs. This helps contextualize each question: from the data it is clear that there are potentialities in the exploitation of the knowledge of the ensemble.

**Boolean variable [ALGO]\_IS\_GT\_: Correct/incorrect reconstruction of the GT DAG**

The different algorithm provides the perfectly correct solution with the following percentages. In the same table is reported the number of times at least one algorithm makes a perfectly correct prediction (for convenience the rightmost column anticipates also the results of our method, which will be discussed later).

Table {XXX\_CORRECT}

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **% [ALGO]\_IS\_GT\_** | **GES** | **GOLEM** | **LINGAM** | **Notears** | **PC** | **CENTROID** | **At least 1 correctly predicts GT** | **Algorithm BR selected** | **Algorithm BR selected** |
| **n.of times True** | 8 | 37 | 35 | 28 | 11 | **38** | **50** | **48** | **49** |
| **n.of cases** | 78 | 78 | 78 | 78 | 78 | **78** | **78** | **78** | **78** |
| **Percentage True** | **10.26** | **47.44** | **44.87** | **35.90** | **14.10** | **48.72** | **64.10** | **61.54** | **62.82** |
| **stderr** | 3.55 | 5.84 | 5.82 | 5.61 | 4.07 | **5.85** | **5.61** | **5.69** | **5.66** |
| **Lower end 95% interval** | 3.30 | 35.98 | 33.46 | 24.89 | 6.12 | **37.25** | **53.10** | **50.38** | **51.73** |
| **Lower end 95% interval** | 17.22 | 58.89 | 56.28 | 46.90 | 22.09 | **60.18** | **75.11** | **72.70** | **73.91** |

\begin{tabular}{|l|r|r|r|r|r|r|r|r|r|}

\hline$\%$ [ALGO]\_IS\_GT & GES & GOLEM & LINGAM & Notears & PC & CENTROID & \begin{tabular}{r}

At least 1 correctly \\

predicts GT

\end{tabular} & \begin{tabular}{l}

Algorithm \\

BR selected

\end{tabular} & \begin{tabular}{l}

Algorithm \\

BR selected

\end{tabular} \\

\hline n. of times True & 8 & 37 & 35 & 28 & 11 & 38 & 50 & 48 & 49 \\

\hline n. of cases & 78 & 78 & 78 & 78 & 78 & 78 & 78 & 78 & 78 \\

\hline Percentage True & 10.26 & 47.44 & 44.87 & 35.90 & 14.10 & 48.72 & 64.10 & 61.54 & 62.82 \\

\hline stderr & 3.55 & 5.84 & 5.82 & 5.61 & 4.07 & 5.85 & 5.61 & 5.69 & 5.66 \\

\hline \begin{tabular}{l}

Lower end $95 \%$ \\

interval

\end{tabular} & 3.30 & 35.98 & 33.46 & 24.89 & 6.12 & 37.25 & 53.10 & 50.38 & 51.73 \\

\hline \begin{tabular}{l}

Lower end $95 \%$ \\

interval

\end{tabular} & 17.22 & 58.89 & 56.28 & 46.90 & 22.09 & 60.18 & 75.11 & 72.70 & 73.91 \\

\hline

\end{tabular}

One can observe that in terms of the number of correct guesses (i.e. in terms of precision) CENTROID prevails over the other, but the difference w.r.t. the second best, here GOLEM, and w.r.t LINGAM and Notears, is **not statistically significant**. GES and PC are significantly less performing than the others.

No individual algorithm among the thee best -- GOLEM, LINGAM and Notears -- outperforms overall the others, nor selecting the centroid DAG is in average significantly better that the individual algorithms.

Nonetheless from the table one can see that in principle there is room for improving the results, by a suitable case by case selection of the algorithm. If an oracle were able to point to the best algorithm(s) for a specific dataset one would achieve a 64.10% precision. Thus there is room for approximately a 15% improvement.

**Distance variables**

The picture gets more precise as we pass from the Boolean variable [ALGO]\_IS\_GT\_ to the distance variables [ALGO\_GT\_DIST\_].

