Structure Learning Bayesian Networks from Heart Failure Data

Radboud University - Bayesian Networks and Causal Inference - NWI-IMC012-2021-KW1-V
Group 23 - Assignment 2

Evander van Wolfswinkel
s1057895

Janneke Verbeek
s1011065

Niek Derksen
s4363779

ABSTRACT

This article will compare two types of Bayesian network structure learning algorithms applied to Chronic Heart Failure (CHF) data: the Hill Climbing (HC) algorithm and the Parents and Children (PC) algorithm. These algorithms are implemented in `bnlearn`*, an R software package implementing various structure learning algorithms. The HC algorithm will make use of two model selection criteria; AIC and BIC and compare results based on these parametric changes, compare with the HC algorithm and benchmark with a manually constructed Bayesian Network (BN) from the same data. Using these structural comparisons; the AIC HC algorithm to provide a better approximate causal structure than the BIC HC. When comparing the AIC/BIC HC to the PC, no notable difference can be discerned between the two algorithms. However, some edges seem to be reversed between respectively HC and PC approaches. The AIC HC is more similar, in particular edge wise, to the benchmark manual BN, than the PC generated structure is when compared to the benchmark BN. However, edges for the AIC HC BN comparison are once again reversed.*

1 INTRODUCTION

This report functions as an extension on earlier work, in which causal inference was conducted on a Heart Failure data based on the insights gained from manually constructing a Bayesian Network. This report will extend on this work, but will use a structure learning approach.

Two structure learning algorithms are compared: the hill climbing algorithm and the PC algorithm. The inner workings and results of these structure learning algorithms are analyzed, and an attempt will be made to compare the models to each other. The before mentioned manually constructed Bayesian Network will serve as a structural benchmark for the structure learning approach on the same data set.

For the hill-climbing algorithm, the effects of changing the scoring criterion, the amount of random restarts and the maximum amount of iterations are examined. In the PC algorithm, the effects of changing the target nominal type I error rate (alpha) as well as the maximum allowed size for conditioning sets are explored; concluding with an overall result summary. Finally, differences, similarities and enhancements to each structure learning algorithm are discussed based on findings from the result section.

1.1 Structure Learning Algorithms

1.1.1 Hill Climbing. The Hill Climbing (HC) algorithm is a local search algorithm. The algorithm was first described by Lin and Kernighan [4], as a way of finding solutions to the traveling salesperson problem. The general (informal) formulation of the types of problems that can be solved with this algorithm is the following: *find from a set S a subset $T*$ that satisfies some criterion C and minimizes an objective function f*. In the case of Bayesian networks, this means that from the set of all possible graph configurations S, we have to find some graph $T*$ that is a DAG (C), and which minimizes some objective scoring function f. f in particular can vary, but in the implementation used in this report the Bayesian Information Criterion (BIC) and Aikake Information Criterion (AIC) are taken. The BIC is preferred due to its wide spread accepted use, penalization of model complexity and coherent and consistent results. As well as the fact that this project has a large enough sample size to parameter ratio in order for the BIC not to suffer from its validity issues, as described by Giraud [3]. Whereas the AIC is better suited for the healtcare natured scenario, in which inherently, false-negative significant relationships and risk want to be avoided at all costs. The AIC trades off with the more false-positive and risk oriented BIC in this regard. However, the AIC is less consistent and unpredictable in its results due to its infinite dimension [8].

In order to find $T*$, we follow the following steps:

(1) Generate a random initial solution T
(2) (a) For each step:
  (b) Apply one operation of either {*add_edge*, *delete_edge*, *reverse_edge*} to T, such that the improvement on f is maximal and T is a DAG
  (c) If no better improvement on f can be found according to some stopping criterion, go to 3, else, return to 2b.
(3) If a best improvement is found in step 2, return to step 2 with new T, if no improvement can be found, continue to 4.
(4) Restart if desired.

As HC is a local search algorithm, this will not result in a globally optimal solution, but rather a locally optimal solution. This is also why step 4 exists: with a random restart, it may be possible to find a solution that is more optimal than the local optimum found on the previous run of the algorithm. For HC, `bnlearn` lists the following parameters:

- `whitelist`: a set of arcs to be included in the graph
- `blacklist`: a set of arcs to be excluded from the graph
- `score`: label of the network score that is to be used in the algorithm. For both discrete and continuous data sets, this is the Bayesian Information Criterion, but this can also be set to other types of scores.
- `debug`: boolean indicating whether debugging output is printed
- `restart`: integer indicating the number of random restarts.
- `perturb`: integer indicating the number of attempts to randomly insert/remove/reverse an arc on the random restarts.
- `max.iter`: maximum number of iterations

- `max.p`: maximum number of parents for a node, infinite by default.
- `optimized`: whether to use score caching to speed up the structure learning.

