Modeling students' academic performance using Bayesian Networks

Student Research Societies

Kristóf Gál

Supervisors:

Roland Molontay
Máté Baranyi

2019

Contents

1 Introduction

Dropping out from higher education and delayed completion are associated with considerable personal and social costs. The dropout rate from STEM programs in Hungary is particularly high, one of the highest in the EU. Exploring the causes of early school leaving, supporting student learning success and finding intervention points have become well-studied research topics of great interest to both educational researchers and educational decision-makers.

The rich data stored in educational databases together with the emergence of efficient statistical and machine learning methods have enabled new approaches, which created a new line of research. This paper joins this research steam and contributes to the literature by exploring the relationship between pre-enrollment achievement measures and indicators of first-year university performance using Bayesian networks. Moreover, we answer the question of which relationships are the most significant and we also make some suggestions on how the explored connections can be used to assist higher education stakeholders.

In this paper, we review the related theory of Bayesian networks, how they are able to reveal dependencies between variables, and we introduce the most important construction algorithms. We demonstrate the applicability of Bayesian networks in the educational domain both based on recent literature and based on the academic data of students from Budapest University of Technology and Economics. We explore the relationship between pre-enrollment achievement measures and university performance outcomes in the first semester, furthermore, we point out the most significant relationships and outline possible intervention points and solutions.

The main contribution of this work is that we use Bayesian networks to model academic performance in the traditional educational setting. We give a detailed description of the applied algorithms and point out the limitations of this modeling approach. Moreover, to the best of our knowledge, this is the first study that investigates academic performance modeling using Bayesian networks in the Central European educational setting.

2 Review of related literature

The literature on educational data mining applications is continuously expanding. The first articles of the field date back to the late 1990s, and the First International Conference on Educational Data Mining [1] was held in 2008. There are several reviews of educational data mining that collect methods and results to describe advancements and present the current state of research [2, 3, 4].

One of the first applications of Bayesian networks in the educational domain was presented in 2004 by Michalis Xenos [5] who suggested an approach to model the progress of students of the Hellenic Open University that used distance education. He investigated the Informatics course facing high dropout rates with the aim of finding the causes of dropouts and to create a support system for educational administrators and tutors. Most of the factors, including for example tutor efforts and usage of textbooks, were broken down into three categories: poor, average and good; some into binary, for example exam results (success and failure) and a category was also introduced for unknown values. These category levels were used for the construction of the network. Using the model it became possible to predict the chance of a student dropping out based on historical performance on the course. Based on historical data, using the proposed modeling framework, tutors can also simulate how a student would react to a certain educational approach that can help the tutors to recommend the method that fits the student's learning style the most.

Another paper in web-based education using Bayesian networks is from García et al. about precision evaluation in student learning style detection [6]. The goal here was to identify the learning styles of students to give them the best available course material. The authors adapted an earlier study of student behavior by discarding factors that are not observable in an online environment and used the remaining variables to construct a Bayesian network. It was found that the Bayesian networks perform with varying precision depending on which variables were fixed as evidence for example perception, processing, and understanding.

In [7] Pardos et al. used Bayesian networks to predict the results of the tests produced by the Massachusetts Comprehensive Assessment System based on the test skill requirements that were broke down into elemental parts like inequality solving and they built up more complex skills like setting up and solving equations from those. The model was created based on domain knowledge and evaluation was carried out using Mean Absolute Difference (MD) of the predicted test results and the real test results. This way they achieved an error rate of 15% which was impressive compared to other results.

There are also works for student performance prediction in traditional education. In their work, Slim et al. [8] use Bayesian networks for reasoning about student progress

and performance with the goal of predicting future student progression based on earlier semesters. They used not only the students' results from previously completed courses but other factors as well, such as academic background and social factors. Their evaluation method used the Mean Squared Error (MSE) and they found that the proposed method can predict student grade point average with small error after observing the performance for one semester.

We have seen so far that Bayesian network based methods were used previously in educational data mining for the prediction of various attributes and that they achieve promising results. There are other techniques in the field. We list a few without attempting to be comprehensive.

