Learning bayesian networks with pyAgrum

Joachim Verschelde

July 2024

1 Introduction

The study of Bayesian networks has become increasingly important in various fields due to their ability to model complex probabilistic relationships among variables. This assignment aims to delve into the details of Bayesian network structure learning through two distinct studies. The first study explores the impact of sample size on the structure of the learned network by comparing results obtained from a search-and-score algorithm and a constraint-based learning algorithm. The second study focuses on learning a classifier for breast cancer diagnosis, where we compare a network constructed from expert knowledge with one learned purely from data.

2 Methods

2.1 Investigating the effect of sample size on learning structure

We will start with the bayesian network from the first assignment, which by coincidence was also a breast cancer classifier, although with a more simplified network topology.

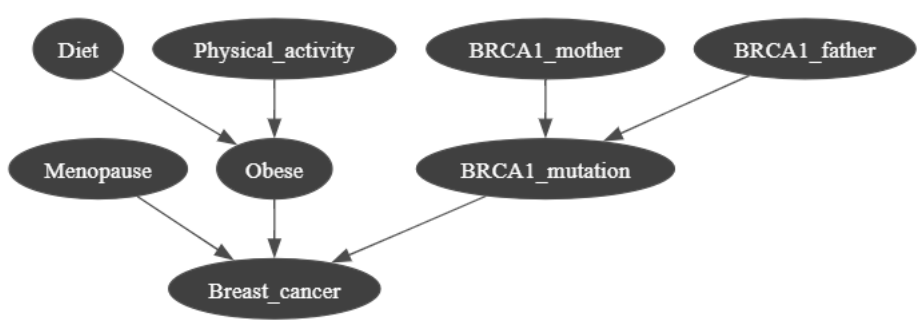


Figure 1: The topology of the bayesian network from assignment 1

This network will be used to generate samples from various sizes, ranging from 100 to 10000 samples. The samples will be generated using the *BNDatabaseGenerator* class from the pyAgrum library. Next we will perform experiments to see how well the structure of the original network can be learned from the generated samples. For each experiment we will vary 2 parameters: the sample size and the learning algorithm. The learning algorithms we will use are the Mutual Information-based Incremental Construction (MIIC) algorithm as a constraint-based method, and the greedy hill climbing algorithm (GHC) as a score-based method. For each experiment, the resulting networks will be compared to the original network to determine how well the topology of the original network is learned. This will be a visual comparison and for this the *getBNDiff* method from the *bn_vs_bn* pyAgrum module will be used. *getBNDiff* visualizes the differences between two Bayesian networks by highlighting the differences in the structure of the networks.

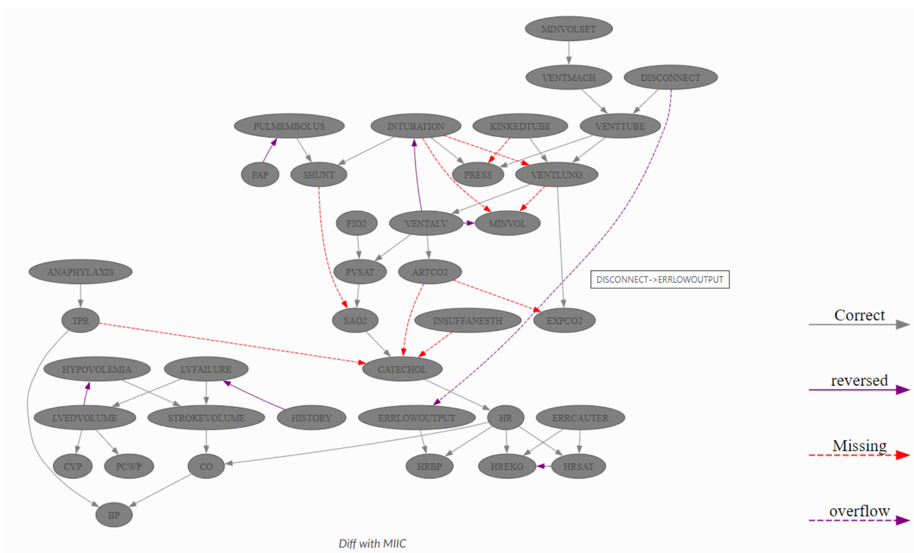


Figure 2: Example of the getBNDiff visualizations

2.2 Learning a classifier

In the second part we will learn a classifier for breast cancer using Bayesian networks.

