Exploring the Effect of Model Assumptions on Prediction Performance of Bayesian Networks

Wessel Goslings
BSc Applied Mathematics
Delft University of Technology

February 6, 2022

During the writing of this Bachelor thesis I have learnt a great deal. Not only about the subjects at hand, but also about myself.
I would like to thank Dr. ir. Tina Nane for her never ending patience and understanding. Her guidance and assistance have helped through it all.

Abstract

This thesis concerns itself with the effect of the normality assumption, the effects of discretisation choices and other assumptions made by software on prediction performance, when using Gaussian Bayesian Networks. To test these effects, different types of Bayesian Networks are constructed and made to perform predictions, using the same dataset. The dataset used contains records regarding the citations and other bibliometric statistics of articles published by authors affiliated with the Delft University of Technology between 2010 and 2014. The first model is a Gaussian Bayesian Network (GBN), which assumes that the conditional probability distributions (CPD) of all variables concerned are Gaussian. The second model is a Multinomial Bayesian Network (MBN), which uses discrete variables. To accommodate to this model, the data is discretized. The third model is the Hybrid Bayesian Network (HBN, which can handle both discrete and continuous data and has no normality assumption on the distribution of the variables. The last model is a non-parametric Bayesian Network (NPBN). To compare the different models, they are used to perform a set of predictions in the form of quantile estimation. The results show that the GBN performs as well as the NPBN, when looking at the bulk of the data. When looking at data, where the mean citation score (mcs), the predicted variable, exceeds the 75-% quantile, the performance of the GBN becomes much worse than that of the NPBN.

Contents

Introduction

In statistics and applied probability, it is common practice to make assumptions regarding the data, when constructing a model. Examples of these assumptions are homogeneity of variance, linearity and normality for linear regression models. These assumptions ensure that the inference made using the data is statistically valid and can be used to make conclusions and decisions with a certain degree of credibility.

However, following assumptions to the letter can make a statistician's life difficult. The data never looks the way you want it to, not every distribution is normal, and outliers occur. This means statisticians have to twist and turn to transform their data into data which follows the assumptions of a model, or use a more complicated model which can be constructed with the data at hand, if they want to make sure their results are statistically valid. The last option is to ignore all assumptions and carry on with the initial model, whether its results are statistically valid or not. So are these assumptions always necessary for the model to produce meaningful results that can be useful to decision makers? There are a lot of complicated regression models to perform linear regression analysis, but do these models always outperform ordinary least squares regression? Even if the variables are not normally distributed, or other assumptions are not met, the ordinary least squares regression can be a useful tool. This raises the question whether all this extra work is necessary.

In this thesis the effect of assumptions on the prediction performance of Bayesian Networks (BNs) is tested by constructing multiple types of Bayesian networks, with different levels of complexity, using the same dataset comprising of variables with skewed distributions. A Bayesian Network is a statistical model used to model the conditional dependencies in a multivariable dataset. It does so by using a combination of graph theory and probability. There are various types of Bayesian Networks, the use of which depends on the type of data at hand. The types used in this thesis are the Gaussian Bayesian Network (GBN), which assumes normally distributed variables, the Multinomial Bayesian Network (MBN), which is meant for use in combination with discrete variables, and the Hybrid Bayesian Network (HBN) and Non-Parametric Bayesian Network (NPBN), which can use data which is discrete and continuous.

The structures of the BNs will be learned from the data using structure learning algorithms. Different algorithms will be used and their results in combination with the different forms of data used by the different types of BNs will also be discussed. When it is decided which structures the models will use, the conditional probability distributions of the variables are estimated. For the GBN, these will be Gaussian for all variables, but parameters will be learned from the data. For the MBN, they will be discrete. For the HBN, the type of distribution and its variables are also learned from the data. The NPBN does not use parametric distributions. After construction of the various models, the BNs will be compared by having them perform the same series of predictions to see which model performs best and whether the extra work really does result in a more useful model that provides more accurate predictions.

The predictions performed by the models will make use of quantile estimations. These quantile estimations could prove to be very interesting since the variables have skewed distributions. Because of this, the performance of models which assume normality can really be tested with the data and it can be seen whether there is a difference in performance between predictions made regarding the bulk of the data or regarding extreme values, as captured by high quantiles.

The data used in this paper was gathered by the Centre for Science and Technology Studies (CWTS), Leiden University, and contains records of the citations of papers published by scientists

affiliated with the TU Delft, between 2010 and 2014. Most papers do not get cited very often, but there are a few papers that will get cited a lot. These outliers cause the data to be skewed, which means it lends itself very well to be used for testing the importance of the normality assumption.
When looking at the issue of the effects of model assumptions on prediction performance of Bayesian Networks, it is to be expected that the answer is not simple. The different models could perform better in different situations and should all have their own merit. To introduce the subject, the following chapter will contain a summary of the necessary theory to conduct the research. This includes a brief refresher on Bayesian probability theory, graph theory and an formal introduction to the notion of Bayesian Networks.
Following is an exploration of the data, where the variables and their characteristics are presented, as well as several statistics regarding the data, such as the means, medians and quantiles.
When the theory and data are introduced, the modelling process will be explained. The various steps to learn the structure, the distributions and their parameters are discussed. So are the methods applied to make predictions and to compare the prediction results.
Ultimately the results and the conclusions based on the results will be presented and discussed.

1 Bayesian Networks

In this section the mathematical theory behind the modelling process will be briefly covered. Firstly, a short review on Bayesian probability and statistics, which will be integral to the research. Secondly, Bayesian networks, the model that will be investigated. And lastly, quantile estimation, which will be used to test the performances of the models .

1.1 Bayesian Probability

Bayesian probability and Bayesian statistics are two special interpretations of their respective fields and have a completely different view on probabilities, when compared to the classical frequentist approach. Instead of looking at how many times an event occurs to estimate the events probability, which is the frequentist approach, the Bayesian approach calculates a priori probabilities using prior knowledge, and adjusting the probabilities when new information arises, which results in the a posteriori probabilities. The approximation these prior distributions and their parameters does not follow a determined set of rules. The process consists of using what is known beforehand to make an educated guess. The use of these prior distributions not only improves estimations of the posterior distributions, but also quantifies hypotheses.
The basis of the Bayesian approach lies in *Bayes' Theorem*:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{1}$$

where A and B are events, $P(A|B)$ is the *conditional probability* that A occurs given that B occurs (vice versa for $P(B|A)$) and $P(A)$ and $P(B)$ are the *marginal probabilities* that events A and B occur.
The use of the Bayesian approach does of course require prior knowledge of the data and the problem to form a priori probabilities. This can be seen as a disadvantage, since there is no real set of rules on how to determine these a priori probabilities. It is however possible to use real-life knowledge for this and it also always possible to try out different priors to see, if this leads to different results.
A big advantage of the Bayesian approach is the possibility of using new knowledge to update probabilities and models without having to start over again. This makes the modelling process flexible and leaves room for contingencies.
As the name suggests, Bayesian Networks rely heavily on Bayes' Theorem and Bayesian probability in general. Bayes' Theorem is used constantly when using BNs to update the probabilities of the network with new knowledge.
Other important notions in probability theory in general are the notions of *independence* and *conditional independence*. Two events A and B are said to be independent when the following is true:

$$P(A \cap B) = P(A) \cdot P(B) \tag{2}$$

When rewritten, this equation shows the reason for this.

$$P(A \cap B) = P(A) \cdot P(B) \Leftrightarrow P(A) = \frac{P(A \cap B)}{P(B)} = P(A|B) \tag{3}$$

The expression on the right side of the arrow shows that when (2) is true, no matter the value of B, the probability of A stays the same, thus making A independent of B and vice versa.

The notion of conditional independence might be less intuitive when compared to general independence. Two events A and B are said to be conditionally independent, given C, when:

$$P(A \cap B|C) = P(A|C) \cdot P(B|C) \tag{4}$$

This means that when C is known, A and B are independent.
For example, let X be the height of a child an Y the number of words a child in third year elementary school knows. At first glance it might seem that when X is high, Y is high. However, when we add the variable Z, the age of a child, we can make these two variables, X and Y conditionally independent. X and Y are not independent, but when X is known they become conditionally independent. This conditional independence is instrumental for BNs and will return later in this thesis.

1.2 Graph Theory

Graph theory is a branch of mathematics that concerns itself with graphs, obviously. A graph is a pair of sets $G = (V, E)$ of vertices and edges[5] or, more simply put, any network of points and lines, respectively. These points and lines can represent a wide variety of subjects and ideas.
There exist various types of graphs, which will not be discussed here, since Bayesian networks are always the same type of graph, namely *directed acyclic graphs* or DAG's.
A graph is directed when all the edges are directed from one vertex, or node, to another. The edges can thus only be traversed in one direction. Examples of directed graphs are a map with only one-way streets or the different streams of a river. A path along the edges of a graph is defined as a sequence of edges, where the ending node of the previous edge is the same as the starting node of the following edge.

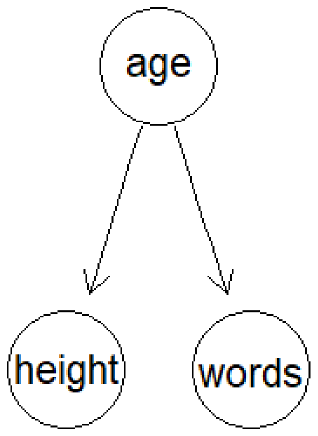


Fig. 1: Example of a directed acyclic graph.

A directed graph is acyclic, when cycles do not exist within the graph. This means that starting at any given node A, if a path is followed along the edges, you can never return to that same node A, as this would be a cycle.
In Figure 1 you can see a DAG with one *parent-node*, 'age', and two *children*, 'height' and 'words'. A more general term for children is *descendant*. The descendants of 'age' are the children of 'age', and recursively the descendants of the descendants of 'age'. The nodes used in Figure 1 come from the example in Section 1.1. The arrows at the end of the arcs indicate the direction of the arcs, so this is a directed graph. Since no cycle can be found, this is a DAG.

1.3 Bayesian Networks

Bayesian networks or BN's are intuitive, yet powerful models, used to describe dependencies, independencies and conditional independencies between stochastic variables. They combine graph theory and Bayesian probabilistic theory to do this. This combination is, as will be shown, a very efficient way to model these conditional independencies.
A Bayesian network is defined as a directed acyclic graph where the nodes represent stochastic variables and the edges represent the dependencies between these stochastic variables.
Conditional independence, which was mentioned before in Section 1.1 is a key element of Bayesian Networks, because of the following assumption, called the *local Markov property*:

Each node X_i is conditionally independent of its non-descendants, meaning nodes X_j for whicht there is no path from X_i to X_j, given its parents.

This means that if no path exists from variable A to variable B, variable A is independent or conditionally independent of variable B, based on the situation, like "height" and "number of words" in Figure 1. These two variables are conditionally independent of each other. In a BN consisting of two parents and one child, the two parents would be independent.
This assumption of conditional independence makes it possible to construct extremely large networks, without making it impractical to use them due to computations of all conditional probabilities in the entire network. Using this conditional independence of variables the models joint probability distribution function (cdf) for continuous models, and probability mass function (pmf) for discrete models, can be represented as follows:

$$p(X_1, X_2, X_3, \cdots) = \Pi_{i=1}^{n} p(X_i|X_{P_i})$$

where $p(X_1, X_2, X_3, \cdots)$ denote the cdf or pmf, X_i represent the different variables and X_{P_i} represents the set of all the parents of the node X_i. The $X(X_i|X_{P_i})$ are all the different local conditional probability distributions. The conditional independence can chop up the graph into independent subgraphs and calculate the probabilities for one part of the BN without having to use the entire network. But what is the minimal amount of nodes necessary to use for those calculations and which nodes are those? To decide whether one node is conditionally independent from another one can use *d-separation*. The idea of d-separation is that nodes that are not connected in the graph, are variables that are conditionally independent from each other. Using d-separation one can find *Markov blankets* for variables. The Markov blanket of a node A is the minimal set of nodes needed to condition upon when one wants to perform inference on said node A. This Markov blanket of node A consists of:

- The parents of node A
- The children of node A
- All other nodes that share children with node A

These nodes form the Markov blanket which d-separates node A from the rest of the network, which means A is conditionally independent from all other nodes that are not in the Markov blanket. Thus the nodes in the Markov blanket of node A are the only nodes needed to condition upon to perform inference on node A.
Looking back at the child example, it is now possible to create a BN, using the aforementioned

principles and basic probability theory. There are multiple types of BN's that can be used, depending on the characteristics of the data used. In the following sections the types of BN's used in this paper will be discussed based on the child example, after which the data used for the actual research will be introduced.