In the article of Yukselturk et al. [9] the goal was to classify dropout students using 10 variables such as age and educational level which were collected through online questionnaires. They used four different approaches for classification: k nearest neighbor, decision tree, naive Bayes and neural network based methods. The 3 nearest neighbors algorithm achieved the highest, 87% detection sensitivity. The collected factors were also assessed in terms of feature importance and the three found to be the most important were online learning readiness, previous online experience, and self-efficacy of the online technology.

In her work, Kabakchieva [10] focuses on the implementation of data mining techniques to predict student performance based on pre-university and personal characteristics including the selection of features that have the strongest predictive power. They used decision trees, Bayes classifiers, k nearest neighbor methods, and rule learners.

There was a research [11] using data from the Budapest University of Technology and Economics with the aim of predicting dropout based on secondary school performance. In this work, both personal details (for example date of birth and address) and previous school performance (mature exam results and admission points) were used to identify students at risk of dropout. Many methods were set up and tested on the data including decision tree based, naive Bayes, k-nearest neighbors classifiers, linear models and deep learning. The model achieving the highest AUC (Area-Under-Curve) score was the Deep Learning model and Gradient Boosted Trees had the second-highest AUC. In a follow-up paper [12] the same authors have also investigated the interpretability of machine learning models for dropout prediction.

We also reviewed the field of Bayesian networks including books and articles to get a better view of the available methods [13, 14, 15].

In [16] the necessity of Bayesian networks for classification was tested through comparison to naive Bayes methods and other classifiers. The main characteristic of the Naive Bayes model is that it makes the assumption that all features are conditionally independent given the class labels while the Bayesian network represents a set of variables as nodes of a graph, modeling dependencies between the variables as edges. After

testing over twenty datasets it was found that there is no significant difference in performance. The paper also points out that the optimality measures used for Bayesian network construction are not the best for classifier applications and a suitable family of functions is introduced instead. With all this in mind, for non-classification problems like prediction and imputation Bayesian networks perform well. Also, in the real world, in real data, features are hardly ever independent and therefore the application of Naive Bayes classifiers can be misleading because of their strong assumption. The Bayesian networks can get a grasp on the dependencies in real data.

There are works regarding the theory of learning Bayesian networks from samples [17]. It was shown previously that for learning Bayesian networks from data there is a greedy search that identifies a perfect map of the generative distribution if the map is a Directed Acyclic Graph (DAG). This two-phased greedy search is implemented and was shown to lead to good solutions when applied to finite sample sizes.

In their paper [18] Perković et al. investigate completed partially directed acyclic graphs, develop terminology and methods and demonstrate the gain in identifiability of causal effects as the available domain knowledge increases. Bayesian network structure building methods were also developed for high dimensional DAGs [19], in particular, the PC-algorithm. These algorithms that search through the space of DAGs were developed over the years [20] and most statistical and machine learning repositories that support Bayesian networks have their own implementations of structure and parameter learning algorithms.

3 Theoretical background

In this section we first overview some probability and graph theory fundamentals, then define Bayesian networks, and present learning algorithms followed by the idea of inference. This section heavily relies on books [21, 22, 23].

3.1 Probability and graph theory

Definition 1. Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. A random variable is a measurable function $X : \Omega \to \mathbb{R}$.

The realization of a random variable is an event, thus definitions of this section are directly applicable to random variables.

Definition 2. Let A and B be two events from the σ-field of the probability space $(\Omega, \mathcal{F}, \mathcal{P})$ with $\mathbb{P}(B) > 0$. Then the conditional probability of A given B is

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)},$$

the probability of the intersection of the events divided by the probability of B.