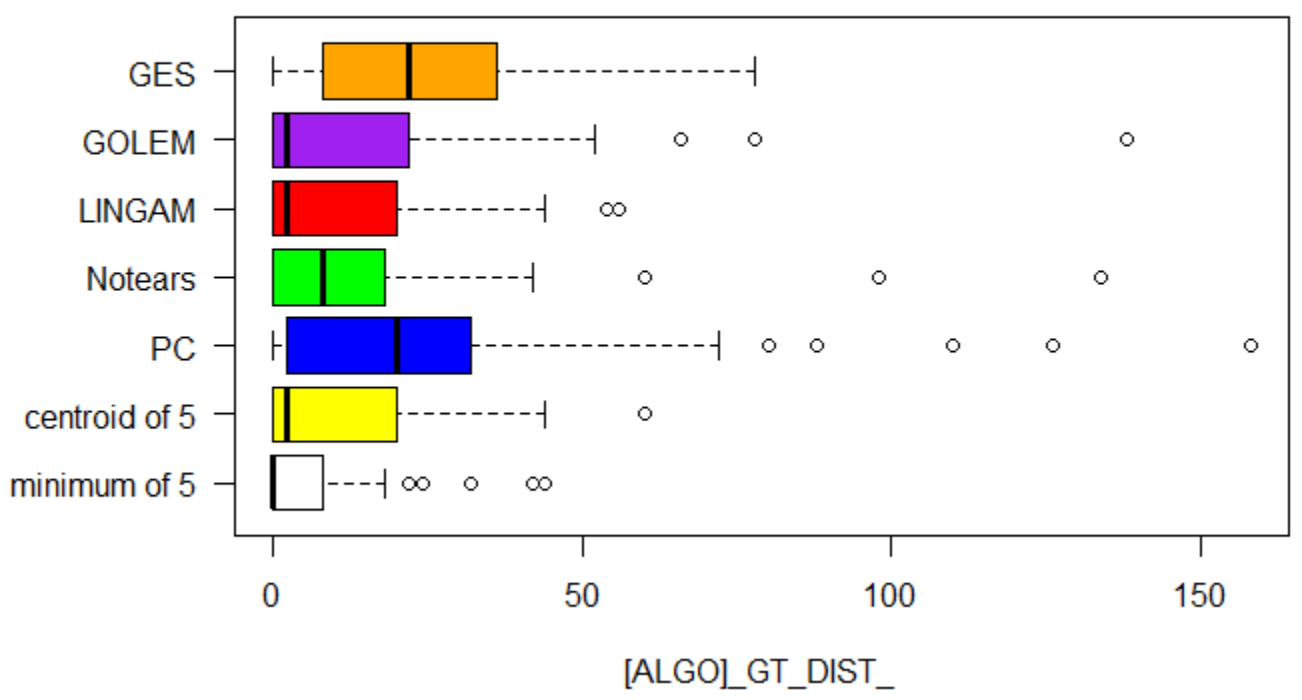


Figure: Boxplot of [ALGO]\_GT\_DIST.

Looking at the box plot of [ALGO]\_GT\_DIST one can see that GOLEM, LINGAM and CENTOID are again the best performing algorithms. In particular CENTROID seems to slightly outperform the others.

The numerical summary of the variables is the following (for convenience the two rightmost columns anticipates also the results of our method, which will be discussed later).

Table {XXX\_DIST}

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **[ALGO] \_GT\_DIST\_** | **GES** | **GOLEM** | **LINGAM** | **Notears** | **PC** | **CENTROID** | **Distance from GT of the closest algorithm** | **Distance from GT of BR selected** | **Distance from GT of CC selected** |
|  |  |  |  |  |  |  |  |  |  |
| **Min:** | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | **0.00** | 0.00 | **0.00** | **0.00** |
| **lst Qu.:** | 8.50 | 0.00 | 0.00 | 0.00 | 2.50 | **0.00** | 0.00 | **0.00** | **0.00** |
| **Median :** | 22.00 | **2.00** | **2.00** | 8.00 | 20.00 | **2.00** | 0.00 | **0.00** | **0.00** |
| **Mean :** | 24.49 | 13.41 | **10.59** | 13.08 | 25.62 | **10.08** | **5.38** | **6.62** | **6.36** |
| **3rd Qu.:** | 36.00 | 21.50 | 20.00 | 18.00 | 32.00 | **19.00** | 8.00 | **9.50** | **8.00** |
| **Max:** | 78.00 | 138.00 | 56.00 | 134.00 | 158.00 | **60.00** | 44.00 | **44.00** | **44.00** |
|  |  |  |  |  |  |  |  |  |  |
| **Std:** | 19.50 | 22.57 | 14.76 | 20.84 | 29.21 | **14.34** | 10.36 | **11.40** | **11.36** |
| **Stderr:** |  |  | 1.67 |  |  | **1.62** | 1.17 | **1.29** | **1.32** |
| **Lower end 95% interval** |  |  | **7.31** |  |  | **6.89** | 3.09 | **4.09** | **3.78** |
| **Lower end 95% interval** |  |  | 13.87 |  |  | **13.26** | 7.68 | **9.14** | **8.94** |

\begin{tabular}{|l|r|r|r|r|r|r|r|r|r|}

\hline [ALGO]\_GT\_DIST\_ & GES & GOLEM & LINGAM & Notears & PC & CENTROID & \begin{tabular}{l}

Distance from GT \\

of the closest \\

algorithm

\end{tabular} & \begin{tabular}{l}

Distance from GT \\

of BR selected

\end{tabular} & \begin{tabular}{l}

Distance from GT \\

of CC selected

\end{tabular} \\

\hline Min: & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\

\hline Ist Qu.: & 8.50 & 0.00 & 0.00 & 0.00 & 2.50 & 0.00 & 0.00 & 0.00 & 0.00 \\

\hline Median : & 22.00 & 2.00 & 2.00 & 8.00 & 20.00 & 2.00 & 0.00 & 0.00 & 0.00 \\

\hline Mean : & 24.49 & 13.41 & 10.59 & 13.08 & 25.62 & 10.08 & 5.38 & 6.62 & 6.36 \\

\hline 3rd Qu.: & 36.00 & 21.50 & 20.00 & 18.00 & 32.00 & 19.00 & 8.00 & 9.50 & 8.00 \\

\hline Max: & 78.00 & 138.00 & 56.00 & 134.00 & 158.00 & 60.00 & 44.00 & 44.00 & 44.00 \\

\hline Std: & 19.50 & 22.57 & 14.76 & 20.84 & 29.21 & 14.34 & 10.36 & 11.40 & 11.36 \\

\hline Stderr: & & & 1.67 & & & 1.62 & 1.17 & 1.29 & 1.32 \\

\hline \begin{tabular}{l}

Lower end $95 \%$ \\

interval

\end{tabular} & & & 7.31 & & & 6.89 & 3.09 & 4.09 & 3.78 \\

\hline \begin{tabular}{l}

Lower end $95 \%$ \\

interval

\end{tabular} & & & 13.87 & & & 13.26 & 7.68 & 9.14 & 8.94 \\

\hline

\end{tabular}

One can see that the difference in means between CENTOID and the second best **is not statistically significant** already from the fact that the two means lie largely inside each other’s 95% confidence intervals. Performing a one tailed Student-t Test one finds out that the two means are not significantly different from one another at the 0.05 confidence level: indeed, the p-value turns out to be 0.4130. So we cannot reject the hypothesis that the observed prevalence of CENTROID over the second best is just due to chance.