1.3.1 Gaussian Bayesian Network

The first model to discuss is the Gaussian Bayesian Network or GBN. When fitting a Gaussian Bayesian Network to the data, the assumption is that all variables are normally distributed. For convenience, it is assumed that the variables in our example are all normally distributed with the following parameters:

$$\begin{aligned} &age \sim N(6.5, 0.5), \\ &height \sim N(1.2, 0.06), \\ &\#words \sim N(6500, 3000), \end{aligned} \tag{5}$$

The parameters are based on the website Kindexpert by the Hanzehogeschool Groningen[13, 8]. This website was started with the help of expert knowledge in the fields of pediatrics and pedagogy to help parents of newborn childeren.

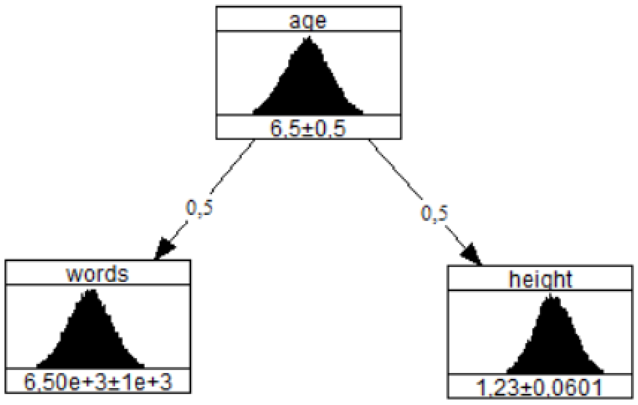


Fig. 2: Example of a Gaussian Bayesian Network, created with Uninet (LightTwist Software, 2020), a software package created for dependence modelling for high dimensional distributions.

In Figure 2 the BN is shown, where the nodes show histograms representing the distributions of the variables with the means displayed below and the arcs show the correlations between the variables. These correlations were estimated to fit the information from Kindexpert. The Uninet software allows its users to put in known values of variables and update the BN. This grants the possibility to illustrate conditional independence in action. First, suppose only the value of "words" is known to be 8000. Then the conditional expected value of "height" increases from 1.23 to 1.25, as shown in Figure 3, and the expected age of a child who knows 8000 words is 6.89. Now suppose the value of "age" is known to be 8. Then the expected value of "height" increases from 1.23 to 1.32, see Figure 4. If following this, the value of words is known to be 5000, the expected value of height stays 1.32, see Figure 5. Thus the variables "words" and "height" are independent when "age" is known.

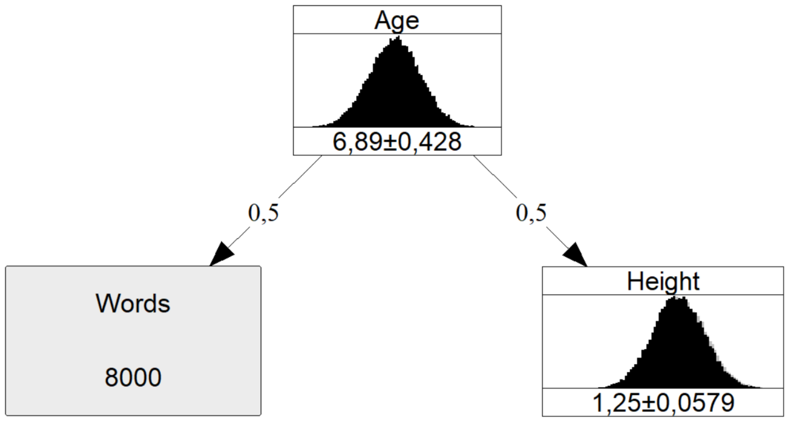


Fig. 3: Example of a Gaussian Bayesian Network with known variable word value 8000, created with Uninet.

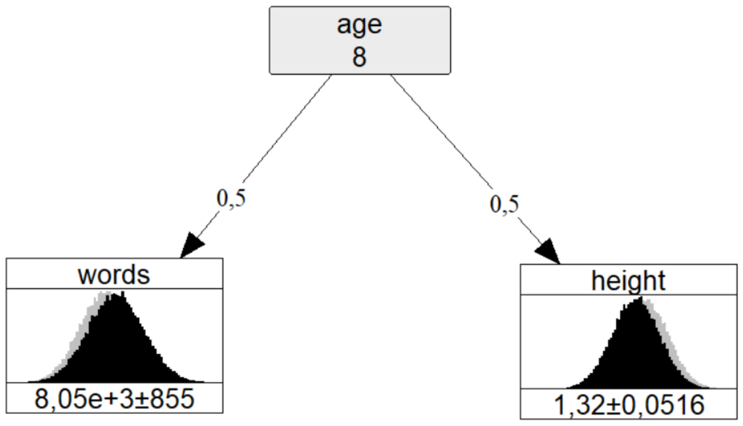


Fig. 4: Example of a Gaussian Bayesian Network with known variable age value 8, created with Uninet.

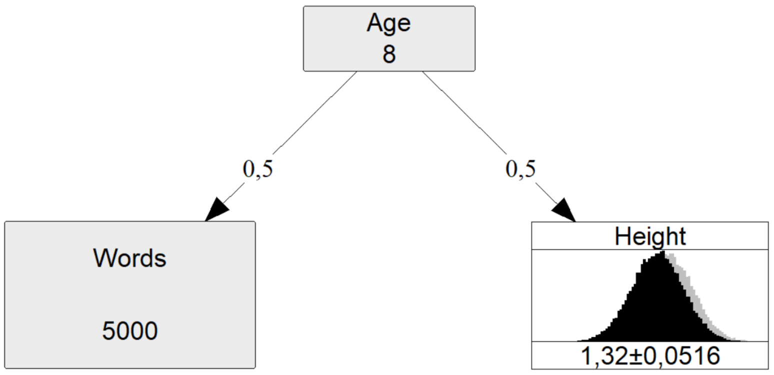


Fig. 5: Example of a Gaussian Bayesian Network with known variable age value 8 and known variable words value 5000, created with Uninet.

1.3.2 Multinomial Bayesian Network

The next model is the Multinomial Bayesian Network, or MBN. The MBN assumes that the variables follow discrete distributions. To implement these variables in the MBN, conditional probability tables are used. For this reason, the variables from the child example will be discretized and *conditional probability tables* will be used to construct the MBN. The discretization process will be rudimentary in this case. The 'age'-variable will be split in four intervals with probabilities 10%, 40%, 40% and 10%, while the other two variables will be split in three intervals with probabilities 25%, 50% and 25%. Because of the use of discrete variables, the MBN is faster and easier to compute, compared to the GBN. The downside is however a loss of information, compared to models using continuous variables. The MBN looks as follows:

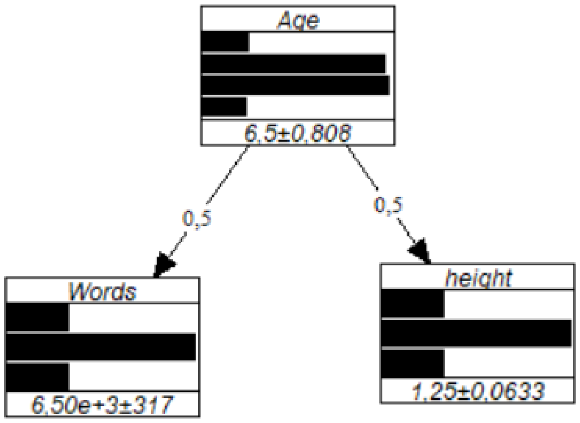


Fig. 6: Example of a Multinomial Bayesian Network, created with Uninet.

1.3.3 Hybrid Bayesian Network

you distribution.

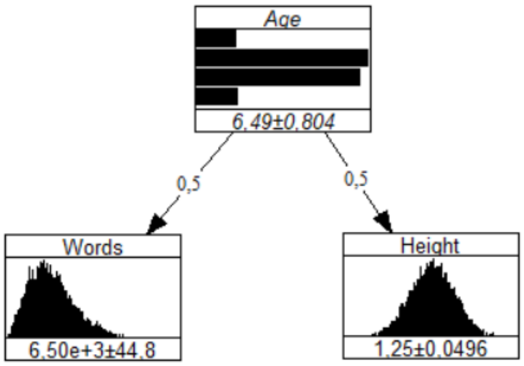


Fig. 7: Example of a Hybrid Bayesian Network, created with Uninet.

1.3.4 Structure Learning

In the previous sections, the structure of the BN's was based on what is called *expert knowledge*. The properties of the variables were already known before constructing the models. Expert knowledge is very useful when the conditional dependencies in the data and the underlying distributions are already known to the statistician, or when the statistician can deduce them from their real-world knowledge. However, more often than not, these dependencies and distributions are not clear and not easy to deduce because of the complexity of the subject. If this is the case, the use of *structure learning algorithms* becomes necessary. This section contains a brief overview of these structure learning algorithms. More detailed explanations can be found in [9, 11, 1].
Structure learning algorithms use data to learn the graph structure of a BN. There are several structure learning algorithms and they can be divided into three main classes: constraint-based algorithms, score-based algorithms and hybrid algorithms.
The constraint-based algorithms test the variables for conditional independence by means of statistical testing, and place arcs between variables that are found to be conditionally dependent. Examples of such algorithms are: Parent & Child (PC), Grow-Shrink (GS) and Hybrid Parents & Child (HPC). These algorithms all go about the process of learning by testing in a different way.
There are multiple versions of the PC-algorithm, but it all comes down to the following basic structure. The algorithm starts with a fully connected graph, where all nodes are connected to each other. Then it starts removing arcs, if they don't pass the conditional independence test. The HPC is a combination of some of the different PC-algorithms. Thus the 'Hybrid' in HPC does not mean that it is considered as a hybrid structure learning algorithm, but that it is a hybrid form of different PC-algorithms. These are not to be confused, as the HPC-algorithm does not use score-based methods hybrid algorithms do.
The Grow-Shrink algorithm, however, works differently. The algorithm consists of two phases: a growing phase and a shrinking phase. This algorithm starts with an empty graph and keeps adding variables to the graph as long as there are variables that are dependent on the variables in the existing graph, the growing phase. In this process it is possible that variables and arcs were added that are no longer dependent, because of new arcs that render them unnecessary. Then comes the shrinking phase that tests the conditional independence of all the variables and thus identifies the arcs and variables that can be removed.
The last constraint-based algorithm that is used in this research is the Incremental Association Markov Blanket (IAMB)-algorithm. This algorithm tries to find Markov Blankets for each variables. If one knows the Markov Blankets for each node, there is no more need to test for independence on the whole set. The algorithm tries to find the Markov blanket $B(X)$ for each variable X, by making a hypothesis set $H(X)$. The algorithm consists of two phases, a *forward phase* and a *backward phase*. In the forward phase the algorithm adds nodes to the $H(X)$ that might belong to $B(X)$ by looking for associations between the nodes and X, conditional on the other nodes in $H(X)$. These associations can be measured in different ways. When $H(X)$ is large enough and no more nodes with associations with X are found, the backward phase starts and the algorithm removes the false positive, the nodes from $H(X)$ that show the weakest association and should not belong to $B(X)$. This goes on until only $B(X)$ remains and we have found the Markov blankets for all nodes. So, while these three algorithms all learn the structure of the BN using conditional independence testing, they use very different approaches.
The score-based algorithms assign a *network score* to a structure based on goodness-of-fit of the model to the data and compares it with the score of another structure to find which structure

fits the data best. There are multiple metrics to measure the goodness-of-fit and network score. Popular score functions are the *Akaike Information Criterion* (AIC), *Bayesian Information Criterion* (BIC) and *Bayesian Dirichlet* (BD) score. These metrics can lead to different scores and ultimately different final BNs, as the various metrics can place more scrutiny on certain aspects of the network, such as number of nodes or vertices, or overfitting. It is up to the user to decide which aspects warrant the most scrutiny in the situation at hand. The most commonly used score-based algorithms are the Hill-Climbing (HC) and Tabu-search.
The HC-algorithm and Tabu-search are both called *greedy search* algorithms. they start of with a graph, which is usually empty, and then keep adding, deleting and reversing arcs of the graph, until the score of the graph can no longer be improved. At this point they have arrived at the 'optimal' graph. The difference lies in the fact that the HC-algorithm adjusts the graph for every possible change that increases the score, while the Tabu-search looks for the change that increases the score the most.
Then there are also hybrid learning algorithms which combine the two other types of algorithms. This is usually done by taking two steps: restricting and maximising. First the structure is restricted by use of conditional independence testing, like the constraint-based algorithms, followed by maximisation of the goodness-of-fit by use of network score optimisation, like the score-based algorithms. These steps are repeated until the optimal network is discovered. Examples of these hybrid learning algorithms are Max-Min Hill-Climbing (MMHC) and Hybrid HPC (H2PC).[7]
Scutari et al. (2019) showed that there is no definitive answer yet to the question which class and which algorithm perform best in terms of speed and accuracy.[9] This research performed a rigorous assessment of different performance metrics of both score-based and constraint-based structure learning algorithms, using different scoring funcions, both discrete and continuous BNs, and simulated data. Based on the results of this research, it is advisable to use different classes of algorithms to find the optimal structure for the specific situation and the data at hand.