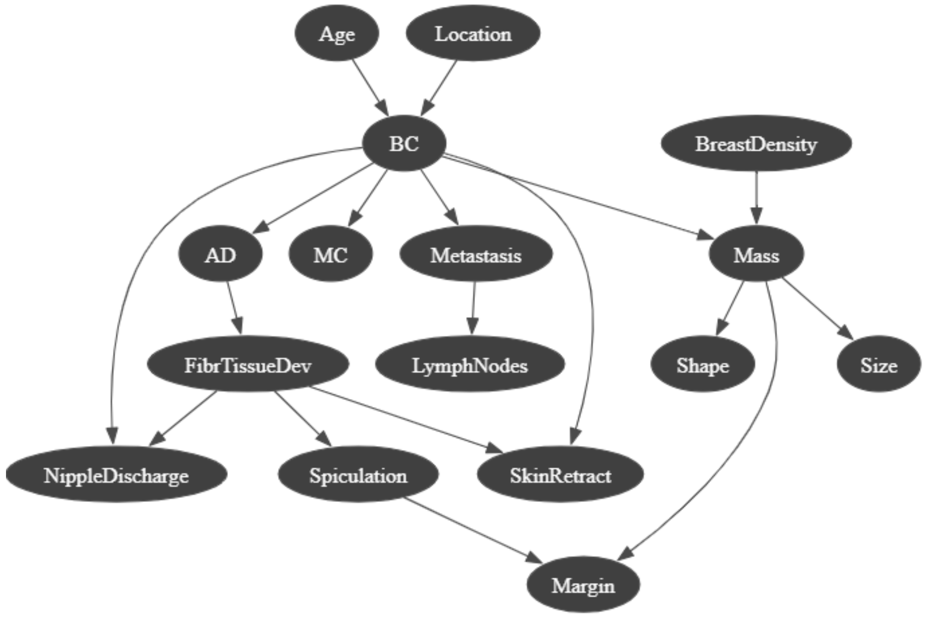


Figure 3: The topology of the breast cancer network

We want to compare the difference in performance between a network learned from data and a network constructed based on expert knowledge. Given is a breast cancer network based on expert knowledge and it will serve as a benchmark to compare our learned network with. Again using the GHC algorithm, a network structure will be derived from the available breast cancer dataset. To test the generalization properties of the learned network, we first need to split the dataset into a training and a test set, this to avoid overfitting on the training data. To quantify the similarity between the learned network and the expert-constructed network, we will use the structural Hamming distance. This distance metric measures the number of edges that differ between two networks. Additionally, the classification performance of both networks will be evaluated in terms of the area under the Receiver Operating Characteristic (ROC) curve, treating the classification problem as binary (no breast cancer vs. breast cancer). Finally, we will explore whether incorporating structural constraints can steer the learning algorithm closer to the benchmark network and how this influences classification performance. This will be done by adding mandatory arcs to the learning algorithm, which will force the algorithm to include these arcs in the learned network.

3 Results

The code for the experiments can be found on the following GitHub repository: `https://github.com/JoachimVerscheldePersonal/BRAL_Assignment2`.