Similar considerations hold for the variable [ALGO]\_GT\_DIST\_STAND\_, obtained from [ALGO]\_GT\_DIST dividing by the number of actual edges (for the sake of clarity we are not reporting the standardized distance data here). The boxplots for the variable [ALGO]\_GT\_DIST\_STAND\_ are shown in the following figure.

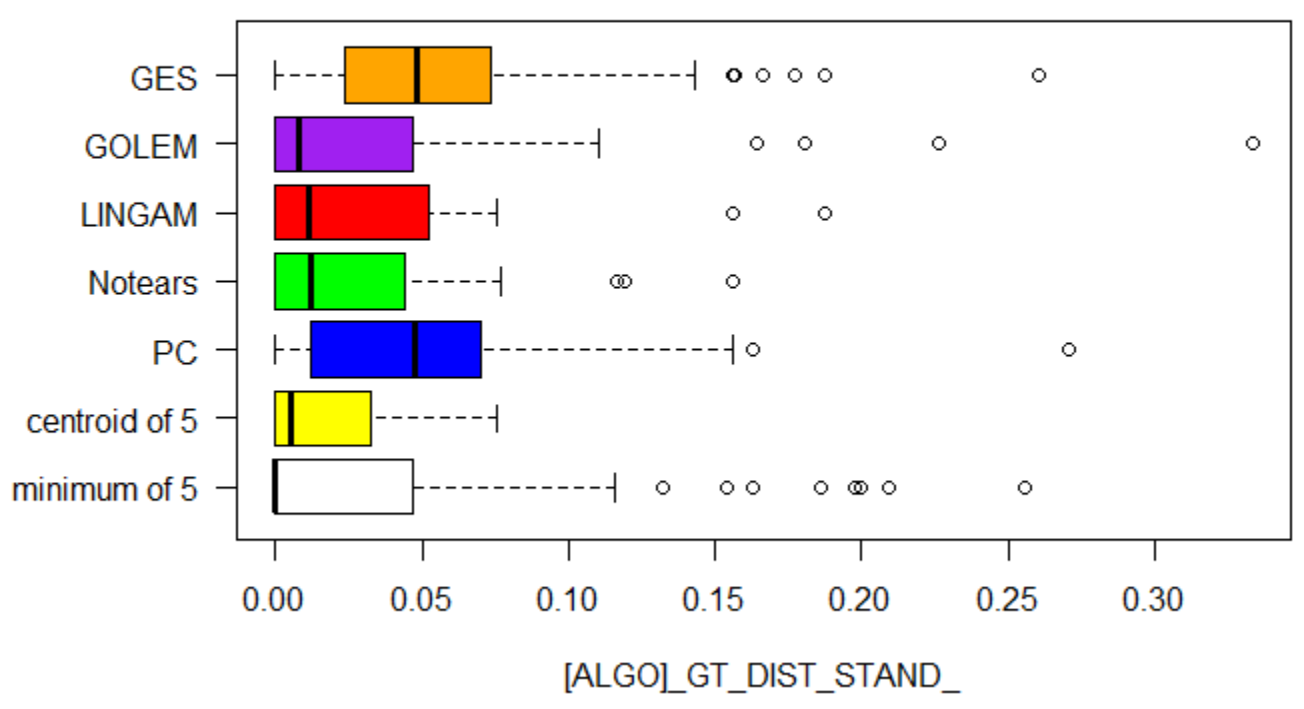


Figure: Boxplot of [ALGO]\_GT\_DIST\_STAND.

Therefore, neither individual algorithm outperforms in average the others, nor a simple rule such as the one of selecting the centroid DAG is in average significantly better that the individual algorithms.

However, one can see that if one were able to select case by case the algorithm with the minimum distance from the GT, the performance would be significantly improved.