Theorem 3 (**Bayes' theorem**). *Let $(\Omega, \mathcal{F}, \mathcal{P})$ be a probability space. Then $\forall A, B \in \mathcal{F}$, $\mathbb{P}(B) > 0$:*

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(B|A)\mathbb{P}(A)}{\mathbb{P}(B)}.$$

Definition 4. Let $(\Omega, \mathcal{F}, \mathcal{P})$ be a probability space and $A, B \in \mathcal{F}$ be two events with positive probabilities: $\mathbb{P}(A) > 0$ and $\mathbb{P}(B) > 0$. Then A and B are independent (denoted as $A \perp B$) if

$$\mathbb{P}(A \cap B) = \mathbb{P}(A) \cdot \mathbb{P}(B),$$

their joint probability is the product of their probabilities. Equivalent forms:

$$\mathbb{P}(A \cap B) = \mathbb{P}(A) \cdot \mathbb{P}(B) \iff \mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)} = \mathbb{P}(A),$$

$$\mathbb{P}(A \cap B) = \mathbb{P}(A) \cdot \mathbb{P}(B) \iff \mathbb{P}(B|A) = \frac{\mathbb{P}(B \cap A)}{\mathbb{P}(A)} = \mathbb{P}(B),$$

meaning that the occurrence of B does not affect the occurrence of A and neither does the occurrence of A affect the occurrence of B.

Although independence is a very useful and interesting notion, a property that we may encounter more often regarding real-world scenarios is conditional independence.

Definition 5. Let $(\Omega, \mathcal{F}, \mathcal{P})$ be a probability space and $A, B, C \in \mathcal{F}$ three events with positive probabilities: $\mathbb{P}(A) > 0$, $\mathbb{P}(B) > 0$ and $\mathbb{P}(C) > 0$. Then A and B are conditionally independent given evidence C (denoted by $A \perp B|C$) if and only if

$$\mathbb{P}(A \cap B|C) = \mathbb{P}(A|C) \cdot \mathbb{P}(B|C).$$

Corollary 5.1. *If A and B are conditionally independent given evidence C then using the definition of conditional probability we have*

$$\mathbb{P}(A|B \cap C) = \frac{\mathbb{P}(A \cap (B \cap C))}{\mathbb{P}(B \cap C)} = \frac{\mathbb{P}(A \cap B|C) \cdot \mathbb{P}(C)}{\mathbb{P}(B|C) \cdot \mathbb{P}(C)} = \frac{\mathbb{P}(A|C) \cdot \mathbb{P}(B|C)}{\mathbb{P}(B|C)} = \mathbb{P}(A|C),$$

and similarly

$$\mathbb{P}(B|A \cap C) = \mathbb{P}(B|C).$$

The events A and B are conditionally independent given evidence C if and only if occurring A provides no additional information on the probability of the occurrence of B, given the occurrence of evidence C, and similarly, the knowledge of whether B occurs or not provides no additional information about the probability of the occurrence of A. After defining conditional independence the following statements can be made:

Corollary 5.2. *Let $(\Omega, \mathcal{F}, \mathcal{P})$ be a probability space and $A, B, C, D \in \mathcal{F}$ then:*

1. $(A \perp B) \Longrightarrow (B \perp A)$ *(symmetry property),*
2. $(A \perp B)|C \wedge (A \perp C) \Longrightarrow (A \perp B, C)$ *(contraction property),*
3. $(A \perp B, C) \Longrightarrow (A \perp B)|C \wedge (A \perp C)|B$ *(weak union property),*
4. $(A \perp B, C) \Longrightarrow (A \perp B) \wedge (A \perp C)$ *(decomposition property) and*
5. *if the joint density function of all variables is positive and continuous with respect to the product measure then*

$$(A \perp B)|C, D \wedge (A \perp D)|B, C \Longrightarrow (A \perp B, D)|C \text{ (intersection property).}$$

Definition 6. A graph is an ordered $\mathcal{G} = (V, E)$ pair, where V is a finite set (of vertices or nodes) and $E \subseteq V^2$ (the set of edges), $E \subseteq \big\{\{x, y\}|x \in V, y \in V\big\}$. The graph $\mathcal{G} = (V, E)$ is simple if $\forall\{x, y\} \in E : x \neq y$, in other words it contains no loops, edges that connect a vertex with itself.

Definition 7. Let $\mathcal{G} = (V, E)$ be a graph. A path from $x \in V$ to $y \in V$ is a sequence of nodes $\{v_i\}_{i=0,\ldots,n}$, such that $\forall i = 0, \ldots, n-1 : \{v_i, v_{i+1}\} \in E$ and $v_0 = x$ and $v_n = y$.