3.1 Investigating the effect of sample size on learning structure

The visualizations of the differences between the learned networks and the original network can be found in appendix A. After 100 samples both the MIIC and the GHC algorithms have a hard time learning the original network structure, not even one arc is correct in the network structure generated by the GHC algorithm. We see a major improvement after 500 samples, where the GHC algorithm already has 5 out of 7 arcs in the correct position, and 4 out of 7 arcs in the correct causal direction. The MIIC algorithm also has 5 out of 7 arcs in the correct position, but only 3 out of 7 arcs in the correct causal direction. Although there are less arcs in the correct causal direction, do notice that the MIIC algorithm does not have any faulty arcs compared to the GHC algorithm. Next using 1000 samples the GHC algorithm has found all of the correct arcs, but has also introduced 2 faulty arcs. Additionally, there are only 2 arcs in the correct causal direction, which is worse compared to the 500 sample experiment. Using 1000 samples the MIIC now has the upperhand as all of the arcs are in the correct position, there are no faulty arcs and 5 out of 7 arcs are in the correct causal direction. It takes 10000 samples for the GHC algorithm to have all 7 arcs in the correct position, but even then there is still a faulty arc and almost half of the arcs is still pointing in the wrong direction. There is little improvement when using more than 1000 samples for the MIIC algorithm, when we reach 5000 samples, only one extra arc is in the correct causal direction, and 10000 samples does not improve this further.

3.2 Learning a classifier

We split the dataset into a training and a test set, with a 80/20 ratio. The GHC algorithm was used to learn the network structure from 16000 training samples. The Hamming distance between the learned network and the benchmark network was two, which means that two edges differed between the two networks. If we visualize these differences we can indeed see that there two edges too many in the learned network, being the edge between *BC* and *FibrTussueDev* and the edge between *Mass* and *Spiculation*.

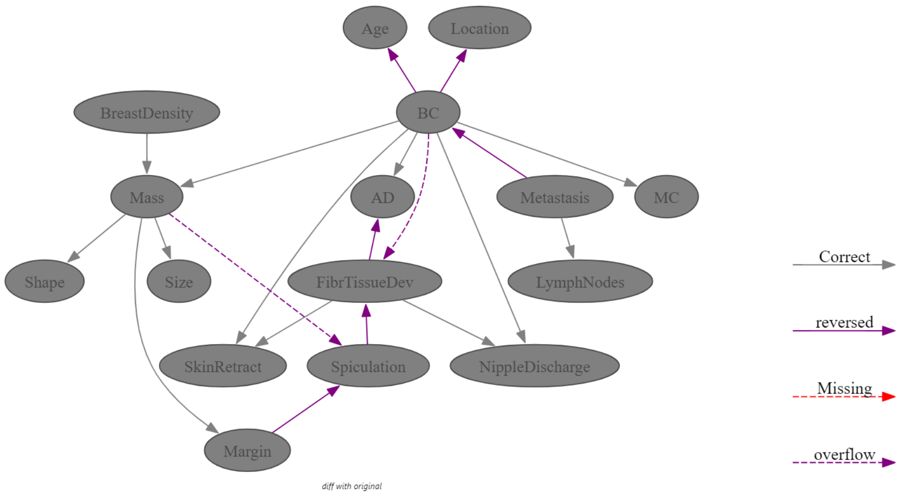
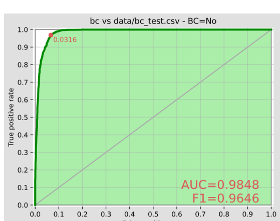
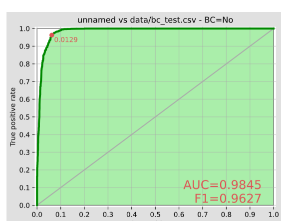


Figure 4: The difference between the learned network and the benchmark network

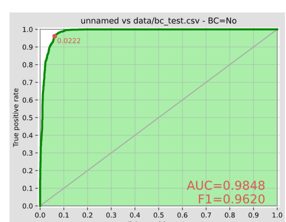
The performance difference between the learned network and the benchmark network on the test set was minimal, with an AUC of 0.9848 for the benchmark network and 0.9845 for the learned network. This could indicates that the learned network generalizes well to unseen data. We could try to improve the learned model by introducing structural constraints. We can derive from the learned model visualizations that we could correct the arc from *Location* to *BC*. If we then look at figure 5c we can see that the area under the curve is has improved to the same as the benchmark network.