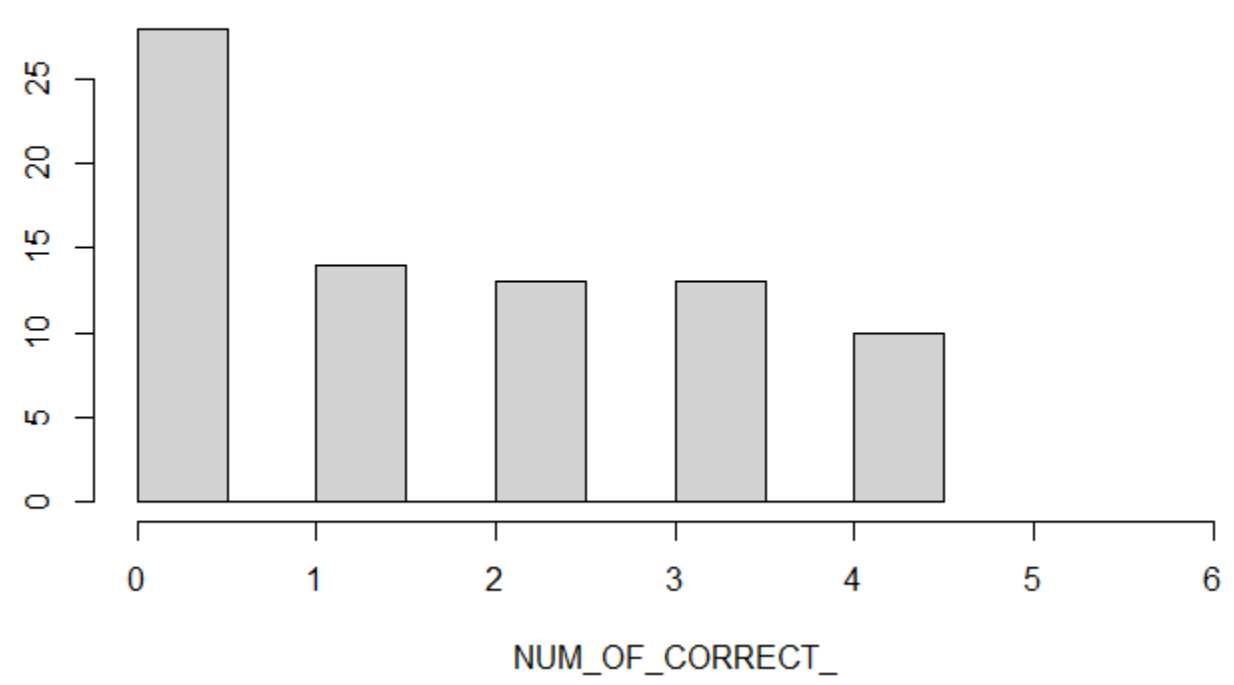


Figure: Histogram of the number of correct predictions

Looking at the number of times a specific algorithm turns out to be the closest to GT sheds further light on the performance of the candidates: LINGAM, Notears and CENTROID are the best performing.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **% [ALGO]\_IS\_MIN\_** | **GES** | **GOLEM** | **LINGAM** | **Notears** | **PC** | **CENTROID** |
| **n.of times True** | 10 | 44 | 43 | **49** | 12 | 48 |
| **n.of cases** | 78 | 78 | 78 | **78** | 78 | 78 |
| **Percentage True** | **12.82** | **56.42** | **55.13** | **62.82** | **15.38** | **61.54%** |
| **stderr** | 4.25 | 6.30 | 6.32 | **6.14** | 4.58 | 6.18 |

\begin{tabular}{|l|r|r|r|r|r|r|}

\hline [ [ALGO]\_IS\_MIN\_ & \multicolumn{1}{l|}{ GES } & \multicolumn{1}{|c|}{ GOLEM } & \multicolumn{1}{|c|}{ LINGAM } & \multicolumn{1}{|c|}{ Notears } & \multicolumn{1}{l|}{ PC } & CENTROID \\

\hline n.of times True & 10 & 44 & 43 & 49 & 12 & 48 \\

\hline n.of cases & 78 & 78 & 78 & 78 & 78 & 78 \\

\hline Percentage True & 12.82 & 56.42 & 55.13 & 62.82 & 15.38 & $61.54 \%$ \\

\hline stderr & 4.25 & 6.30 & 6.32 & 6.14 & 4.58 & 6.18 \\

\hline

\end{tabular}

Despite the prevalence of the three mentioned algorithms, there appears to be no simple strategy to select case by case an algorithm so as to **minimize the average distance** of the selected algorithm from the GT: for instance choosing always Notears, would get the minimum distance about 63% of the times, but in the other 37% of the times the resulting distance could be far from minimal, indeed (from the distance table) we know that the average distance of Notears would be 13.10, i.e. far above the theoretically allowed minimum of 5.38.

For these reasons address the task of selecting case by case which algorithm is the closest to GT given only the input variables by using a supervised ML approach.

**The ML task ----------------------------------------------------------------------------------------**

Given the input variables describing the relative d-separation-based distances between pairs of DAGS we want to select the algorithms/DAGS closest to the ground truth (possibly coinciding with it). Several DAGs at the same time have the same distance from the GT (or even coincide with it), thus several labels can coexist.

The problem can be cast in a multi-label classification problem, with the peculiarity that the prediction can be considered right if even just one of the correct labels is designated by the predictor.

To address the task, we set up two approaches:

* One built as a variant of Binary Relevance (BR)
* The other as a variant of Classifier Chain (CC)

BR works by training a separate binary classifier – sometimes called detector - for each label. Each classifier predicts whether the corresponding label applies or not.

CC works by training a separate binary classifier for each label, similar to BR, but also taking into account the predictions of previous classifiers in the chain: for each label in the chain, it trains a binary classifier using the input features and the binary labels of all preceding labels in the chain as additional input features. For example, when training the classifier for the third label in the chain, the input features would include the original input features as well as the binary predictions (0 or 1) of the first and second labels in the chain.

BR has the drawback that it does not incorporate possible label dependencies: in our case, if two algorithms yield the same value of [ALGO]\_SUM\_DIST their DAGS coincide (in the d-separation sense); despite this objective condition the two independently trained detectors could in principle predict inconsistent results (one predicting that the first label applies, the other that the second does not apply, despite both do).