1.1.2 Peter Spirtes Clark Glymour (PC) Algorithm. The PC algorithm appears to first have been described by Spirtes and Glymour, [5]. The steps of the algorithm are the following:

(1) Start with fully connected graph T
(2) For each edge (X, Y) in T, try to find a set Z_{XY} such that $X \perp\!\!\!\perp Y | Z_{XY}$. If Z_{XY} exists, remove the edge (X, Y).
(3) For each pair of variables (X, Y) that is not linked but that do have a common neighbor W, if $W \in Z_{XY}$, make the edges (X, W) and (Y, W) directed.
(4) Reorient the graph into a CPDAG.

In `bnlearn` all constraint-based structure algorithms, including the PC algorithm, share the same parameters. These are the following:

- `cluster`: optional cluster object
- `whitelist`: a set of arcs to be included in the graph
- `blacklist`: a set of arcs to be excluded from the graph
- `test`: the type of conditional independence test to be used in the algorithm. Since our data is discrete, this is mutual information.
- `alpha`: target nominal type I error rate
- `B`: number of permutations considered for each permutation test
- `max.sx`: maximum allowed size of the conditioning sets used in conditional independence tests.
- `debug`: boolean indicating whether debugging output is printed
- `undirected`: boolean whether the algorithm will attempt to infer the direction of the arcs.

Obviously, some of these parameters are either very optional (`whitelist` and `blacklist`) or do not directly influence the results of the algorithm (`debug`)

2 METHODS

2.1 Data and Preprocessing

The data used in this project was collected at the Faisalabad Institute of Cardiology and the Allied Hospital in Faisalabad (Punjab, Pakistan) and contains medical records of 299 heart failure patients [1]. The data consists of 14 variables, where the variable of interest is $DEATH\_EVENT$, indicating whether a subject has died from CHF.

Since the data has both discrete (binary) and continuous variables, all continuous variables were binned. Binning was done by examining the histograms of the data and dividing the data into regularly spaced bins. Outliers were grouped in the last bin for each corresponding variable. Details for data prepossessing can be found in this projects preceding works.

2.2 Implementation Details

The PC and HC algorithms discussed within this report are implementations provided by the `bnlearn` package. Follow this link to the code base of this project for the exact implementation.

In order to compare the two graph structures, we use the `compare` function from `bnlearn`. The `compare` function takes a source and target structure and computes three quantities: true positives, indicating how many arcs are both in source and in target, false positives, indicating how many arcs are in target but not in source, and false negatives, indicating how many arcs are in source but not in target. Furthermore, we compute the Hamming distance between the (adjacency matrices of) two graphs. The Hamming distance is nothing more than a distance measure between two binary strings, which in this case are the adjacency matrices; it is a quantification of how many string positions are different between the two strings. This is distance is implemented as a `bnlearn` function. The count of true positives (similar arc directions), false positives (similar arc, different direction) and false negatives (no similar arc, direction irrelevant) are also computed to show the comparison of directed and undirected edges between structures. For structure change comparison, the average amount of children nodes over all parent nodes was also computed.

3 RESULTS

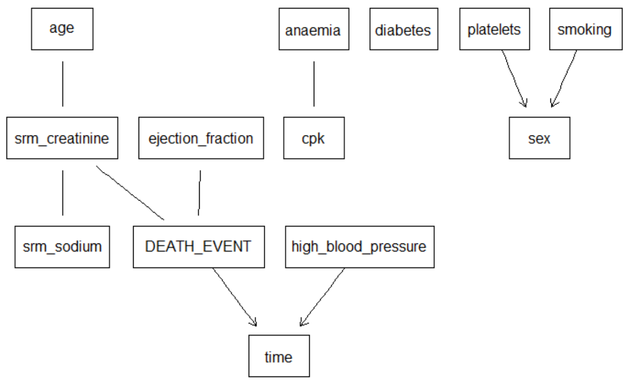
Several graphical models with various parameter settings were generated using the PC and HC structure learning algorithms. In this section, we describe the results of parametric changes to these algorithms, and compare the structures of the graphs resulting from these two algorithms. It is important to note that the structures resulting from the PC algorithm are CPDAGs, meaning that the structure describes an equivalence class of DAGs. The HC algorithm finds a DAG.