(a) Benchmark network



(b) Learned network



(c) Learned network with structural constraints

Figure 5: ROC curves for performance comparison

4 Conclusions and discussion

4.1 Investigating the effect of sample size on learning structure

The GHC algorithm performed better than the MIIC algorithm when using small sample sizes < 1000. This could be due to the conservative nature of the MIIC algorithm, to only link nodes based on statistical tests. This is also reflected in the fact that the MIIC algorithm did not introduce any faulty arcs, while the GHC algorithm did. The GHC algorithm is not dependant on these statistical tests, and only tries to maximize a scoring function. This is more of a greedy approach and could explain why the GHC algorithm performed better with small sample sizes. Using sample sizes of 1000 and up, the MIIC algorithm had the best performance by far.

4.2 Learning a classifier

We divided the dataset into training and test sets in an 80/20 ratio, using 16,000 samples for training. The Greedy Hill Climbing (GHC) algorithm was employed to learn the network structure. The Hamming distance between the learned network and the benchmark network was two, indicating two differing edges. Visual inspection confirmed these differences, specifically the edges between *BC* and *FibrTussueDev*, and between *Mass* and *Spiculation*. Despite these structural differences, the performance of the learned network on the test set was nearly identical to the benchmark network, with AUC values of 0.9845 and 0.9848. This suggests that the learned network generalizes well to unseen data. Introducing structural constraints, such as correcting the arc from *Location* to *BC*, improved the model's performance to match that of the benchmark network.