CC has some issues too: Error Propagation and Order Sensitivity. If a classifier makes a mistake in predicting a label early in the chain, it can lead to cascading errors in the predictions of subsequent labels, potentially reducing overall performance. The choice of label ordering can impact the predictive performance of the model, while finding the optimal ordering can be computationally expensive (in our case 6 possible labels would entail 720 possible orderings). We address the latter point by ordering the chain of classifiers in decreasing order of accuracy (the accuracy was determined empirically when running BR, and we used the decreasing order of precision in case of break-even).

Both BR and CC produce for each case a set of labels that apply to that statistical unit. The variant we use in both cases consists in the final choice of a single representative of the predicted label set. Indeed, eventually, we want to take the output of the BR algorithm and choose exactly one of the labels (then take its distance to answer Question 2, and see whether it coincide with GT to answer Question 1).

The criterion we chose for selecting the [ALGO] among the candidates predicted by the individual detectors to be closest to the GT is based on precision and is described below.

**The AutoML (Automated Machine Learning) approach**

To choose an ideally optimal binary classifier we adopted an AutoML approach, for model selection and hyperparameter optimization, using the pycaret Python libraries (pycaret.org). This library automates the entire learning pipeline, from data preprocessing to model deployment, making it easier and faster to build high-performing classification models.

We chose to keep the default optimization strategy, aimed at maximizing the accuracy through Bayesian Optimization (BO). BO uses probabilistic modeling of the solution space to efficiently find the optimum of a costly black-box objective function (in our case the ML model test performance) by iteratively selecting the most promising candidates to evaluate (in our case the data subsets), while balancing exploration and exploitation.

**Selection criterion**

The selection of the [ALGO] among the candidate labels predicted by the detectors works as follows. We start looking at the [ALGO] whose detector has the highest test precision as evaluated by the optimizer (in case of break-even we use decreasing order of cardinality of the true positives), and if that [ALGO] is predicted True, we assign that label to the statistical unit; otherwise we look at the [ALGO] with second best precision, and so on.

If no [ALGO] detector predicts TRUE (i.e. if COUNT\_MIN\_PRED=0), we adopt as default the [ALGO] that performs best when all the other detectors predict FALSE. In our case the Notears detector significantly stands out as the best with respect to this performance criterion: in the 17 cases where a single [ALGO] is predicted to be at the minimum distance to GT, that [ALGO] corresponds to Notears 10 times, Lingam 5, Golem and CENTROID 1, GES and PC 0.

Thus we adopt Notears as default. This choice turns out to be the best 4 times out of 5 (5 are the statistical units where no detector predicts its [ALGO] to be the minimum distance one).

**Outcomes of Binary Relevance (BR) appraoch**

We searched for the optimal binary classifier detectors of each [ALGO] using an AutoML approach as described above. The configurations of the highest accuracy issued by the process were the following.

The base set of input variables was the following

**X ={**

**'NNODES','NPOSSIBLE\_PAIRS', 'GES\_SUM\_DIST', 'GES\_SUM\_DIST\_STAND','GES\_IS\_CENTROID', 'GOLEM\_SUM\_DIST', 'GOLEM\_SUM\_DIST\_STAND','GOLEM\_IS\_CENTROID', 'LINGAM\_SUM\_DIST', 'LINGAM\_SUM\_DIST\_STAND', 'LINGAM\_IS\_CENTROID', 'Notears\_SUM\_DIST', 'Notears\_SUM\_DIST\_STAND', 'Notears\_IS\_CENTROID', 'PC\_SUM\_DIST', 'PC\_SUM\_DIST\_STAND', 'PC\_IS\_CENTROID', 'CENTROIDS\_COUNT', 'MAX\_SUM\_DIST', 'MAX\_SUM\_DIST\_STAND', 'CENTROID\_SUM\_DIST', 'CENTROID\_SUM\_DIST\_STAND', 'AVE\_SUM\_DIST\_of\_5', 'AVE\_SUM\_DIST\_STAND\_of\_5', 'STD\_SUM\_DIST\_of\_5', 'STD\_SUM\_DIST\_STAND\_of\_5', 'SKW\_SUM\_DIST\_of\_5'  
}**