Definition 8. A directed graph is an ordered $\mathcal{G} = (V, E)$ pair where V is the finite set of nodes and $E = \{(x, y) \text{ ordered pairs} \mid x, y \in V\}$, there is a directed edge (arc) from x to y if $(x, y) \in E$. Directed edges are represented by arrows: we denote $(x, y) \in E$ by $x \to y$.

Definition 9. Let $\mathcal{G} = (V, E)$ be a directed graph and $e = (x, y) \in E$ then x is a parent of y and y is a child of x. The parents of node x in $\mathcal{G}$ is a set denoted by $Par_{\mathcal{G}}(x)$ and the children of node x in $\mathcal{G}$ is a set denoted by $Chi_{\mathcal{G}}(x)$.

Definition 10. Let $\mathcal{G} = (V, E)$ be a directed graph. There is a directed path from $x \in V$ to $y \in V$ if there are nodes $\{v_i\}_{i=0,\ldots,n}$, $\forall i = 0, \ldots, n-1 : (v_i, v_{i+1}) \in E$ and $v_0 = x$ and $v_n = y$.

Definition 11. Let $\mathcal{G} = (V, E)$ be a directed graph and $x, y \in V$. If there is a directed path from x to y then x is an ancestor of y and y is a descendant of x. The parents of a node are also ancestors and children of a node are also descendants. The descendants of node x in $\mathcal{G}$ is a set denoted by $Des_{\mathcal{G}}(x)$.

Definition 12. Let $\mathcal{G} = (V, E)$ be a directed graph and $x \in V$. If x has no parents then x is called a root.

Definition 13. Let $\mathcal{G} = (V, E)$ be a directed graph and $x \in V$. A directed path that goes from x to x is called a cycle.

Definition 14. A directed graph that contains no cycles is called a directed acyclic graph (DAG).

Remark. We only mentioned undirected and directed graphs. It is possible to define and use partially directed graphs but in this work we focus on directed ones.

3.2 Bayesian networks

A probabilistic graphical model is a probabilistic model for which a graph expresses the conditional dependence structure between random variables, for example, Markov random fields and Bayesian networks. Bayesian networks represent a set of random variables and their conditional dependencies. Their structure is a DAG and each node has its own probability table.

Definition 15. A Bayesian network $(\mathcal{G}, \mathcal{P})$ is a representation of a probability distribution over a vector of random variables $\mathbf{X} = \{X_1, \ldots, X_n\}$, where $\mathcal{G}$ is a DAG, the structure of the Bayesian network, and $\mathcal{P}$ is a set of conditional probability distributions. The nodes of $\mathcal{G}$ represent the random variables and the edges represent

conditional dependence statements. Each node has its own conditional probability distributions for the possible value combination of its parents:

$$\mathbb{P}(X_i|Par_{\mathcal{G}}(X_i)),$$

where $Par_{\mathcal{G}}(X)$ is the set of the parents of the node corresponding to X in $\mathcal{G}$.

Based on its conditional independence assumptions, the Bayesian network defines the joint distribution as

$$\mathbb{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbb{P}(X_i|Par_{\mathcal{G}}(X_i)).$$

This is a well-defined distribution: $\mathbb{P}(X_1, \ldots, X_n)$ is a product of conditional probability distributions, thus non-negative and $\sum \mathcal{P} = 1$.

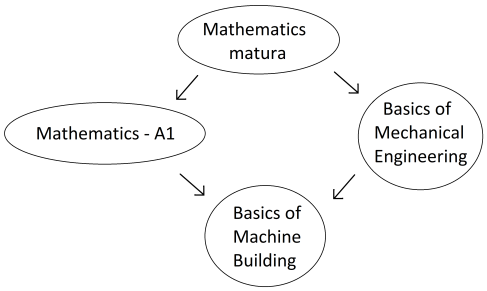


Figure 1: Example of a simple Bayesian network

In Figure 1, a simple discrete Bayesian network structure is shown. It is created based on our own assumptions only, so its goodness is not tested. It models a situation with four variables: mathematics matura (denoted by M) influences the result of two first-semester courses, Mathematics A1 (A) and Basics of Mechanical Engineering (E), and further, both influence the result of the second-semester course Basics of Machine Building (B). The conditional probabilities in each node only depend on the given node's parents.