1.4 Quantile estimation

Bayesian Networks, like other statistical models, are used to perform statistical inference on data. A common example of such inference is finding the conditional mean, $E(Y|X_1, X_2, ..., X_n)$, of a variable, with knowledge of the other variables. However, one can also be interested in other statistics of the conditional distribution. For example, for which values x_i of the prediction variables X_i and which value y of Y is there a 5% chance that this value will be exceeded, $P(Y \geq y|X_1 = x_i, ..., X_n = x_n)$? This is a question of *quantile estimation* . A *θ-th quantile* is defined as a value y such that:

$$P(Y \leq y) = \theta$$

.[3] Quantiles are often used when exploring data, by inspecting the 50%-th quantile, also known as the median, and the $25\% - th$ and 75%-th quantiles, used in constructing a box-plot.
When distributions are sufficiently skewed, it becomes more and more interesting to look at quantile in the tails of these distributions. Think of 90%-th or 95%-th quantiles. In these parts of the distribution one can sometimes find more interesting information than in the bulk of the distribution. In the context of citations, it could be more interesting to see if there are certain characteristics that lead to extreme numbers of citations instead of searching for the mean number of citations.
Since this research focuses on data which is not normally distributed, it could be informative to

perform quantile estimation with higher quantiles.

2 Data Exploration

To test the importance of the normality assumption the Gaussian Bayesian network makes, different models are built based upon the same dataset. The data used in this paper was gathered by the Centre for Science and Technology Studies (CWTS), Leiden University. It contains records of contains records regarding the citations and other bibliometric statistics of 10726 papers published in the period from 2010 until 2014, with at least one author affiliated to the TU Delft. The following variables are included in the data:

- MCS : Mean citation score, the average number of citations from a given publication from publication until the end of 2017;
- JS: Journal citation score, the average citation score of publications in a journal;
- N_refs: number of references in the paper;
- N_authors: number of authors of the paper;
- P_max: maximum number of publications of all the authors of the paper;
- MCS_max: maximum mcs of all papers of all authors of the paper;
- pp_top_10_prop_max: maximum percentage of authors; publications in the top 10% of their field of all the publications of all authors of the paper;
- academic_age_max: maximum difference in the publication year of the paper and the first publication of all authors of the paper;

As seen in Figure 8, the histograms in the dataset are skewed for the most part and thus clearly not normally distributed. This is an important property of the dataset, since the goal of the research is to test the importance of the assumption of normally distributed data for BNs. Because the data is clearly not normally distributed, the assumption will not hold.

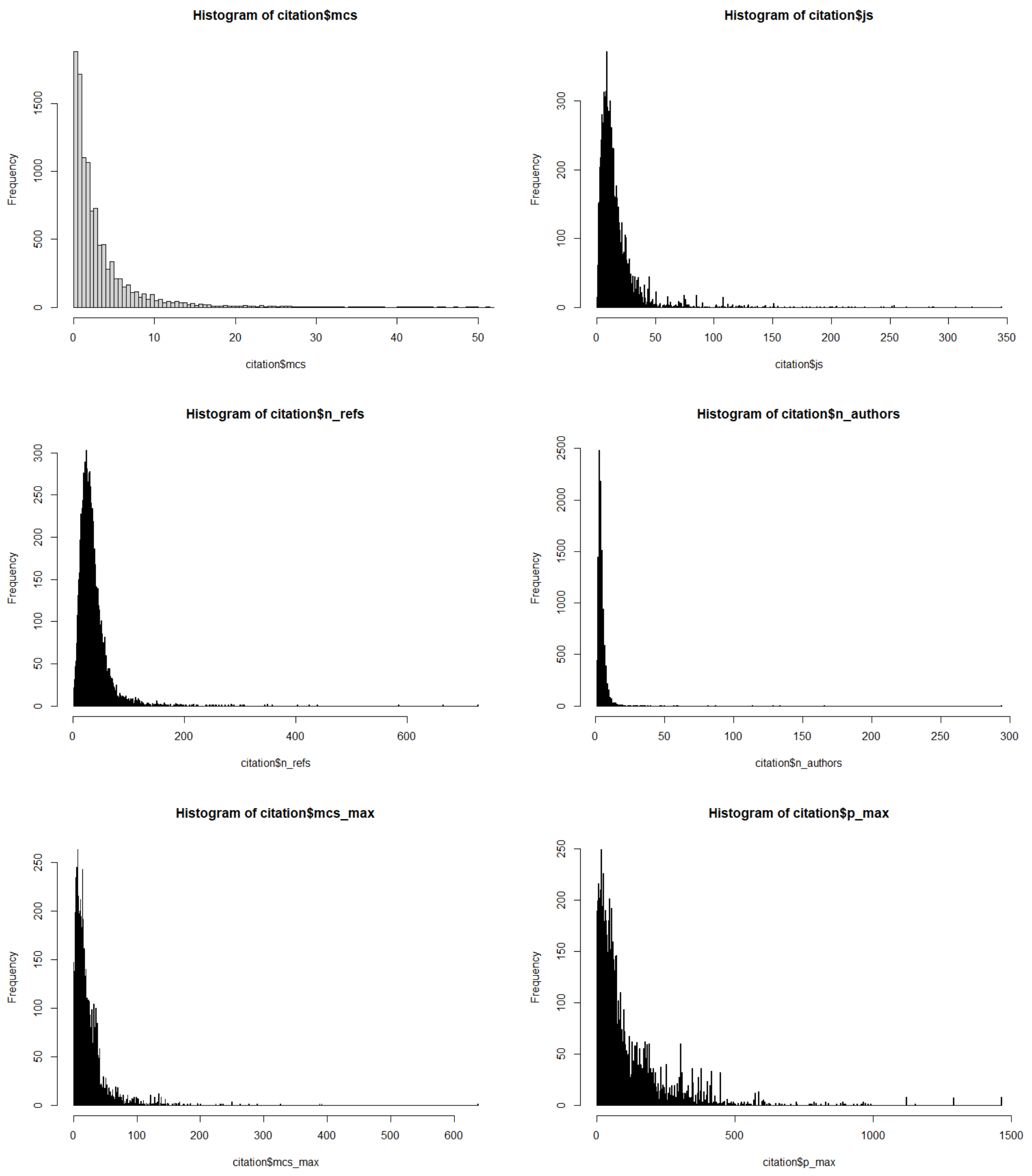


Fig. 8: Marginal distributions of the variables mcs, js, n_refs, n_authors, mcs_max and p_max in the dataset of citation scores of papers published by authors affiliated to the TU Delft.

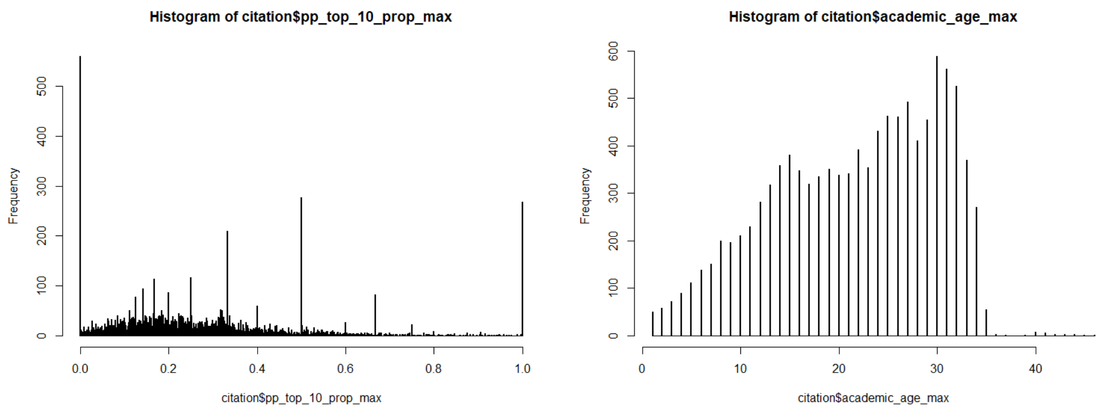


Fig. 9: Marginal distributions of the variables pp_top_10_prop_max and academic_age_max in the dataset of citation scores of papers published by authors affiliated to the TU Delft.

Variable	Mean	Max	Min	Median	75-th quant.	90-th quant.
mcs	3.664	297.8	0	2	4	8
js	16.150	344.631	0	11.618	18.602	29.164
n_refs	35.720	727	1	30	43	60
n_authors	4.670	294	1	4	5	8
p_max	104.621	1464	1	61	139	251
mcs_max	20.599	637.026	0	14.333	26.248	40.662
pp_top_10...	0.272	1	0	0.232	0.354	0.512
academic_age..	21.617	46	1	23	29	32

Tab. 1: Exploratory statistics of the variables in the dataset of citation scores of papers published by authors affiliated to the TU Delft.

The first six variables all show similar skewed distributions with long right tails in Figure 8. The fact that the mean is higher than the median for all these variables is in agreement with this. The length of the tail varies with each variable, as can also be seen in Table 1, by inspecting the higher quantiles and maximum values. The pp_top_10_prop_max and academic_age_max show different types of distributions, shown in Figure 9 .
The pp_top_10_prop_max is a percentage of usually low numbers, since most scientists don't publish enormous amounts of papers and 90% of these papers are of course not included in the top 10% of their field, so most authors have 0 in this variable. Authors who have written only one paper, which happened to be very good, and was thus in the top 10% of their field, will have had 100% of their papers considered in the top 10% of their field, resulting in the value 1. as seen in Figure 9, this value occurs quite often, which suggests that there are a lot of scientists who could be called one-hit wonders. The same reasoning explains the high number of occurrences of 0.25, 0.5, 0.33

and 0.67, for authors with two, three or four papers.
The academic_age_max variable shows a large tail to the left and a sudden drop after 35. It would be fair to assume that a scientists first publishes a paper under their own name around the time they are 30. In the Netherlands most people retire around the time they are 65, so the sudden drop after 35 would be explained by the retirement of most scientists. Still the values around 30 occur the most. This is probably because all people looking to earn their doctorate are mentored by a professor, who will also be accredited as an author. These professors, who are usually senior members of the scientific community, cause the high values of academic_age_max.

Figure 10 shows a correlation matrix which displays the Pearson's correlations between the different variables. Pearson correlation tests were also performed to test the significance of the correlations and all correlations turned out to be significant.
Figure 10 shows that most correlations are not very strong. The strongest correlations exist between:

- mcs and js : 0.47;
- p_max and academic_age_max: 0.53;
- mcs_max and pp_top_10_prop_max: 0.51;
- p_max and mcs_max: 0.43;

The correlation between mcs seems logical since journals which get cited often will contain papers that get cited more often. Therefore the correlation between the two variables could be stronger.
The correlation between p_max and academic_age_max can be explained by the fact that scientists with a longer career have published more papers.
The mcs_max and pp_top_10_prop_max could show a stronger correlation because an author with more papers in the top 10% in their field would naturally be cited more.
It would also be fair to assume that the more papers a scientist publishes, the more cited they get, because of their gained fame and respect in their field. Since correlation should not be confused with causation, these interpretations of the strong correlations could also be vice versa. For example, because of the high mcs of the papers, the js of the journal could be higher. Table 3 shows that the correlations between the variables are all significant with extremely low p-values, even the weak ones. This could be explained by the large size of the dataset. This statistical significance does not equal practical relevance. Most of these correlations will not result in arcs between the variables, when included in a BN. The variables that show strong correlations would be more likely to have arcs between them. It can not be inferred from this information which direction the arc would have.