**Input variables X   
Target variable CENTROID\_IS\_MIN\_  
Output variable CENTROID\_IS\_MIN\_PRED**

**RidgeClassifier(alpha=1.0, class\_weight=None, copy\_X=True, fit\_intercept=True,**

**max\_iter=None, positive=False, random\_state=7906, solver='auto',**

**tol=0.0001)**

**Input variables X  
Target variable GES\_IS\_MIN\_  
Output variable GES\_IS\_MIN\_PRED**

**XGBClassifier(base\_score=None, booster='gbtree', callbacks=None,**

**colsample\_bylevel=None, colsample\_bynode=None,**

**colsample\_bytree=None, device='cpu', early\_stopping\_rounds=None,**

**enable\_categorical=False, eval\_metric=None, feature\_types=None,**

**gamma=None, grow\_policy=None, importance\_type=None,**

**interaction\_constraints=None, learning\_rate=None, max\_bin=None,**

**max\_cat\_threshold=None, max\_cat\_to\_onehot=None,**

**max\_delta\_step=None, max\_depth=None, max\_leaves=None,**

**min\_child\_weight=None, missing=nan, monotone\_constraints=None,**

**multi\_strategy=None, n\_estimators=None, n\_jobs=-1,**

**num\_parallel\_tree=None, objective='binary:logistic', ...)**

**Input variables X  
Target variable GOLEM\_IS\_MIN\_  
Output variable GOLEM\_IS\_MIN\_PRED**

**ExtraTreesClassifier(bootstrap=False, ccp\_alpha=0.0, class\_weight=None,**

**criterion='gini', max\_depth=None, max\_features='sqrt',**

**max\_leaf\_nodes=None, max\_samples=None,**

**min\_impurity\_decrease=0.0, min\_samples\_leaf=1,**

**min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,**

**n\_estimators=100, n\_jobs=-1, oob\_score=False,**

**random\_state=4223, verbose=0, warm\_start=False)**

**Input variables X  
Target variable LINGAM\_IS\_MIN\_  
Output variable LINGAM\_IS\_MIN\_PRED**

**ExtraTreesClassifier(bootstrap=False, ccp\_alpha=0.0, class\_weight=None,**

**criterion='gini', max\_depth=None, max\_features='sqrt',**

**max\_leaf\_nodes=None, max\_samples=None,**

**min\_impurity\_decrease=0.0, min\_samples\_leaf=1,**

**min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,**

**n\_estimators=100, n\_jobs=-1, oob\_score=False,**

**random\_state=2990, verbose=0, warm\_start=False)**

**Input variables X  
Target variable Notears\_IS\_MIN\_  
Output variable Notears\_IS\_MIN\_PRED**

**ExtraTreesClassifier(bootstrap=False, ccp\_alpha=0.0, class\_weight=None,**

**criterion='gini', max\_depth=None, max\_features='sqrt',**

**max\_leaf\_nodes=None, max\_samples=None,**

**min\_impurity\_decrease=0.0, min\_samples\_leaf=1,**

**min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,**

**n\_estimators=100, n\_jobs=-1, oob\_score=False,**

**random\_state=3789, verbose=0, warm\_start=False)**

**Input variables X**

**Target variable PC\_IS\_MIN\_  
Output variable PC\_IS\_MIN\_PRED**

**DecisionTreeClassifier(ccp\_alpha=0.0, class\_weight=None, criterion='gini',**

**max\_depth=None, max\_features=None, max\_leaf\_nodes=None,**

**min\_impurity\_decrease=0.0, min\_samples\_leaf=1,**

**min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,**

**random\_state=7051, splitter='best')**

In the Table {XXX\_Detectors\_Performance} the reported test Precision and Recall are averaged over the positive and negative class.

\begin{table}\label{XXX\_Detectors\_Performance}

\begin{tabular}{lllllllll}

Target &Model & Accuracy & AUC & Recall & Precision & F1 & Kappa & MCC \\

CENTROID$\_$IS$\_$MIN$\_$ &Ridge Classifier & 0.8700 & 0.0000 & 0.8750 & 0.9300 & 0.8860 & 0.7296 & 0.7659 \\

GES\_IS\_MIN\_& Extreme Gradient Boosting & 0.9033 & 0.6800 & 0.4000 & 0.4000 & 0.4000 & nan & 0.4000\\

GOLEM\_IS\_MIN\_& Extra Trees Classifier & & 0.8400 & 0.8028 & 0.8667 & 0.8500 & 0.8514 & 0.6782 & 0.6914\\

LINGAM\_IS\_MIN\_ & Extra Trees Classifier & 0.8000 & 0.8000 & 0.8667 & 0.8183 & 0.8312 & 0.5879 & 0.6100 \\

Notears\_IS\_MIN\_ & Extra Trees Classifier & 0.7233 & 0.8125 & 0.8250 & 0.7717 & 0.7785 & 0.3815 & 0.4061 \\

PC\_IS\_MIN\_ & Decision Tree Classifier & 0.9833 & 0.7900 & 0.8000 & 0.7500 & 0.7667 & nan & 0.7632

\end{tabular}

\end{table}

The individual detectors have shown the following performances on the full dataset (number of instances 78).

TABLE {BR\_performance}

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Binary Relevance** | **CENTROID** | **GES** | **GOLEM** | **LINGAM** | **Notears** | **PC** |
| **TP** | 43 | 7 | 42 | 39 | 45 | 10 |
| **FP** | 5 | 0 | 4 | 4 | 2 | 0 |
| **FN** | 5 | 3 | 2 | 4 | 4 | 4 |
| **TN** | 25 | 68 | 30 | 31 | 27 | 66 |
| **Precision %** | 89.58 | 100.00 | 91.30 | 90.70 | 95.74 | 100.00 |
| **Recall %** | 89.58 | 70.00 | 95.45 | 90.70 | 91.84 | 83.33 |
| **Accuracy %** | 87.18 | 96.15 | 92.31 | 89.74 | 92.31 | 97.44 |

\begin{table}\label{BR\_detectors\_performance}

\begin{tabular}{|l|l|l|l|l|l|l|}

\hline Binary Relevance & CENTROID & GES & GOLEM & LINGAM & Notears & PC \\

\hline TP & 43 & 7 & 42 & 39 & 45 & 10 \\

\hline FP & 15 & 0 & 4 & 4 & 2 & 0 \\

\hline FN & 5 & 3 & 2 & 4 & 4 & 4 \\

\hline TN & 25 & 68 & 30 & 31 & 27 & 66 \\

\hline Precision $\%$ & 89.58 & 100.00 & 91.30 & 90.70 & 95.74 & 100.00 \\

\hline Recall $\%$ & 89.58 & 70.00 & 95.45 & 90.70 & 91.84 & 83.33 \\

\hline Accuracy $\%$ & 87.18 & 96.15 & 92.31 & 89.74 & 92.31 & 97.44 \\

\hline

\end{tabular}

\end{table}

We took the predictions of the binary classifier detectors and selected the [ALGO] as described above. The ranking in precision used by the aggregator function was PC, GES, Notears, GOLEM, Lingam, CENTROID. The outcomes displayed also in table {XXX\_Correct} and {XXX\_Distance} are the following.