The distribution of this Bayesian network is a product of the factors conditional

distributions:

$$\begin{aligned}
&\sum_{M,A,E,B} \mathbb{P}(M,A,E,B) = \\
&\quad = \sum_{M,A,E,B} \mathbb{P}(M) \cdot \mathbb{P}(A|M) \cdot \mathbb{P}(E|M) \cdot \mathbb{P}(B|A,E) \\
&\quad = \sum_{M,A,E} \mathbb{P}(M) \cdot \mathbb{P}(A|M) \cdot \mathbb{P}(E|M) \cdot \sum_{B} \mathbb{P}(B|A,E) \\
&\quad = \sum_{M,A} \mathbb{P}(M) \cdot \mathbb{P}(A|M) \cdot \sum_{E} \mathbb{P}(E|M) \\
&\quad = \sum_{M} \mathbb{P}(M) \cdot \sum_{A} \mathbb{P}(A|M) \\
&\quad = 1
\end{aligned}$$

Here we used multiple times that $\sum_X \mathbb{P}(X|Par(X)) = 1$.

3.2.1 "Bayes-ball"

To gain a better understanding of the flow of probabilistic influence in Bayesian networks we present the rules of "Bayes-ball" [24]. Let $(\mathcal{G}, \mathcal{P})$ be a Bayesian network, $\mathcal{G} = (V, E)$ and $X, Y \in V$ random variables from the Bayesian network.

Definition 16. Let $(\mathcal{G}, \mathcal{P})$ be a Bayesian network and $\mathcal{X}$ be an undirected path (trail) between X_1 and X_m, $\mathcal{X} = (\{X_1, X_2\}, \{X_2, X_3\}, \ldots, \{X_{m-1}, X_m\})$. This trail in $\mathcal{G}$ is an active trail for evidence $Z \subseteq \{X_1, \ldots, X_m\}$ if every consecutive (X_{i-1}, X_i, X_{i+1}) triplet of variables on the trail is active. The active triplets are:

1. $X_{i-1} \to X_i \to X_{i+1}$ and $X_{i-1} \leftarrow X_i \leftarrow X_{i+1}$ if $X_i \notin Z$ (X_i is not observed)
2. $X_{i-1} \leftarrow X_i \to X_{i+1}$ if $X_i \notin Z$ (X_i is not observed)
3. $X_{i-1} \to X_i \leftarrow X_{i+1}$ if $X_i \in Z$ or $Des(X_i) \cap Z \neq \varnothing$ (X_i or any of it descendants is observed)

In Figure 2, one can see the rules for active triplets but only the middle node of each triplet is shown. The pair of triplets in each row corresponds to the row with the same number in Definition 16. White nodes are unobserved, gray nodes are observed, black arrows represent the edges of the triplet. The straight dark red arrows show where the "Bayes-ball" of probabilistic influence can roll through and the curved ones show where it bounces back.

Remark. Let $(\mathcal{G}, \mathcal{P})$ be a Bayesian network, $\mathcal{G} = (V, E)$ and $X, Y \in V$ random variables from the network. If $X \to Y$ or $X \leftarrow Y$ then X can influence Y, and vice versa.

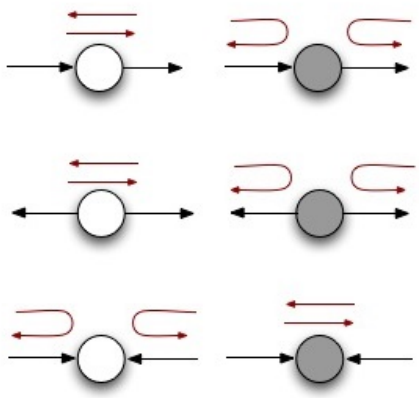


Figure 2: The rules of "Bayes-ball"

Definition 17. Let $(\mathcal{G}, \mathcal{P})$ be a Bayesian network, $\mathcal{G} = (V, E)$ and $A, B, C \subset V$ three subsets of nodes from $\mathcal{G}$. A and B are directional-separated (d-separated) in $\mathcal{G}$ given evidence C, if there are no *active trails* between A and B with evidence C. It is denoted by $DS_{\mathcal{G}}(A, B|C)$. A and B are conditionally independent given C if A and B are d-separated given evidence C.