To determine whether the variables are not normally distributed, Shapiro-Wilk tests [12] were performed on all variables. The Shapiro-Wilk test is a goodness-of-fit test for normal distributions, where the null hypothesis is that the sample is normally distributed. The results are shown in Table 2. Since all p-values are infinitesimal, it follows that none of the variables are normally distributed.

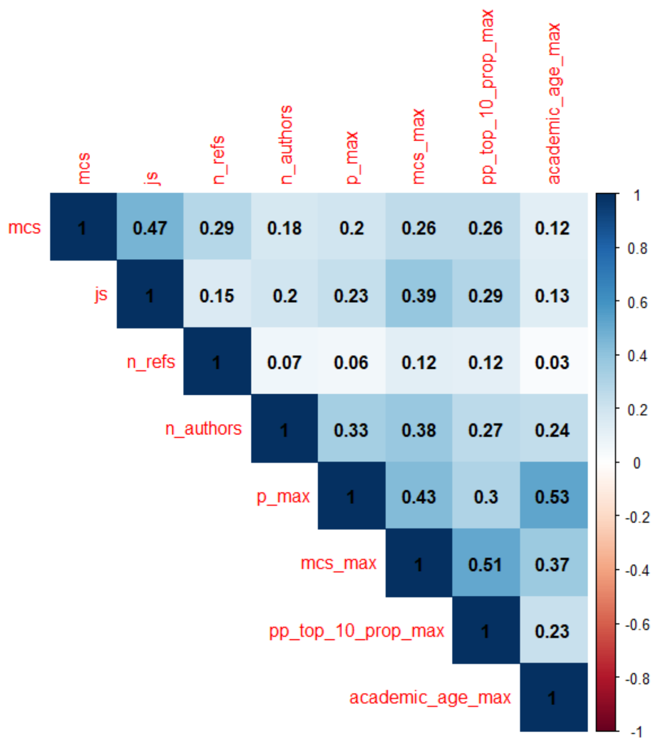


Fig. 10: Correlation plot of all variables contained in citation-dataset

Variable	p-value
mcs	<2.2e-16
js	<2.2e-16
n_refs	<2.2e-16
n_authors	<2.2e-16
p_max	<2.2e-16
mcs_max	<2.2e-16
pp_top_10_prop_max	<2.2e-16
academic_age_max	<2.2e-16

Tab. 2: P-values of Shapiro-Wilk tests performed on all variables contained in citation-dataset

	mcs	js	n_refs	n_authors	p_max	mcs_max	pp_top_10...	academic_...
mcs	0	0	1.7e-200	3.5e-77	1.2e-94	7.4e-160	5.3e-164	2.5e-37
js	0	0	1.9e-55	3.2e-95	8.5e-132	0	5.2e-211	2.7e-43
n_refs	1.7e-200	1.9e-55	0	1.3e-12	3.4e-09	2.6e-37	3.1e-35	2.1e-03
n_authors	3.5e-77	3.2e-95	1.3e-12	0	6.6e-278	0	1.7e-177	2.2e-143
p_max	1.2e-94	8.5e-132	3.4e-09	6.6e-278	0	0	1.6e-225	0
mcs_max	7.4e-160	0	2.6e-37	0	0	0	0	0
pp_top_10...	5.3e-164	5.2e-211	3.1e-35	1.7e-177	1.6e-225	0	0	7.4e-124
academic_age...	2.5e-37	2.7e-43	2.1e-03	2.2e-143	0	0	7.4e-124	0

Tab. 3: P-values of Pearson's correlation tests performed on all variables.

3 Methods

In this section the modelling process will be discussed. Four different models are used. These different models approach the data each in a different way.
All models will first use structure learning algorithms to learn the structure of the network. These algorithms will be used in the bnlearn-package (M. Scutari, version 4.7, 2020) in R. When deciding on the structure of the network for the model, multiple structure learning algorithms are used. These are the following :

- Grow-Shrink (GS) Algorithm (constraint-based);
- PC (constraint-based);
- Incremental Association Markov Blanket (IAMB) (constraint-based);
- Hybrid Parents and Children (HPC) (constraint-based);
- Hill-Climbing (HC) (score-based);
- Tabu-search (Tabu) (score-based);
- Max-Min Hill-Climbing (MMHC) (Hybrid);
- Max-Min Parents Children (MMPC) (Hybrid);

Four separate processes of structure learning for four different sets of data will take place. The first set of data contained the unaltered data with continuous variables, as they were presented in Section 2. The second, third and fourth sets contained the discretised variables, using the Equal Frequency, Equal Interval and K-means Clustering methods, respectively.
Every structure-learning process has the same setup. First of all, each algorithm is used with the data, to see if it resulted in a graph that consists of one part. Then the fit of each structure with the data is measured using the Bayesian Information Criterion (BIC) scoring function.[2] Then these different structures will be compared using *k-fold cross-validation* to make an informed decision on which structure should have the best predictive power. K-fold cross-validation splits the data into training and testing sets, by creating k folds. Each fold is used once as a testing set, while the other k-1 folds are used to train the model. By performing k-fold cross-validation,

the models ability to perform predictions using new data is tested. This way, overfitting can be avoided. In this case 10 folds will be used to over 100 runs. The loss function in this case will be a log-likelihood loss function.

3.1 Gaussian Bayesian Network

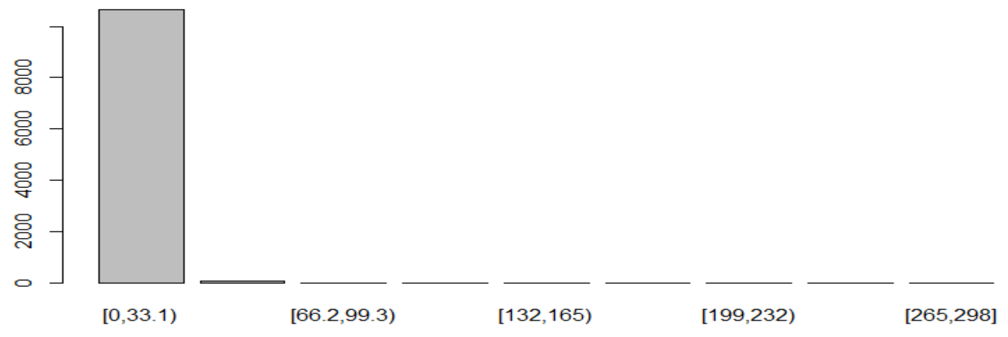
First of all, the Gaussian Bayesian network. The data will not be altered when using this model. Because of this the normality-assumption will be ignored, while the model still assumes the variables are normally distributed and fits normal distributions to them. The GBN will be programmed using the `bnlearn`-package in R.

3.2 Multinomial Bayesian Network

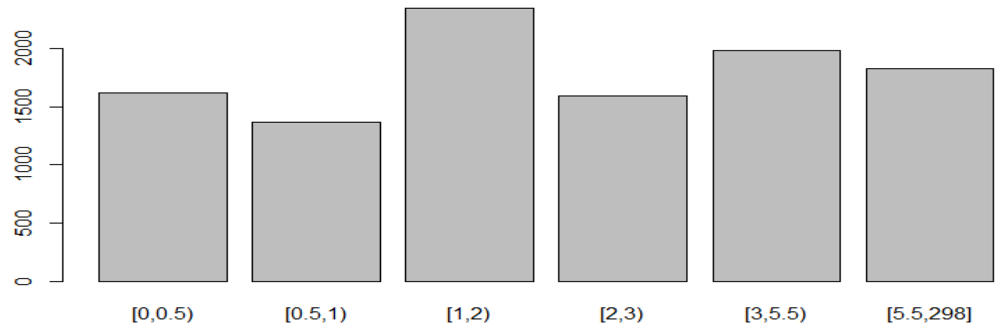
Following, the Multinomial Bayesian network. The data will be discretized, so it is possible to fit this model to the data at hand. There are multiple discretization methods. Three of them will be used here:

- Equal Interval: Creating n intervals of equal length;
- Equal Frequency: Creating n intervals with equal parts of the data;
- K-means clustering: assign data-points into K clusters according to the mean of the cluster they are closest to;

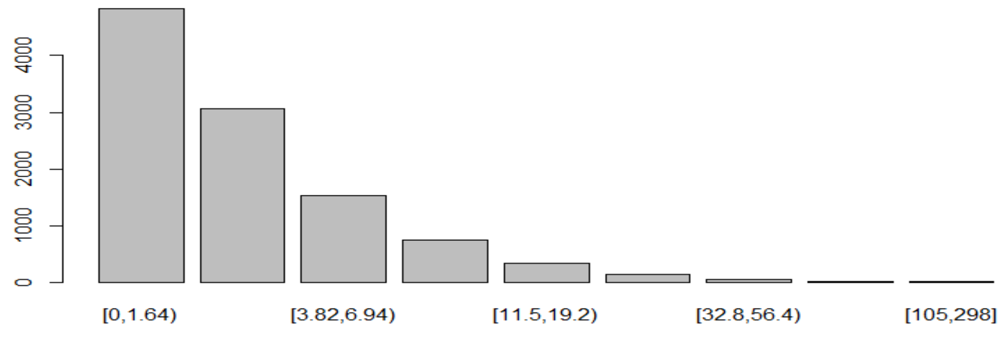
Figure 11 shows the distribution of mcs after using these three discretization techniques. The first histogram shows that the bulk of the data-points lies in the interval $[0, 33.1)$, and the other bins are almost empty. The second histograms shows six bins with similar heights. The equal frequency discretization method could not handle more breaks in this case, which is why this distribution contains six intervals instead of ten like the other two distributions. It should be noted that the bins have different interval sizes. This is because of the long right tail of the original distribution of the data, which contains very few instances. The lack of instances has to be compensated to create intervals that contain similar numbers of data-points. The distribution of the last histogram seems to stay closest to the original distribution and would thus result in a smaller loss of information, compared to the other two methods. Consequently, it is to be expected that models that are built with data that is discretized using the K-means clustering method should perform better than models that are built with the other two discretization methods.



(a) Equal interval



(b) Equal frequency



(c) K-means clustering

Fig. 11: Distribution of mcs variable after discretization with the equal interval, equal frequency, K-means clustering methods, respectively.

After discretizing the data, the MBN will be fitted with the three different discretized data sets to see which of the three models performs the best prediction. The discretization process will be handled by the `arules`-package in R and the MBN itself will be implemented using the `bnlearn`-package in R.

3.3 Hybrid and Non-parametric Bayesian Network

3.3.1 HydeNet

Next, the Hybrid Bayesian network. This model will not change the data and will try to fit distributions to the data that fit better than normal distributions. The HBN is the model that requires the most effort to construct. The `bnlearn`-package does not support hybrid Bayesian networks. This means that more human work is required to build such a network. The package `HydeNet`-package (J.E. Dalton and B. Nutter, version 0.10.11, 2020)[4] does support Hybrid Bayesian networks, but cannot learn their structure or the distributions of the variables if they are not normal. This means that the structure and the distributions have to be presented to the package. This is why the structure learned by the bnlearn-packages will be assumed while using the `HydeNet`-package. Afterwards, the package can be used to perform inference. Because of this the structure learned by using the `bnlearn`-package is used to build the network in `HydeNet`-package and distributions are fitted to the variables manually using the `fitdistrplus`-package in R. This is to say that every variable is considered separately and a marginal distribution is fitted to this variable. These distributions and the fitted variables are plugged into the network and used to perform inference. During the modelling process, the `HydeNet`-package proved to be insufficient. It could not make predictions using non-normal distributions and since this was the reason to use this package and not the `bnlearn`-package, the modelling of the HBN was moved to Uninet.