While the lowest mean individual [ALGO]’s distance was CENTROID’s **10.08** the mean of the selected [ALGO]’s distance using BR was **6.62**, very close to the actual average minimum distance **5.38**: this result represents a significant improvement as shown by the standard errors and 95% confidence intervals reported in Table {XXX\_Distance}: comparing the selected algorithm’s and CENTROID’s distance through a one-tailed t-Student test we find that the former distance is lower than the latter with a p-value of 0.0487, hence below the conventional 0.05 confidence level threshold.

Expectedly, the selection of the [ALGO] according to the above method improves also the prediction of which [ALGO] perfectly corresponds to the GT. The results are summarized in the Table {XXX\_Distance}.

While the single most effective individual [ALGO] in correctly guessing the GT was CENTROID with **48.72%** correct predictions, the [ALGO] selected using BR is correct **61.54%** of the time, very close to the actual percentage of times when at least one algorithm in the group guesses correctly, which is **64.10%.**

**Outcomes of Classifiers Chain (CC) approach**

Using the incremental approach to multilabel classification provided by CC (ordering of targets in decreasing accuracy of their individual detectors: PC\_IS\_MIN\_, GES\_IS\_MIN\_, Notears\_IS\_MIN\_, GOLEM\_IS\_MIN\_, LINGAM\_IS\_MIN\_, CENTROID\_IS\_MIN\_), the configurations of the highest accuracy issued by the AutoML process turned out to be the following.

**Input variables X  
Target variable PC\_IS\_MIN\_  
Output variable PC\_IS\_MIN\_PRED**

**RidgeClassifier(alpha=1.0, class\_weight=None, copy\_X=True, fit\_intercept=True,**

**max\_iter=None, positive=False, random\_state=7467, solver='auto',**

**tol=0.0001)**

**Input variables {X, PC\_IS\_MIN\_PRED},   
Target variable GES\_IS\_MIN\_  
Output variable GES\_IS\_MIN\_PRED**

**XGBClassifier(base\_score=None, booster='gbtree', callbacks=None,**

**colsample\_bylevel=None, colsample\_bynode=None,**

**colsample\_bytree=None, device='cpu', early\_stopping\_rounds=None,**

**enable\_categorical=False, eval\_metric=None, feature\_types=None,**

**gamma=None, grow\_policy=None, importance\_type=None,**

**interaction\_constraints=None, learning\_rate=None, max\_bin=None,**

**max\_cat\_threshold=None, max\_cat\_to\_onehot=None,**

**max\_delta\_step=None, max\_depth=None, max\_leaves=None,**

**min\_child\_weight=None, missing=nan, monotone\_constraints=None,**

**multi\_strategy=None, n\_estimators=None, n\_jobs=-1,**

**num\_parallel\_tree=None, objective='binary:logistic', ...)**

**Input variables {X, PC\_IS\_MIN\_PRED, GES\_IS\_MIN\_ PRED }   
Target variable Notears\_IS\_MIN\_  
Output variable Notears\_IS\_MIN\_PRED**

**RandomForestClassifier(bootstrap=True, ccp\_alpha=0.0, class\_weight=None,**

**criterion='gini', max\_depth=None, max\_features='sqrt',**

**max\_leaf\_nodes=None, max\_samples=None,**

**min\_impurity\_decrease=0.0, min\_samples\_leaf=1,**

**min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,**

**n\_estimators=100, n\_jobs=-1, oob\_score=False,**

**random\_state=3346, verbose=0, warm\_start=False)**

**Input variables {X, PC\_IS\_MIN\_ PRED, GES\_IS\_MIN\_ PRED, Notears\_IS\_MIN\_ PRED }   
Target variable GOLEM\_IS\_MIN\_  
Output variable GOLEM\_IS\_MIN\_PRED**

**ExtraTreesClassifier(bootstrap=False, ccp\_alpha=0.0, class\_weight=None,**

**criterion='gini', max\_depth=None, max\_features='sqrt',**

**max\_leaf\_nodes=None, max\_samples=None,**

**min\_impurity\_decrease=0.0, min\_samples\_leaf=1,**

**min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,**

**n\_estimators=100, n\_jobs=-1, oob\_score=False,**

**random\_state=1692, verbose=0, warm\_start=False)**

**Input variables {X, PC\_IS\_MIN\_ PRED, GES\_IS\_MIN\_ PRED, Notears\_IS\_MIN\_ PRED, GOLEM\_IS\_MIN\_ PRED }   
Target variable LINGAM\_IS\_MIN\_  
Output variable LINGAM\_IS\_MIN\_PRED**