3.1 Effect of parametric changes

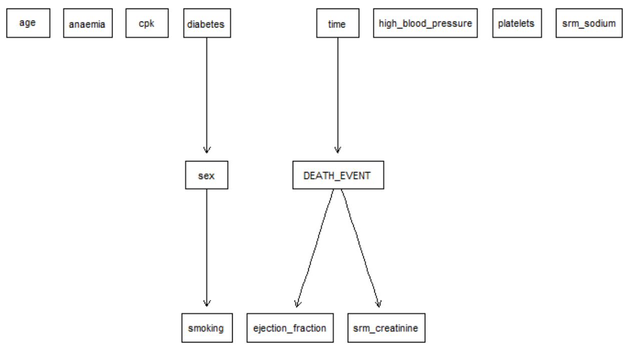
Before parametric changes were added, we used default values for both the PC and HC model as a starting point for exploration. The starting networks can be viewed in Figure 1.

For the PC algorithm, alpha and the maximum allowed size for conditioning sets in independence tests were changed. Lowering the alpha to 0.01 from 0.05 ensured that only the most significant independencies were considered. The resulting network contains more directed edges than its predecessor and some relationships are no longer visible, as can be seen on Figure 2a. Changing the maximum allowed size for conditioning sets in the independence tests to 1 results in a network with more direct relationships, as only 2 variables are tested at a time.

For the HC algorithm, the influence of changing the number of randomly inserted/removed or reserved arcs on every random restart was found to have no influence on the original graphical structure, when the BIC is used as scoring metric. Changing the maximum amount of iterations to a value below 5 will result in a network with fewer edges as each iteration is necessary to derive one edge. Changing the amount of restarts for perturbations also seems of no effect, depending on the network scoring metrics used. When the Akaike Information Criterion (AIC) instead of the Bayesian Information Criterion (BIC) is used for scoring, the resulting network shows a denser structure than that of the network generated by the PC algorithm, with an arc count of 13 (compared to 8 of the changed PC graph in Figure 2a). This network can be viewed in Figure 2b.

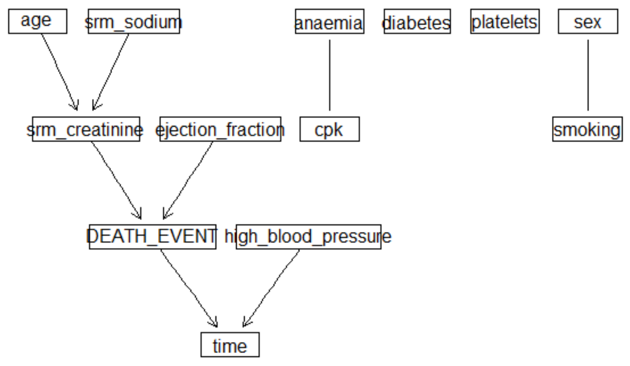


(a) PC algorithm

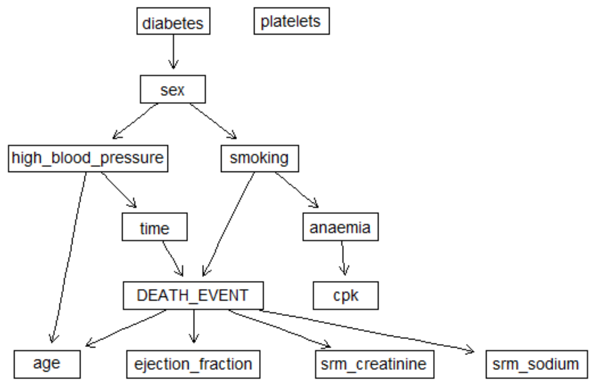


(b) HC algorithm

Figure 1: Results of structure learning for PC and HC algorithms



(a) PC algorithm (max conditioning set size = 1, alpha = 0.01)



(b) HC algorithm (scoring metric: Akaike Information Criterion)

Figure 2: Results of parametric changes of structure learning for PC and HC algorithms

3.2 Comparison structure learning algorithms

To compare structure learning algorithms on the algorithms themselves, default settings of both PC and HC algorithms were used as starting points of comparison, the graphs of which are plotted in Figure 1.