3.3.2 Uninet

Uninet can handle the prediction process of HBN's. However, the Uninet-software can only create a conditional sample for one case at a time. To do this thousands of times would be extremely impractical. This is where the UninetEngine comes in. The UninetEngine allows users to apply Uninet functionalities in a programming language of their choice. Dan Ababei of the Uninet-team was kind enough to grant a license for the UninetEngine. By using the UninetEngine in combination with R it is possible to programmatically make predictions for each case. One issue that arises when doing this however is that the UninetEngine cannot handle instances that lie outside of the range of the distribution that was used for the model. When there are more extreme values in a test-set than a training-set this causes the program to fail. To surmount this problem it seems best to forget the estimated distributions and base the model directly on the data, by using empirical distributions and thus use a non-parametric Bayesian Network. The Uninet-software supports this and by doing this their are no instances that lie outside the range of the distribution, because the distribution is defined exactly by the dataset.
As for the structures, the same goes for Uninet as for the `HydeNet`-package. The structures used are the structures learned `bnlearn`-package.

3.4 Model Comparison

First the performance of the various structure learning algorithms for the different types of BN will be assessed. These structures could turn out to be different for the different types of data that are presented to the algorithms, and could lead to interesting results.
After the structure learning there will be three models to assess, which will be one GBN, one MBN and one NPBN. To do so the data will be divided into an 80/20 split. The distributions of the models will be fitted using the training set and then the models will perform the a series of predictions on the test set to see which of the models performs the best. To inspect whether a certain model performs better when presented with fewer variables, predictions will be made using one known variable, two known variables, three known variables and one prediction where all variables are known except the mcs. The variables with the strongest correlation to the mcs will be used to make these predictions with fewer variables. These variables are, from strongest correlation to weakest, js, n_refs and mcs_max.

4 Results

In this section the results of the research will be discussed. In the first section, the results of the structure-learning algorithms on the different types of data will be displayed and discussed. Then performance of Multinomial Bayesian Network, followed by the Gaussian Bayesian Network. Lastly, the non-parametric Bayesian Network will be discussed.

4.1 Structure Learning

The various structure learning algorithms displayed very different performances and results. Table 4 shows the BIC-scores for the different algorithms and datasets. Some scores could not be computed as the graphs were only partially directed, which means that the algorithm could not learn from the data which direction an arc had and thus left it without direction. As can be seen, the HPC-algorithm does not result in a useful graph for any of the data-types. Overall the constraint-based algorithms, GS, PC, IAMB and HPC, do not perform well. Especially the cluster and continuous datasets seem hard to work with for these constraint-based algorithms.
The score-based algorithms perform better, where the resulting structures could be scored. Also the MMHC, which is a hybrid algorithm, which is based on the score-based HC-algorithm, performs well.

	Equal Interval	Equal Frequency	Cluster	Continuous
Grow-Shrink	-54139.59	-153656.8	N/A	N/A
PC-Stable	-52864.49	-144665.8	N/A	N/A
IAMB	N/A	-149983	N/A	N/A
HPC	N/A	N/A	N/A	N/A
Hill-Climb	-52300.03	-138961.6	-125495.3	-312096.7
Tabu-search	-52300.03	-138870.5	-125495.3	-312096.7
Max-Min Hill-Climb	-52360.29	-138961.6	-129374.7	-312100
Max-Min Parent-Child	-52360.29	-139420.5	N/A	N/A

Tab. 4: BIC-scores for structure-learning algorithms used in combination with different datasets

Besides comparing the algorithms using the score-function, cross-validation was also performed. The results of the cross-validation are shown in Table 5. As the loss function was the log-likelihood function, the lower the score, the better the result. Again, in some cases the algorithms did not produce fully directed graphs and losses could not be computed. The results are similar to those of the normal score function. The HC-, Tabu- and MMHC-algorithms perform the best overall and will be used in further steps.

	Equal Interval	Equal Frequency	Cluster	Continuous
Grow-Shrink	4.985776	13.89468	12.3074	N/A
PC-Stable	4.869717	13.46307	12.79708	N/A
IAMB	4.78243	13.03657	12.00172	N/A
HPC	4.73528	N/A	N/A	N/A
Hill-Climb	4.734186	12.88314	11.50621	29.24099
Tabu-search	4.732102	12.88271	11.50629	29.24553
Max-Min Hill-Climb	4.773989	12.95684	12.1993	29.23716
Max-Min Parent-Child	4.772483	N/A	12.19541	N/A

Tab. 5: Cross-validated average loss over 100 runs of 10 folds of different structure learning algorithms on different datasets

Figure 12-Figure 23 contain strength plots of the structures learned from the four different datasets with these three algorithms. Strength plots display the structure of the graph and the thickness of the arcs represent the strength of the arc. The strength is measured by removing and adding the different arcs and measure the loss/gain using scoring functions. As seen in Figure 12-Figure 23, The different datasets result in completely different structures.
The set with EI-discretised data did not result in any structure with a connected graph, where every variable was included in one graph. This set of data did not offer a lot of information as most of the data was collected in the first interval. All three algorithms resulted in different structures. The arc from p_max to academic_age_max was strong in all of them.
The EF-discretised data resulted in three connected graphs. This suggests that it was the easiest to work with for the algorithms. The HC- and MMHC-algorithms resulted in the same structure, while the Tabu-algorithm gave a different one. Strikingly, the Tabu-algorithm switched the direction of a lot of arcs, next to adding and removing a few.
The structures based on the clustered data show very different results. Here the HC- and Tabu-algorithms produced the same structures, while the MMHC-algorithm could not produce a connected graph. It does however show two groups of nodes that are related in most networks. It shows the strongest arc from p_max to academic_age_max, and it shows the mcs, js and n_refs connected
When the structures are learned from continuous data, they immediately become more dense, meaning there are more arcs. Most of these arcs are also stronger than their counterparts from the structures learned with discretised data. Because of this, one can have more confidence in these networks. Again the HC- and MMHC-algorithm produce the same structure. This is not strange as the MMHC is the hybrid version of the HC. The Tabu-algorithm created a very similar structure, where the directions of the arcs between n_refs and mcs, and js and p_max were changed. It is striking that the mcs-node is located very high in the BN in all cases. The BNs that were created using the HC- and MMHC-algorithms have a mcs-node without parents. Since this research analyses the way the other variables influence the mcs, this is a strange result. It should

be noted that it was possible to instruct the algorithms to blacklist and suppress networks where the mcs-node had children, but these networks did not have as good a fit as the networks that are shown here. The difference of goodness-of-fit were however not large.
There are some similarities we can find between the structures learned using different forms of the data and different algorithms. The first thing that is clear is a strong arc from p_max to academic_age_max. This arc is present in all twelve structures, with always the same direction. It is not a surprise that this arc is so strong, as Figure 10 showed that the Pearson's correlation found between these two variables, is the strongest in the entire dataset. The arc from mcs_max to pp_top_10_prop_max also occurs often, while not nearly as strong as the one from p_max to academic_age_max. The direction also varies. The only structure where there is no arc between these two nodes, is the structure learned from the clustered data using the MMHC-algorithm, which is not a connected graph.
The arcs between mcs and n_refs, and mcs and js are also frequently seen. While not as strong as the p_max to academic_age_max, these two arcs are present in all of the connected graphs, while the direction of the arcs may vary. The correlations between these variables are again strong.
Especially the structures learned from continuous data display strong arcs where there are strong correlations.

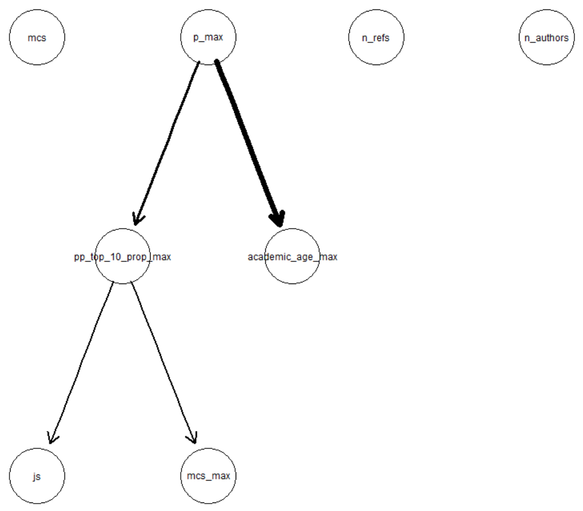


Fig. 12: Strength plot of structure learned using Hill-Climb algorithm in combination with data, discretised using Equal Interval method.

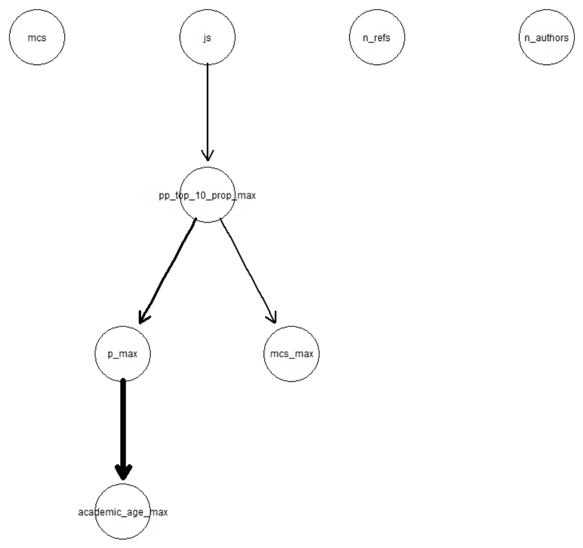


Fig. 13: Strength plot of structure learned using Tabu-search algorithm in combination with data, discretised using Equal Interval method.

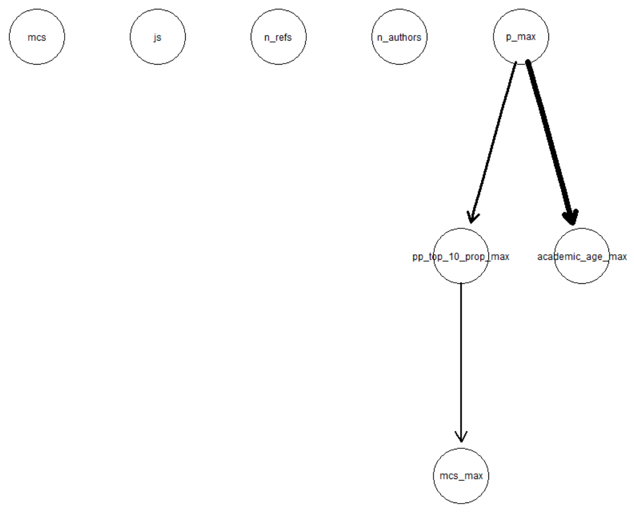


Fig. 14: Strength plot of structure learned using Max-Min Hill-Climb algorithm in combination with data, discretised using Equal Interval method.

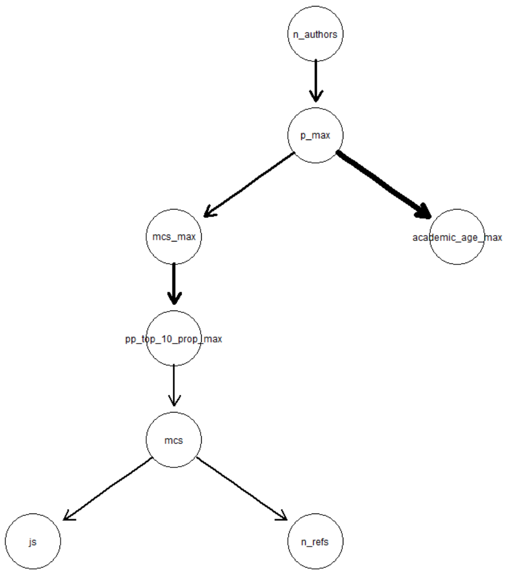


Fig. 15: Strength plot of structure learned using Hill-Climb algorithm in combination with data, discretised using Equal Frequency method.

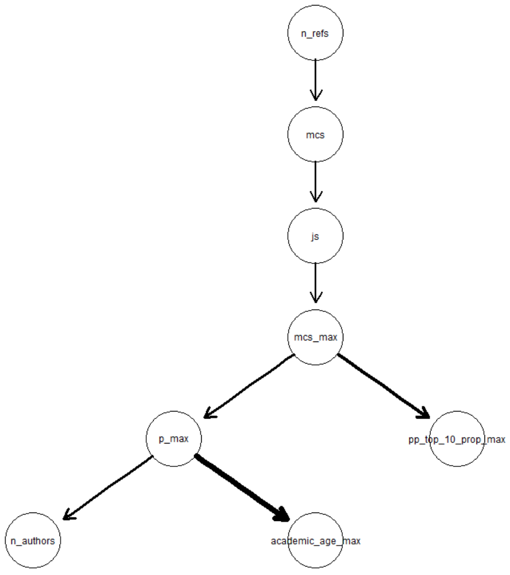


Fig. 16: Strength plot of structure learned using Tabu-search algorithm in combination with data, discretised using Equal Frequency method.