**ExtraTreesClassifier(bootstrap=False, ccp\_alpha=0.0, class\_weight=None,**

**criterion='gini', max\_depth=None, max\_features='sqrt',**

**max\_leaf\_nodes=None, max\_samples=None,**

**min\_impurity\_decrease=0.0, min\_samples\_leaf=1,**

**min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,**

**n\_estimators=100, n\_jobs=-1, oob\_score=False,**

**random\_state=8154, verbose=0, warm\_start=False)**

**Input variables {X, PC\_IS\_MIN\_ PRED, GES\_IS\_MIN\_ PRED, Notears\_IS\_MIN\_ PRED, GOLEM\_IS\_MIN\_ PRED, LINGAM\_IS\_MIN\_ PRED}   
Target variable CENTROID\_IS\_MIN\_**

**Output variable CENTROID\_IS\_MIN\_PRED**

**RidgeClassifier(alpha=1.0, class\_weight=None, copy\_X=True, fit\_intercept=True,**

**max\_iter=None, positive=False, random\_state=7184, solver='auto',**

**tol=0.0001)**

\begin{table}\label{CC\_detectors\_performance}

\begin{tabular}{|l|l|r|r|r|r|r|r|r|}

Target & Model & Accuracy & AUC & Recall & Prec. & F1 & Kappa & MCC \\

PC\_IS\_MIN\_ ridge & Ridge Classifier & 0.9800 & 0.0000 & 0.8000 & 0.8000 & 0.8000 & nan & 0.8000 \\

GES\_IS\_MIN\_ & Extreme Gradient Boosting & 0.9100 & 0.6175 & 0.3000 & 0.2500 & 0.2667 & nan & 0.2632 \\

Notears\_IS\_MIN\_ & Random Forest Classifier & 0.6767 & 0.7208 & 0.7833 & 0.7433 & 0.7373 & 0.2959 & 0.3248 \\

GOLEM\_IS\_MIN\_ & Extra Trees Classifier & 0.8067 & 0.8333 & 0.9000 & 0.8033 & 0.8348 & 0.6040 & 0.6453 \\

LINGAM\_IS\_MIN\_ & Extra Trees Classifier & 0.7567 & 0.8389 & 0.8000 & 0.8100 & 0.7764 & 0.5113 & 0.5508 \\

CENTROID\_IS\_MIN\_ & Ridge Classifier & 0.8867 & 0.0000 & 0.8833 & 0.9500 & 0.8981 & 0.7722 & 0.8058

\end{tabular}

\end{table}

The individual detectors have shown the following performances on the full dataset (number of instances 78).

TABLE {CC\_performance}

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Binary Relevance | CENTROID | GES | GOLEM | LINGAM | Notears | PC |
| TP | 47 | 8 | 44 | 41 | 46 | 12 |
| FP | 6 | 0 | 3 | 5 | 3 | 1 |
| FN | 1 | 2 | 0 | 2 | 3 | 0 |
| TN | 24 | 68 | 31 | 30 | 26 | 65 |
| Precision |  |  |  |  |  |  |
| Recall |  |  |  |  |  |  |
| Accuracy |  |  |  |  |  |  |
| precision rank | 6 | 1 | 3 | 5 | 2 | 4 |

\begin{table}

\begin{tabular}{|l|r|r|r|r|r|r|}

\hline Binary Relevance & CENTROID & GES & GOLEM & LINGAM & Notears & PC \\

\hline TP & 47 & 8 & 44 & 41 & 46 & 12 \\

\hline FP & 6 & 0 & 3 & 5 & 3 & 1 \\

\hline FN & 1 & 2 & 0 & 2 & 3 & 0 \\

\hline TN & 24 & 68 & 31 & 30 & 26 & 65 \\

\hline Precision & $88.68 \%$ & $100.00 \%$ & $93.62 \%$ & $89.13 \%$ & $93.88 \%$ & $92.31 \%$ \\

\hline Recall & $97.92 \%$ & $80.00 \%$ & $100.00 \%$ & $95.35 \%$ & $93.88 \%$ & $100.00 \%$ \\

\hline Accuracy & $91.03 \%$ & $97.44 \%$ & $96.15 \%$ & $91.03 \%$ & $92.31 \%$ & $98.72 \%$ \\

\hline precision rank & 6 & 1 & 3 & 5 & 2 & 4 \\

\hline

\end{tabular}

\end{table}

We took the predictions of the incrementally trained binary classifiers and selected the [ALGO] as described above. The precision based ranking used by the selector function was GES, Notears, GOLEM, PC, Lingam, CENTROID. The outcomes displayed also in table {XXX\_Correct} and {XXX\_Distance} are the following.

We recall that the lowest mean individual [ALGO]’s distance was CENTROID’s **10.08**. The mean of the selected [ALGO]’s distance using CC was **6.36**, very close to the actual average minimum distance **5.38**: this result represents a significant improvement as shown by the standard errors and 95% confidence intervals reported in Table {XXX\_Distance}: comparing the selected algorithm’s and CENTROID’s distance through a one-tailed t-Student test we find that the former distance is lower than the latter with a p-value of 0.0387, hence below the conventional 0.05 confidence level threshold. The result from CC was slightly better than the result by BR (6.62) but not significantly different from it.

As with BR, the selection of the [ALGO] according to CC improves also the prediction of which [ALGO] perfectly corresponds to the GT. The results are summarized in the Table {XXX\_Distance}.

While the single most effective individual [ALGO] in correctly guessing the GT was CENTROID with **48.72%** correct predictions, the [ALGO] selected using BR is correct **62.82%** of the time, very close to the actual percentage of times when at least one algorithm in the group guesses correctly, which is **64.10%.**Also in this respect the result from CC was slightly better than the result by BR (61.54) but not significantly different from it.