A Appendix

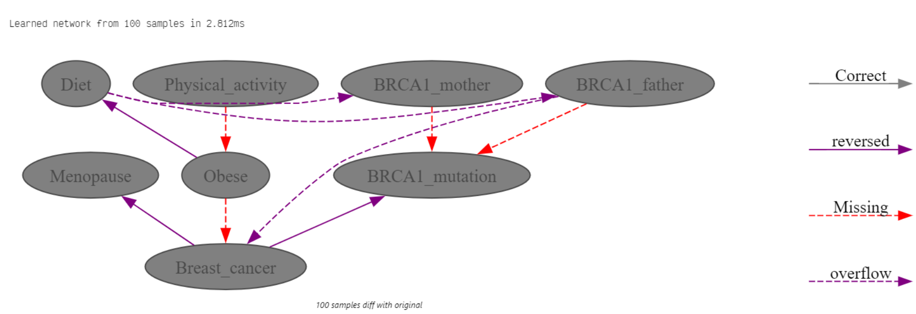


Figure 6: Greedy Hill Climbing algorithm using 100 samples

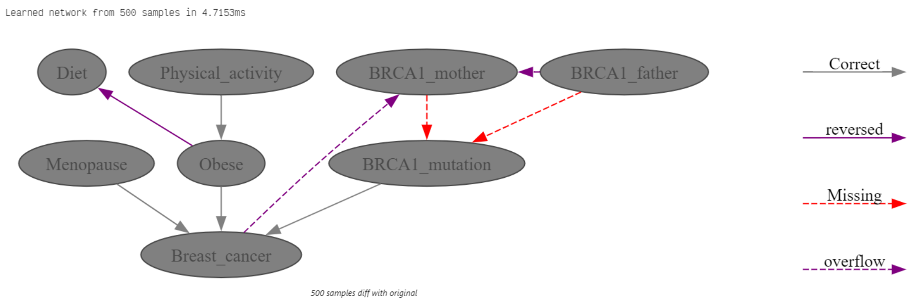


Figure 7: Greedy Hill Climbing algorithm using 500 samples

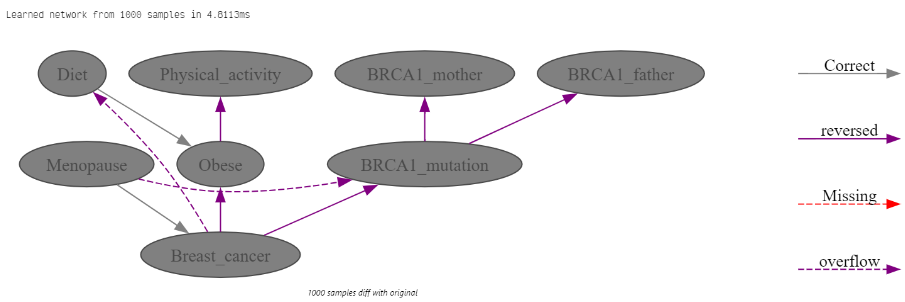


Figure 8: Greedy Hill Climbing algorithm using 1000 samples

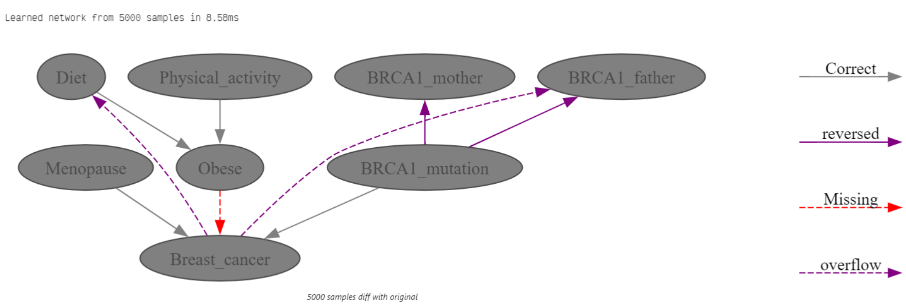


Figure 9: Greedy Hill Climbing algorithm using 5000 samples

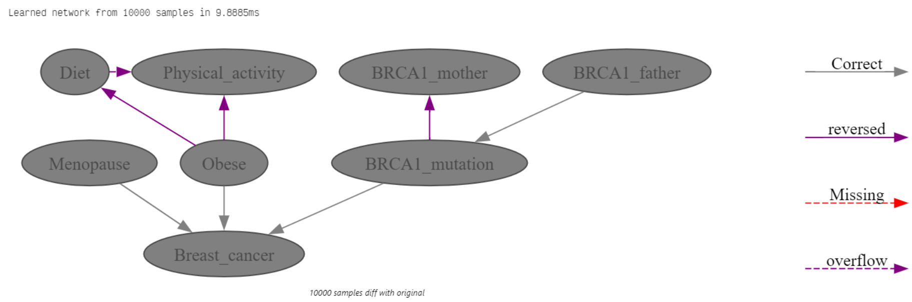