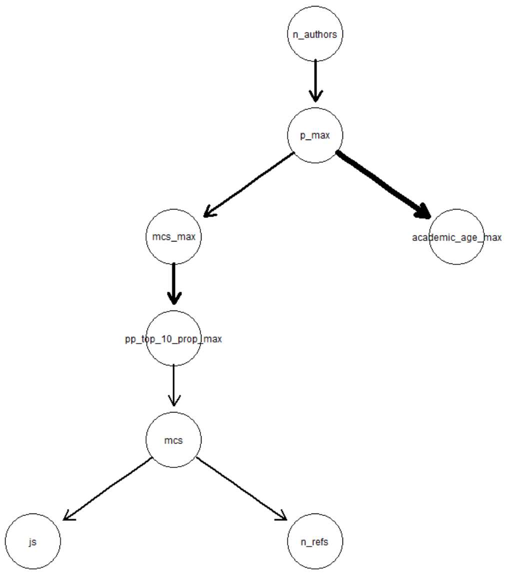


Fig. 17: Strength plot of structure learned using Max-Min Hill-Climb algorithm in combination with data, discretised using Equal Frequency method.

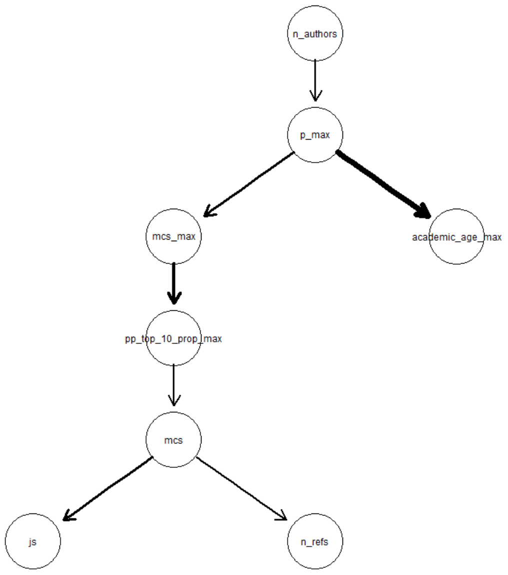


Fig. 18: Strength plot of structure learned using Hill-Climb algorithm in combination with data, discretised using Cluster method.

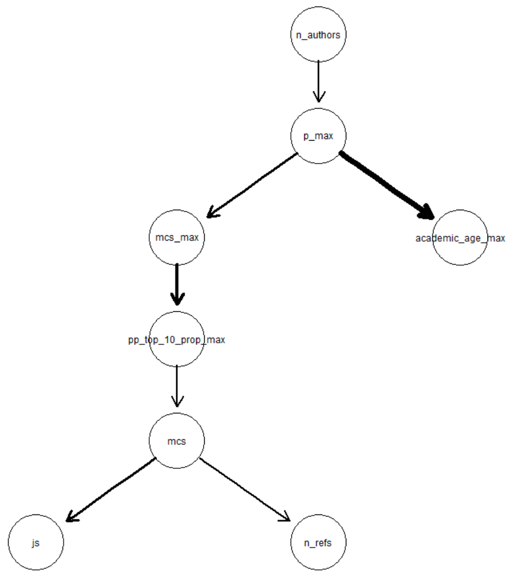


Fig. 19: Strength plot of structure learned using Tabu-search algorithm in combination with data, discretised using Cluster method.

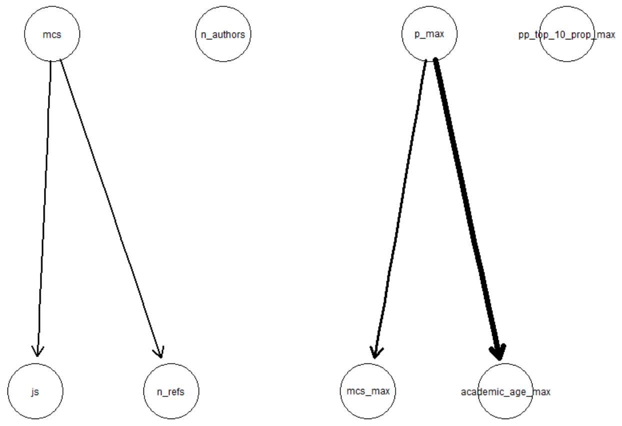


Fig. 20: Strength plot of structure learned using Max-Min Hill-Climb algorithm in combination with data, discretised using Cluster method.

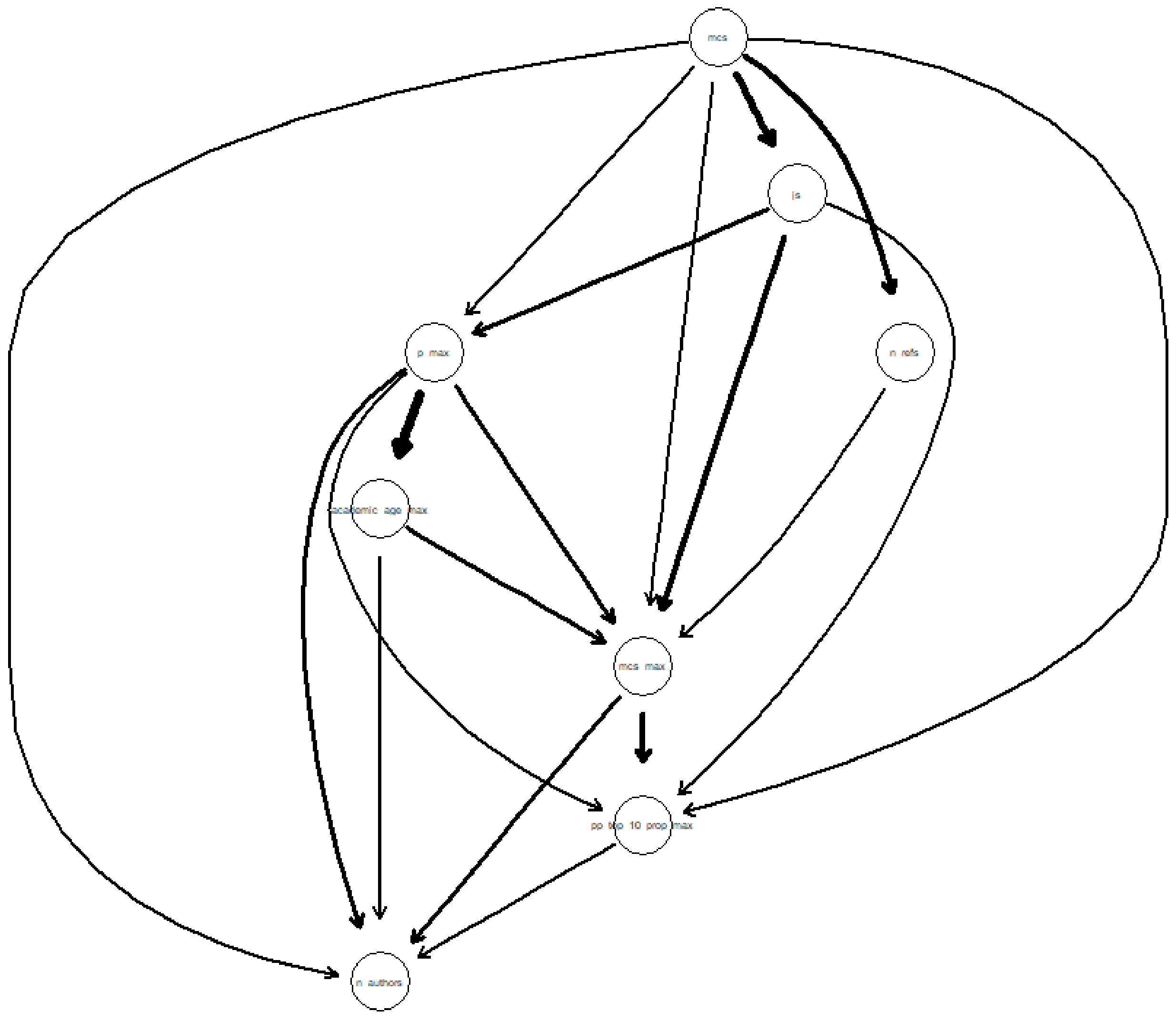


Fig. 21: Strength plot of structure learned using Hill-Climb algorithm in combination continuous data.

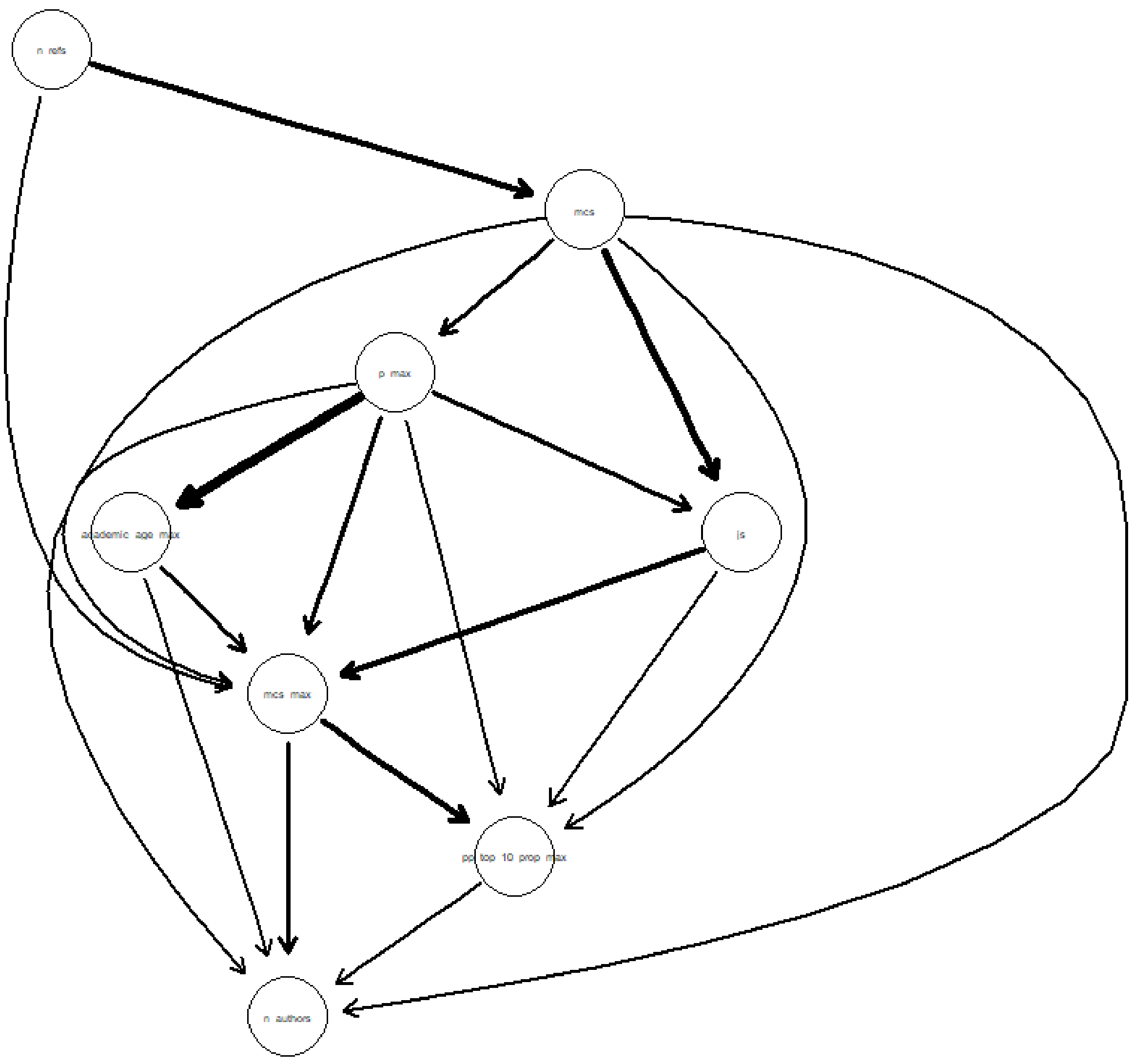


Fig. 22: Strength plot of structure learned using Tabu-search algorithm in combination continuous data.