Returning to Figure 1, we can see that M and B are conditionally independent if we know A and E. They are dependent if we do not A or E because in that case M→A→B triplet or the M→E→B triplet becomes active and creates an active trail between M and B.

Definition 18. Let $I(\mathcal{G}) = \{(A \perp B|C) : DS_{\mathcal{G}}(A, B|C)\}$ be the set of conditional independencies corresponding to d-separation.

Definition 19. Let $\mathcal{G}$ be a directed graph with nodes $\mathbf{X} = (X_1, \ldots, X_n)$ and $\mathcal{P}$ a set of conditional probability distributions of $\mathbf{X}$. $\mathcal{P}$ factorizes over $\mathcal{G}$ if the chain rule holds for $\mathcal{P}$:

$$\mathbb{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbb{P}(X_i|Par_{\mathcal{G}}(X_i)),$$

where $Par_{\mathcal{G}}(X)$ is the set of parents of X in $\mathcal{G}$.

Definition 20. Let $\mathcal{G}$ be a directed graph with a set of independencies $I(\mathcal{G})$, $A, B, C \subset V$ and $\mathcal{P}$ be a distribution over A. Let $I(\mathcal{P})$ be the set of independencies that hold in $\mathcal{P}$ in the form $(A \perp B|C)$. $\mathcal{G}$ is an independency map for $I(\mathcal{P})$ if $I(\mathcal{G}) \subset I(\mathcal{P})$.

Theorem 21. *If $\mathcal{P}$ factorizes over $\mathcal{G}$ then $\mathcal{G}$ is an independency map for $\mathcal{P}$.*

Theorem 22. *Let $\mathcal{G}$ be a Bayesian network structure for a $\boldsymbol{X}$ set of random variables and $\mathcal{P}$ be a joint distribution of $\boldsymbol{X}$. If $\mathcal{G}$ is an independency map for $\mathcal{P}$ then $\mathcal{P}$ factorizes over $\mathcal{G}$.*

Corollary 22.1. *The previous two theorems together state that $\mathcal{P}$ factorizes over $\mathcal{G}$ if and only if $I(\mathcal{G}) \subset I(\mathcal{P})$.*

This theorem explains why the distribution of Bayesian networks factorize like stated before. Returning to the example of Figure 1:

$$\mathbb{P}(M, A, E, B) = \mathbb{P}(M) \cdot \mathbb{P}(A|M) \cdot \mathbb{P}(E|M) \cdot \mathbb{P}(B|A, E)$$

$$\begin{aligned}\mathbb{P}(M) &= \sum_{A,E,B} \mathbb{P}(M) \cdot \mathbb{P}(A|M) \cdot \mathbb{P}(E|M) \cdot \mathbb{P}(B|A, E) \\ &= \mathbb{P}(M) \cdot \sum_{A} \mathbb{P}(A|M) \cdot \sum_{E} \mathbb{P}(E|M) \cdot \sum_{B} \mathbb{P}(B|A, E) \\ &= \mathbb{P}(M)\end{aligned}$$

3.3 Learning

The learning of Bayesian networks consists of two steps: structure learning and parameter learning. In the simplest case, the Bayesian network is specified by an expert and no learning is necessary. In most of the cases, the conditional distributions and/or the structure of the Bayesian network are not given therefore have to be estimated from data.

Structure learning, the construction of the underlying graph from data, is hard: inferring causality is difficult and exponential time is required to search through all possible DAGs to find the optimal one, see Section 3.3.1.

To completely represent the probability space in possession of the graphical structure, it is necessary to specify the conditional probability distribution for each node conditioned on its parents. This is called parameter learning, or in other words, fitting. Parameter learning can be done by summing up counts through the data, see Section 3.3.4.