In Figure 3, a visual comparison is made between the default PC and HC algorithms. From the figure, where PC is considered the true network, it is observed that many arcs found in the PC graph are not found in the results of HC structure learning (represented by the dashed blue lines). However, some arcs with differently directed edges are found between the same nodes, represented by the red lines. These differences make the networks structure seem much less similar than they really are, as the relationships that are found are only false positive due to being undirected or in the other direction comparatively.

The networks Hamming distance from each other is 6, meaning the number of arcs that are different between the two. Ignoring missing arcs and directions, it can be seen that of all 5 arcs of the HC structure are also found in the PC network structure. When looking at the direction of these 5 arcs, 0 are in the same direction as the other graph.

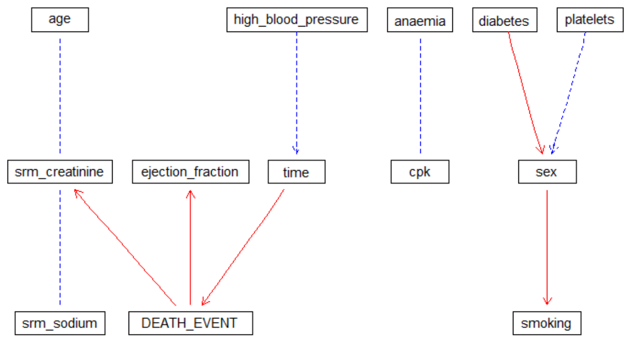


Figure 3: Comparison between results of structure learning (Figure 1) for PC and HC algorithms. For this visualisation (where PC is considered the true network) true positive arcs are in black, false positive arcs (missing or different directions in PC network) are in red and the dashed blue line represents false negatives.

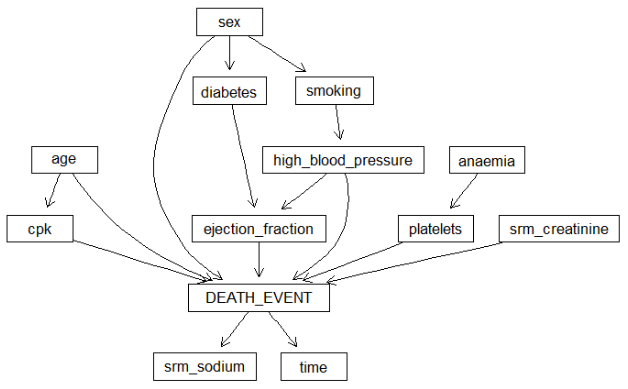


Figure 4: Manually constructed network using theoretical knowledge and tested through Chi-squared independence tests

When parameters are changed for the PC algorithms to include more significant relationships and only size 1 conditioning size in-dependency tests sets are considered, as seen in Figure 2 with the HC algorithm using the AIC scoring metric, the structures can be compared again. The Hamming distance has now increased to 8, meaning more different arcs between networks. Ignoring any missing arcs and directions, 6 arcs of the HC structure are also found in the PC structure, with 1 of these 6 having the same direction. Comparing the average amount of children over all parent nodes reveals an average of 2.16 children for the HC structure and an average of 1 for the PC algorithm. This is due to the small conditioning size set, but the previous default network of Figure 1a. shows similar behaviour over the directed edges with an average of 1 child per parent node.

3.3 Comparison with manually constructed network

A manually constructed network was devised using theoretical knowledge from literature and tested through chi-squared independence tests, visible in Figure 4. We used this network's structure, which has been tested through iterations of tests and uses domain knowledge over both the dataset and known medical relationships, as a somewhat ground truth point of comparison. When this structure seen in Figure 4 is compared to the HC algorithm default structure of Figure 1b, we find a Hamming distance of 11 compared to 17 for the default PC structure of Figure 1a. This suggests that the default HC algorithm is much closer in similarity to our manually constructed network. When direction of arcs and missing arcs are ignored, all 5 of the HC algorithm arcs are also found in the manually constructed network, with only 1 arc having a similar direction. For the default PC network structure, 4 arcs are also found in the manually constructed network, with none having a similar direction.