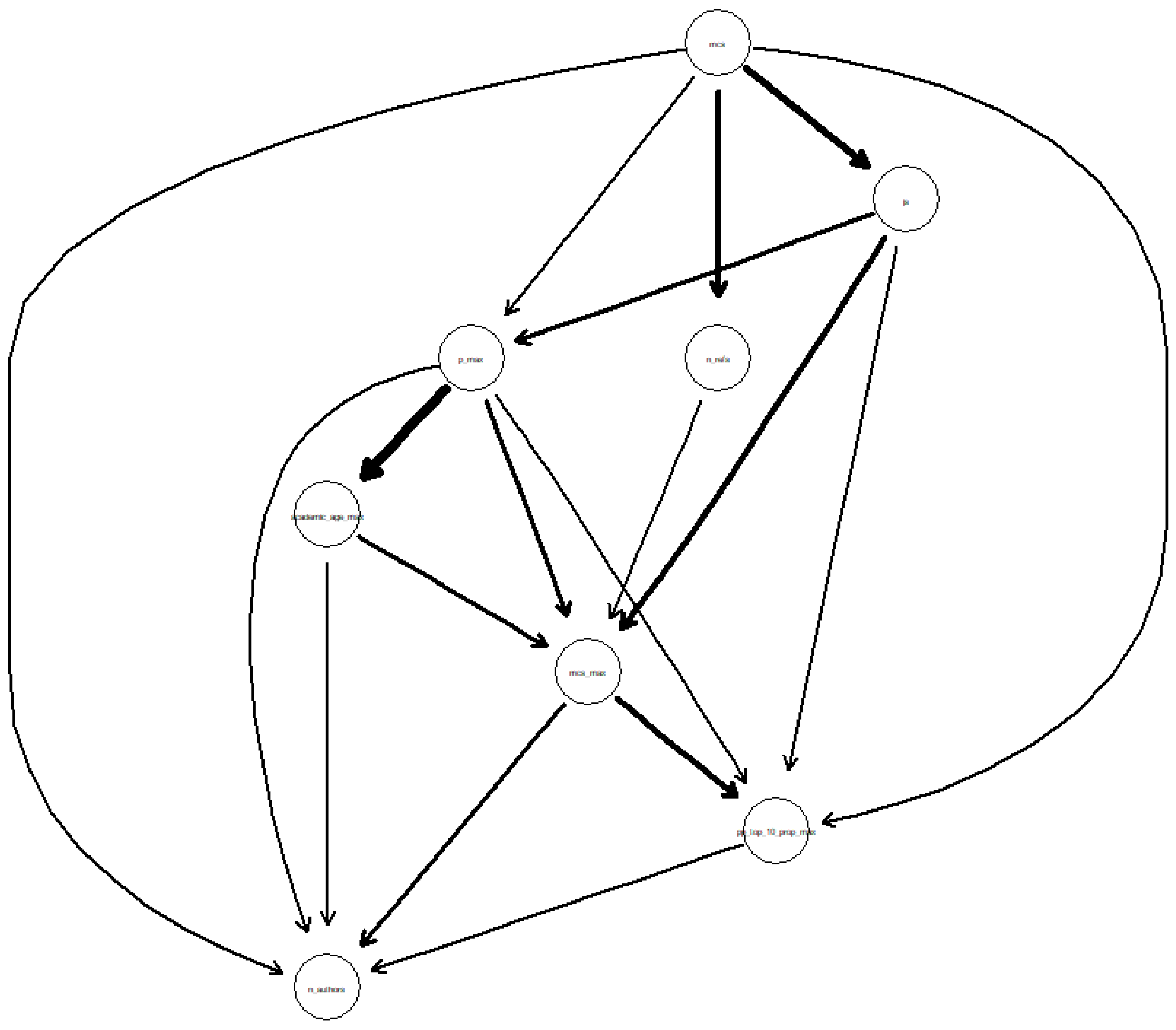


Fig. 23: Strength plot of structure learned using Max-Min Hill-Climb algorithm in combination continuous data.

4.2 Multinomial Bayesian Networks

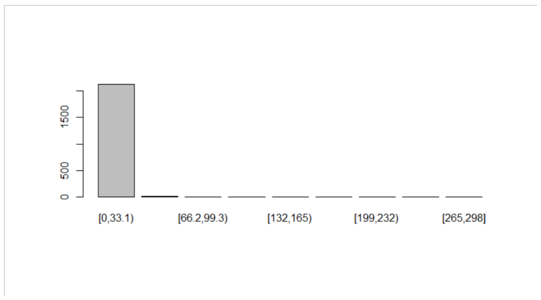
Since the variables were discretized to make use of the MBN, it becomes harder to compare the results from the MBN to the results of the other models that don't discretize the data. To compare the different MBNs themselves is already a small issue because of the different discretization methods.
Table 6 shows the percentage of correct predictions made by the different models obtained with various structure learning algorithms. In this context, 'correct' means that the BN predicted the correct interval the mcs would fall into. Since the data was discretized, a large part of the information was lost. Because of this loss of information, some of the structure learning algorithms could not construct complete graphs containing all variables. The ones that could are included in Table 6. These are two score-based algorithms, the HC-algorithm and Tabu-search, and one hybrid algorithmn the MMHC. This suggests that constraint-based algorithms are not well-suited for use on discretized data.
As seen in Table 6 there are large differences in the correct percentage of predictions between the various discretization techniques, but very small differences between the structure learning algorithms. The different discretization methods led to the large differences, as the percentages of correct predictions for each method are very close to each other or the same for the different structure learning algorithms. This is a natural consequence of the big difference in the structure of the data after discretization, seen in Figure 11, when using different techniques.
This brings into question whether the models are still useful after the discretization of the data, when looking at some of the higher quantiles.

Discretization	Structure Learning	Correct %
Equal interval	HC	99.07%
Equal interval	Tabu	99.07%
Equal interval	MMHC	99.07%
Equal frequency	HC	26.37%
Equal frequency	Tabu	26.61%
Equal frequency	MMHC	26.46%
Cluster	HC	45.24%
Cluster	Tabu	45.24%
Cluster	MMHC	45.24%

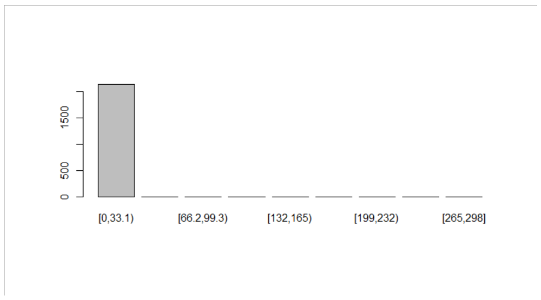
Tab. 6: Percentage of correct predictions MBN's

Discretization	Structure Learning	75%-Quantile	90%-quantile
Equal interval	HC	96.32%	89.90%
Equal interval	Tabu	96.32%	89.90%
Equal interval	MMHC	96.32%	89.90%
Equal frequency	HC	12.62%	10.90
Equal frequency	Tabu	9.37%	5.41%
Equal frequency	MMHC	12.53%	10.90%
Cluster	HC	0%	0%
Cluster	Tabu	0%	0%
Cluster	MMHC	0%	0%

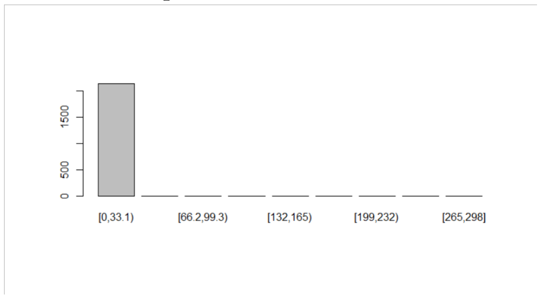
Tab. 7: Percentage of correct predictions made by MBNs above 75%-quantile and above 90%th-quantile of the mcs-variable



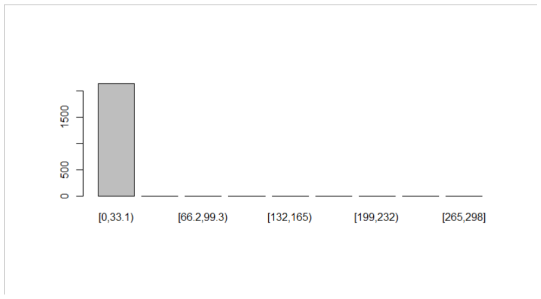
Equal Interval Test-set



HC MBN

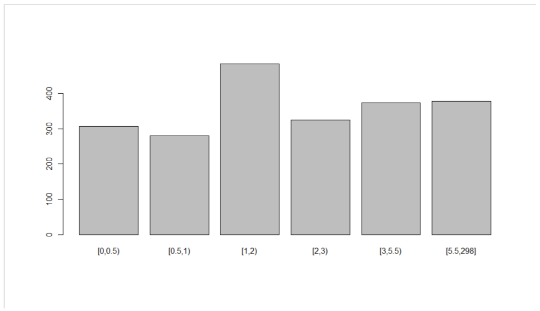


Tabu MBN

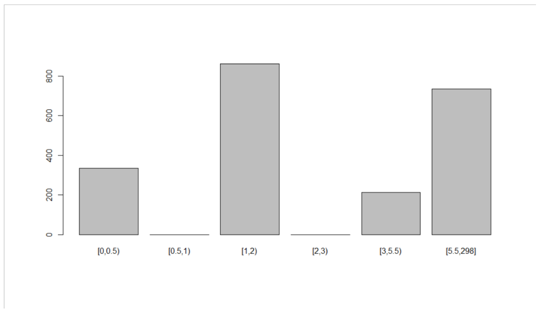


MMHC MBN

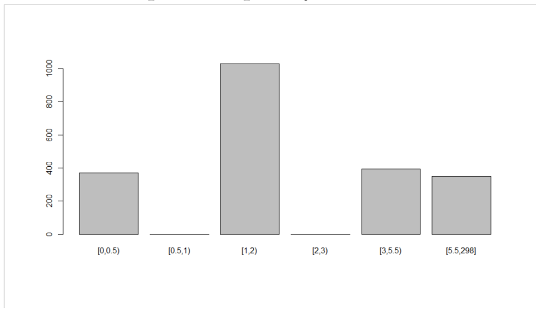
Fig. 24: Predictions of the Multinomial Bayesian Networks constructed using Equal Interval discretized data, compared to the mcs-variable of the test-set



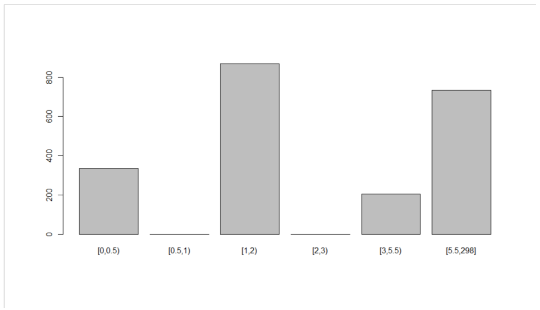
Equal Frequency Test-set



HC MBN



Tabu MBN



MMHC MBN

Fig. 25: Predictions of the Multinomial Bayesian Networks constructed using Equal Frequency discretized data, compared to the mcs-variable of the test-set