3.3.1 Structure learning in general

Structure learning methods can be categorized for example in the following way:

Search and score algorithms: these algorithms have two parts, a search strategy to choose DAGs for consideration and an objective function that evaluates (scores) the DAGs in some way. For example, a typical objective function attempts to balance the probability of the data given the model, and the complexity of the model. The naive version of this algorithm is super-exponential with the number of variables. The repeated calculations can be bypassed with dynamic programming, and special algorithms can be used for smarter searching over the space of DAGs, in order to reduce the computational burden.

Constraint learning: these methods usually calculate some measure of (partial) correlation to identify an undirected underlying graph structure and then direct these edges until a DAG is reached. The undirected structure is commonly built by iterating over possible triplets to identify conditional independencies. These methods are faster than search and score algorithms but do not have simple probabilistic interpretations.

Approximate algorithms: these attempt to balance the interpretability of search and score and the short execution time of constraint methods. Several heuristics were developed to build good structures in an acceptable amount of time, for example the Chow–Liu tree building algorithm.

3.3.2 Bayesian network based on the Chow–Liu tree

The Chow–Liu algorithm provides a simple and efficient way to find the best undirected tree structure approximation that maximizes the likelihood of the data [15]. In this case we want to find the tree structure that maximizes the likelihood over all fully observed tree structures.

The algorithm goes as follow:

1. compute the mutual information for every possible edge based on the sample;
2. order the values in decreasing order;
3. start with an empty graph and following the ordered values, add the corresponding edge to the graph unless the addition creates a cycle
4. stop if no more edges can be added (in other words if $n - 1$ edges were added, where n is the number of nodes)

The mutual information between two variables X and Y over the space $\mathcal{X} \times \mathcal{Y}$ is:

$$I(X,Y) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p_{X,Y}(x,y) \log \left(\frac{p_{X,Y}(x,y)}{p_X(x) p_Y(y)} \right),$$

where $p_{X,Y}$ is the joint probability function, p_X and p_Y are the probability functions of X and Y. The probability function values can be estimated from the data.

In the case when there are no equal mutual information values this returns a unique best solution. If there are equal values then there are multiple best solutions, each giving the same likelihood function value.

Again, the resulting graph will be undirected. To direct the edges a root node is selected and edges are directed away from the root. Iteratively, for nodes that are reachable on directed edges from the root, one directs all undirected edges away from the reached nodes. This trivially results in a DAG.

3.3.3 Structure learning as a shortest path problem

The Bayesian network structure learning problem can be transformed into a shortest path problem by utilizing another special graph structure, called lattice or order graph over the variables. It is defined in the following way, in context of the Bayesian network structure learning problem on variables $\{X_1, \ldots, X_n\}$:

1. let the root, the starting node for the shortest path algorithm, be the empty set (layer 0);

2. let the leaf, the ending node for the shortest path algorithm, be the set of all variables in the network $\{X_1, \ldots, X_n\}$ (n^{th} layer);

3. add each possible variable subset to the graph as a node ordered into layers, such that all subsets of a layer has the same number of elements (the i^{th} layer contains all i-element subsets of $\{X_1, \ldots, X_n\}$, and no other subsets);

4. connect the layers in a way that from each node S_1 of the i^{th} layer directed edges go to each node S_2 of the $(i+1)^{\text{th}}$, with $S_1 \subset S_2$ ($|S_1| + 1 = |S_2|$);

5. the weights of the edges are given by a function (some goodness of fit measure of the structure).

In the graph defined above a path from the root to the leaf represents a topological sort of the variables. The path with the lowest total weight corresponds to the optimal topological sort and Bayesian network. The topological order of the variables describes the network in the way that each variable may have the nodes preceding it in the ordering as its parents.

One of the "goodness" measures for learning Bayesian networks is Minimal Description Length (MDL). The MDL is a two-component score that estimates the number of bits required to represent a model and the data given that model. In case of Bayesian networks, this is the number of bits required for the model representation (graph and conditional probability tables) while the bits required for data approximation is inversely proportional to the probability of the data given the model.