Figure 10: Greedy Hill Climbing algorithm using 10000 samples

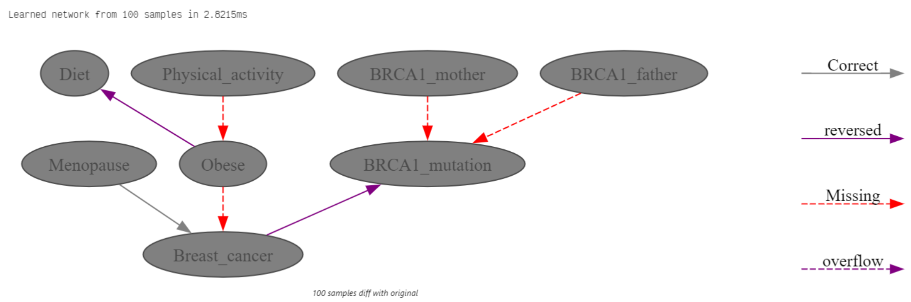


Figure 11: MIIC algorithm using 100 samples

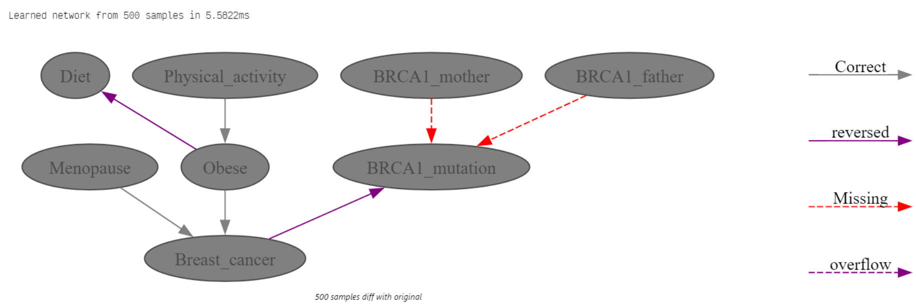


Figure 12: MIIC algorithm using 500 samples

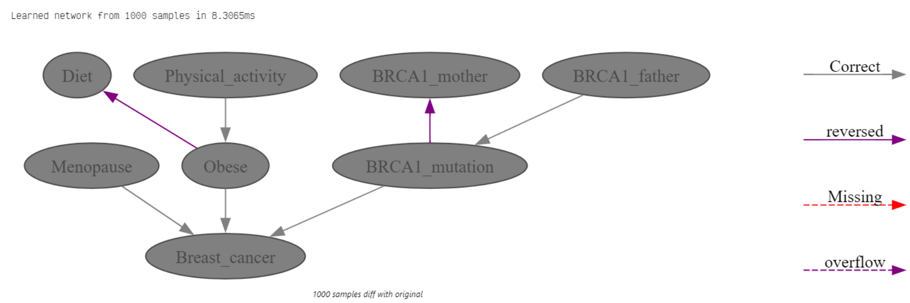


Figure 13: MIIC algorithm using 1000 samples

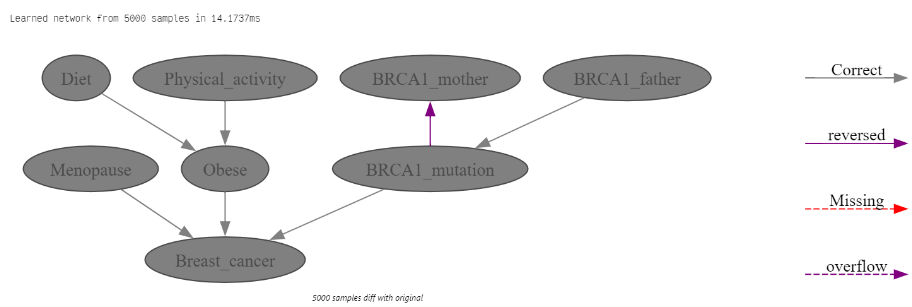


Figure 14: MIIC algorithm using 5000 samples

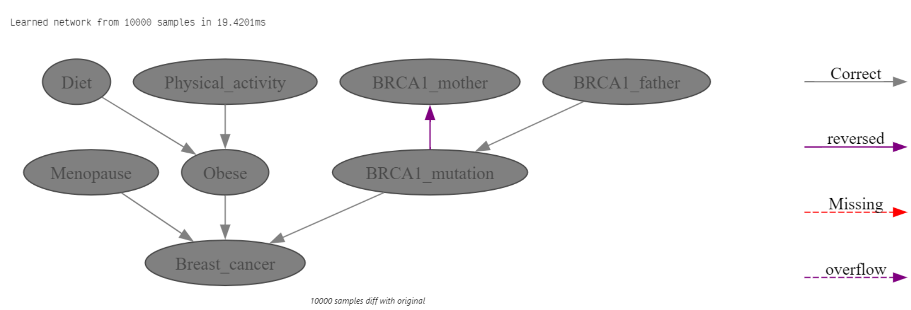


Figure 15: MIIC algorithm using 10000 samples

B References

References