2

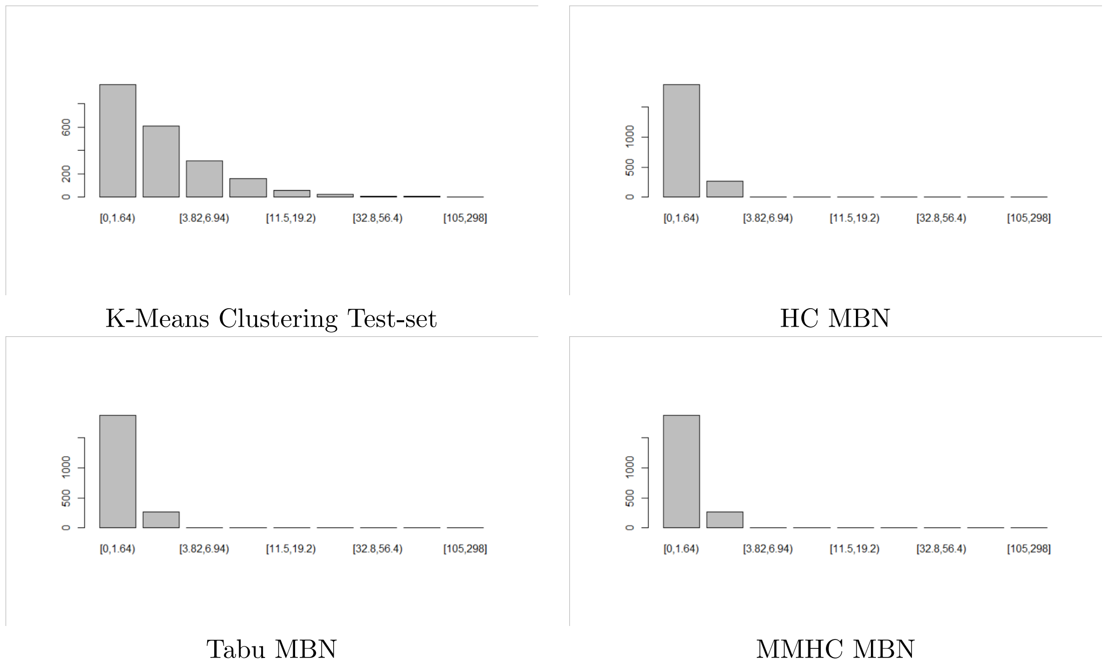


Fig. 26: Predictions of the Multinomial Bayesian Networks constructed using K-means Clustering discretized data, compared to the mcs-variable of the test-set

As seen in Table 6 and Table 7, the models based on the Equal Interval data produce the highest amount of correct predictions, but these are arguably the least valuable, since all of the instances are included in the first interval except for twenty. This can be seen in Figure 24. The networks based on this dataset were also not fully connected, because of this. The models based on the Equal Frequency and K-means Clustering data produce more realistic results, but the Cluster models do not perform well for the higher quantiles as they were unable to produce any correct predictions, shown in Table 7. Figure 26 shows that the bulk of predictions were made in the first interval. The models based on the Equal Frequency data might have the worst results on first glance, but did produce the most useful predictions for higher level quantiles. Interestingly enough the model based on the Tabu-learned structure performed not as well as the other two when it came to higher quantiles, even though it showed similar performance overall. As seen in Figure 25, it made a lot less predictions in the highest interval. A strange result is that non of the models predicted values in either the $[0.5, 1)$ interval or the $[2, 3)$ interval. These two intervals account for 28.14% of the instances and all predictions made for these instances were wrong.

4.3 Gaussian and Non-Parametric Bayesian Networks

After learning the different structures from the data, the parameters were fitted, resulting in the findings shown in Table 8. The first column contains the model type, whether it is Gaussian or Non-parametric, the second contains the structure learning algorithm, the third contains the

number of variables used for the prediction and the fourth contains the root mean squared error or RMSE. This is a way to measure the size of the error of a prediction. The RMSE using all the predictions made on the test-set.
When comparing the RMSE in Table 8, it can be seen that the performance of the different GBNs is very similar and the NPBN performs slightly better. There seems to be no large difference in prediction performance, between models based on different structure-learning algorithms. The Tabu-based BN seems to have a slight edge over the other two algorithms.
Weirdly enough the models all perform best when they are only presented with 4 variables It seems the last 3 variables, namely the n_authors, p_max and academic_age_max, do not improve the prediction.
When comparing the GBN's and NPBN's performance, looking only at the upper 75% and upper 90% in Table 9, it is clear that the NPBN outperforms the GBN. In this case the instances, where the mcs was higher than the 75th and 90th quantile, were inspected separately. The RMSE was calculated using these subsets, giving an insight into how big the errors are, when the real value of the mcs is known to be large. This is where the difference in prediction performance between the GBN and NPBN shows. Where the GBNs show RMSEs of around 445 and 640 above the 75%- and 90%-quantile respectively, the NPBN 'only' shows 106 and 155. While these are still large errors, they are around 25% the size of the GBN's errors. Again the model with the Tabu-learned structure performs slightly better than the other two, but this difference is marginal. The fact that the GBNs assume normal distributions for the conditional probability distributions of its variables and do not account for the large tails, which were present in the data, is displayed here. The NPBN does not have this issue as it does not make any assumptions on the distributions of the variables. This has caused the large difference in performance, when looking at the instances with extreme values. This difference would not have been detected if one would onlu have looked at the total RMSE's.

Model Type	Structure Learning	# Variables Used	RMSE
GBN	Tabu	2	12.6742048679046
GBN	Tabu	3	12.7427599880728
GBN	Tabu	4	10.7127814997178
GBN	Tabu	7	11.9985288034766
GBN	HC	2	12.8932488276976
GBN	HC	3	13.3270640907567
GBN	HC	4	11.1503183750799
GBN	HC	7	12.1658546627032
GBN	MMHC	2	13.0849528805077
GBN	MMHC	3	13.4076248009933
GBN	MMHC	4	11.1952561677952
GBN	MMHC	7	11.9608957560582
NPBN	HC	2	9.59464774873779
NPBN	HC	3	8.88682433967068
NPBN	HC	4	9.92035452871566
NPBN	HC	7	12.843069529702

Tab. 8: RMSE comparison Gaussian and Non-parametric Bayesian Networks for different amounts of variables presented to the model

Model Type	Structure Alg.	Total RMSE	75th Quant. RMSE	90th Quant. RMSE
GBN	Tabu	11.9985	442.4563	636.9295
GBN	HC	12.1658	446.4769	643.9369
GBN	MMHC	11.9608	447.1012	642.2934
NPBN	HC	12.8431	106.3981	155.8367

Tab. 9: RMSE comparison Gaussian and Non-parametric Bayesian Networks for the total set, 75%- and 90%-quantiles

Conclusion

The answer to the question whether the normality assumption can be ignored when using Gaussian Bayesian Networks for predictions in combination with non-normally distributed data is not simple. First of all, the comparison of the different models is complex. The performance of the MBNs is hard to set side by side with the performance of the GBN and NPBN. Even comparing the MBNs between themselves was not straightforward.
In the end it does seems that models that were constructed with data that was discretized with the Equal Interval-method gave the most correct predictions, but they were the least useful, as 96% of the data was included in one interval, rendering the predictions useless.
The models based on the K-means Clustered data performed surprisingly poor. When looking at the data after discretization, it looked the most similar to the original data, which should make the MBNs built using the data the easiest to compare with the GBN and NPBN. The models however were unable to predict any of the higher values correctly, rendering the models rather useless in this context. Since we are mostly interested in the performance in connection with the higher quantiles, the clustered data disappointed.
After all, the models that were built with Equal Frequency-discretized data proved to be the most useful. Even though the models do not make a lot of correct predictions, the predictions in the higher quantiles are the most useful when compared to the other MBNs. The large loss of information caused by discretization and the resulting difficulty of interpreting the predictions of the models accurately does however mean that the MBNs are less useful than the models that do not discretize the data.
The models based on the original continuous data might have a harder time making correct predictions, but the predictions are more useful in comparison to the models using discretized data. Analysing the results of this research, on first glance the performance of the GBN seems similar to that of the NPBN, when comparing prediction accuracy. Only by looking at the accuracy when predicting larger to extreme values or outliers it can be seen that the GBN falls short and its predictions are extremely far off, when compared to the NPBN. Even though the NPBN does result in large errors, its error is a lot smaller than that of the GBN.
All in all, the NPBN performs best when making predictions, but the extra knowledge, work and computing time are reasons to reassess whether it is necessary to use this more complicated model. If one is interested in looking at extreme values, the predictions. Otherwise the GBN performs similarly to the NPBN and could be used as well, even if the normality assumption is not satisfied. Since the GBN is so much easier to use, it would be a fitting choice in such cases.

Discussion

The findings of this research did not lead to a definitive answer to the question whether the normality assumption should be ignored when using Gaussian Bayesian Networks. Rather did it provide different answers that depend on the context and goal of the model. When one is interested in finding, for example, the conditional expected value of a variable, the GBN will be adequate. However, when one wants to know which variables are likely to cause the mcs to exceed a high conditional quantile estimate, say 90%, the NPBN should be used.
These results should encourage further research into neglecting model assumptions, as they show that in certain contexts, this is possible. What these contexts are, would be an interesting topic for future research.

It is important to note that the results of this research can not be extrapolated to different situations without reservations. The conclusions that were made are not necessarily applicable to new datasets or models. It should be encouraged to research these results further using new datasets and variables in order to reevaluate the conclusions stated in this research.
The methodical choices of this research were constrained by time. Because of this, certain techniques and methods have not been used.
For one, there are more single variable discretization methods than those that were used during the modelling process. These should not lead to completely different results as the loss of information by discretization should not increase or decrease much more with different methods. The methods used during this research are some of the most common ones and have shown very different results, so it could be assumed that they cover the subject sufficiently. Farnaz Nojavan A. (2017) [6] contains a deeper dive into the subject and introduces different methods.
The use of multivariate discretization was also not discussed, because the added value was not immediately clear and the comparison of discretization techniques was not the main focus of the research. There has been more research into the use of multivariable discretization in combination with BN's. Monti (2013)[10] describes a technique using a Bayesian scoring metric that is relative to the BN's structure. This technique could be used in later research.

References

[1] Stefano Beretta et al. "Learning the Structure of Bayesian Networks: A Quantitative Assessment of the Effect of Different Algorithmic Schemes". In: *Complexity* 2018 (2018), pp. 1–12. DOI: `10.1155/2018/1591878`.

[2] Harish Bhat and Nitesh Kumar. "On the Derivation of the Bayesian Information Criterion". In: (Jan. 2010).

[3] Marilena Furno Cristina Davino and Domenico Vistocco. *Quantile Regression: Theory and Applications*. Wiley, 2013. ISBN: 978-1-119-97528-1.

[4] Jarrod Dalton and Benjamin Nutter. *Working with HydeNet Objects*. July 6, 2020. URL: `https://cran.r-project.org/web/packages/HydeNet/vignettes/WorkingWithHydeNetObjects.html`.

[5] Reinhard Diestel. *Graph Theory*. Springer, 2017. ISBN: 978-3-662-53621-6.

[6] Craig A. Stow Farnaz Nojavan A. Song S. Qian. "Comparative analysis of discretization methods in Bayesian networks". In: *Environmental Modelling Software* 87 (2017), pp. 64–71.

[7] Maxime Gasse, Alex Aussem, and Haytham Elghazel. "An Experimental Comparison of Hybrid Algorithms for Bayesian Network Structure Learning". In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by Peter A. Flach, Tijl De Bie, and Nello Cristianini. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 58–73. ISBN: 978-3-642-33460-3.

[8] "Groei". In: (). URL: `https://www.hanze.nl/nld/onderzoek/kenniscentra/hanzehogeschool-centre-of-expertise-healthy-ageing/lectoraten/lectoraten/lahc/producten/producten/kindexpert/ontwikkeling-van-kinderen/groei/groei`.

[9] José Manuel Gutiérrez Marco Scutari Catharina Elisabeth Graafland. "Who learns better Bayesian network structures: Accuracy and speed of structure learning algorithms". In: *International Journal of Approximate Reasoning* 115 (2019), pp. 235–253. ISSN: 0888-613X. URL: `http://www.sciencedirect.com/science/article/pii/S0888613X19301434`.

[10] Stefano Monti and Gregory F. Cooper. *A Multivariate Discretization Method for Learning Bayesian Networks from Mixed Data*. 2013. arXiv: `1301.7403 [cs.AI]`.

[11] Marco Scutari and Jean-Baptiste Denis. *Bayesian Networks With Examples in R*. CRC Press, 2015. ISBN: 978-1-4822-2559-4.

[12] S. S. Shapiro and M. B. Wilk. "An Analysis of Variance Test for Normality (Complete Samples)". In: *Biometrika* 52.3/4 (1965), pp. 591–611. ISSN: 00063444. URL: `http://www.jstor.org/stable/2333709`.

[13] "Taalontwikkeling". In: (). URL: `https://www.hanze.nl/nld/onderzoek/kenniscentra/hanzehogeschool-centre-of-expertise-healthy-ageing/lectoraten/lectoraten/lahc/producten/producten/kindexpert/ontwikkeling-van-kinderen/taal/taal#:~:text=Het%20kind%20kent%20steeds%20meer,zinnen%20worden%20langer%20en%20complexer.`.