When algorithms with parametric changes (Figure 2) are compared to the manually constructed network, Hamming distance increases to 15 for the HC network and decreases to 16 for the PC network. Between HC and the manual network, 2 similarly directed relationship arcs are found and 11 arcs with different directions for the same relationship. For the PC structure, 3 similarly directed relationships are found and only 5 arcs with different directions for the same relationship. This implies that for the parametrically changed algorithms, HC shows much more similar relationships than the PC algorithm.

4 DISCUSSION

Determining whether the HC or PC algorithm resulted in a better performance is in this case quite hard, as there is no way to compare the structures generated by these algorithm to the 'true' Bayesian network, and no quantitative metric by which we can compare whether one structure is better than another. Therefore, we can only speculate as to which structure may be better using our domain knowledge as a guide. We are inclined to trust the structure generated by the PC algorithm (using the default settings) more, since the structure resulting from this algorithm contained more plausible dependencies than the structure resulting from the HC algorithm, and also found some of the dependencies resulting from our manually constructed network.

Between structure learning algorithms and their parametrically changed counterparts, HC is found to generate networks with differently directed arcs compared to the PC algorithm. When the scoring metric for the HC algorithm is changed to use AIC rather than BIC, a much more dense network is generated. The complexity penalization of BIC, resulting in the very sparse network of figure 1b, may not be preferred in our model, especially when compared to our manually constructed network, which is more similarly dense to the AIC structure. For the PC algorithm, the default parameters returns a decent model but suffers from many undirected arcs, making inference hard. Including only very low alpha independencies results in more directed edges, making comparison easier. Limiting the size of the conditioning set results in more robust tests, but also limits the dimensionality, suggesting that the max conditioning set size should be increased.

When these structure learning algorithms are compared to a manually constructed network that uses domain knowledge from literature and has been devised through iterations of independency tests, we can use this structure as a semi ground truth. Results indicate that the AIC HC algorithm computed the most similar relationships but with many arc directions inverted. While more unpredictable than the BIC HC, the increased dimensionality and capturing of false-negative significant relationships help the AIC HC algorithm perform better in our health-care focused network [8].

Since HC searches the space of all possible network configurations, HC is exponential in the number of variables, thus any approach that somewhat reduces the complexity of the algorithm can be considered an improvement - and there are many. One such improvement is for instance restricting the search space to only equivalence classes of DAGs (CPDAGs), which is the approach taken by Greedy Bayesian Pattern Search, [6], but a plethora of other variants are available. HC can also be used as a local search method *within other structure learning algorithms*, which is the case in max-min hill-climbing search (MMHC), proposed by Tsamardinos et al. [7]. MMHC first constructs a skeleton of a Bayesian network, and then uses HC to orient the edges. As the PC algorithm finds CP-DAGs, it could be interesting to examine the DAGs corresponding to the CPDAG found by the PC algorithm. Colombo and Maathuis suggest that stability selection can be used to choose α in a somewhat principled manner, which should increase the amount of true positives found by the structures resulting from the PC algorithm, [2].

REFERENCES

[1] Ahmad, T., Munir, A., Bhatti, S. H., Aftab, M., and Raza, M. A. Survival analysis of heart failure patients: A case study. *PloS one 12*, 7 (2017), e0181001.

[2] Colombo, D., and Maathuis, M. H. Order-independent constraint-based causal structure learning. *J. Mach. Learn. Res. 15*, 1 (jan 2014), 3741–3782.

[3] Giraud, C. *Introduction to high-dimensional statistics.* CRC Press, Taylor & Francis Group, 2015.

[4] Lin, S., and Kernighan, B. W. An effective heuristic algorithm for the traveling-salesman problem. *Operations research 21*, 2 (1973), 498–516.

[5] Spirtes, P., and Glymour, C. An algorithm for fast recovery of sparse causal graphs. *Social science computer review 9*, 1 (1991), 62–72.

[6] Spirtes, P., and Meek, C. Learning bayesian networks with discrete variables from data. In *KDD* (1995), vol. 1, pp. 294–299.

[7] Tsamardinos, I., Brown, L. E., and Aliferis, C. F. The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning 65*, 1 (2006), 31–78.

[8] Wang, Y., and Liu, Q. Comparison of akaike information criterion (aic) and bayesian information criterion (bic) in selection of stock–recruitment relationships. *Fisheries Research 77*, 2 (2006), 220–225.