For a Bayesian network B and learning data D, the MDL score is

$$MDL(B, D) = \gamma MDL(B) + MDL(D|B),$$

where MDL(B) is the bits required to represent the network, MDL($D|B$) is bits required to represent the data given the model and γ is a weighting parameter. The value of γ can be used to control the complexity of the result, the lower the value the more complex the network will be.

After the transformation of the problem, a shortest path algorithm can be used to find the optimal network. To solve this shortest past problem for the MDL score, edge weights of the lattice graph should be defined as

$$\begin{aligned} w(S_1, S_2) &= BestMDL(X, S_1) \\ &= \underset{Par(X) \subseteq S_1}{\operatorname{argmin}} MDL(B_{X \cup Par(X)}, D), \end{aligned}$$

which is the weight between S_1 and S_2, $X \in S_2 \setminus S_1$ and $BestMDL(X, S_1)$ is the best score over all possible parents for X in S_1 when we attach X to the best possible Bayesian network over S_1.

The naive way to find the best network is to evaluate all possible topological orderings. For example, the A* algorithm can be used to only partially evaluate the graph and still get an optimal network in the end. The description of the application of A* search for Bayesian network structure learning can be found in [25].

A greedy search based on the same lattice graph is also possible, but it will not result in an optimal graph in every case. It is greedy in the sense that it will not use the heuristics, as A* search does, to determine the weights that will be discovered later, it only selects the next edge to be the one with the lowest weight.

3.3.4 Parameter learning

Parameter learning from data given the structure is fairly simple: for each node we consider the data regarding the node and its parents only. In the discrete case, the maximum likelihood estimate is the number of occurrences of each set of possible value combinations divided by the number of samples in the data: let D be the data, X be a node of the network, $Par(X)$ be the set of parents of X and $\mathbf{z}$ be a realization of $Par(X)$, then the estimation of the conditional probability from the data is

$$\mathbb{P}\left(X = x | Par(X) = \mathbf{z}\right) \approx \frac{|\{X = x, Par(X) = \mathbf{z} | D\}|}{|\{Par(X) = \mathbf{z} | D\}|}.$$

For example, considering a dataset with two binary variables X and Y, and n number of records, we can find $\mathbb{P}(X = 0)$, $\mathbb{P}(X = 1)$, $\mathbb{P}(Y = 0)$ and $\mathbb{P}(Y = 1)$. Let X be a parent of Y: the necessary estimations of the probabilities $\mathbb{P}(Y = 0|X = 0)$, $\mathbb{P}(Y = 0|X = 1)$, $\mathbb{P}(Y = 1|X = 0)$, and $\mathbb{P}(Y = 1|X = 1)$ can be done from the data:

$$\mathbb{P}(Y = 0|X = 0) = \frac{\mathbb{P}(Y = 0, X = 0)}{\mathbb{P}(X = 0)} \approx \frac{|\{Y = 0, X = 0\}|}{|\{X = 0\}|}.$$

3.4 Inference

One of the most powerful abilities of Bayesian networks is that they can perform inference efficiently: given any set of observed variables Z from the network (including the absence of evidence) a Bayesian network can make predictions for all variables X from the network quicker than the plain calculation of $\mathbb{P}(X|Z)$, by using the graph of the network $\mathcal{G}$ and calculating $\mathbb{P}(X|Z \cap \mathcal{G})$. In the rest of this section the loopy belief propagation algorithm is described.

3.4.1 Loopy belief propagation algorithm

Belief propagation is a dynamic programming approach to do conditional probability queries in a graphical model. The task is to infer $\mathbb{P}(X|\mathbf{Z} \cap \mathcal{G})$, the conditional probability of each variable X on the graph $\mathcal{G}$ given an observed subset $\mathbf{Z}$ of nodes as evidence. The belief propagation algorithm has polynomial complexity in the number of nodes, while exact probabilistic inference, in general, has been proven to be NP-hard [26].

The idea of the belief propagation algorithm is to have each node determine a distribution by listening to its neighbors. Evidence enters the network at the observed nodes and propagates through it: adjacent nodes exchange messages in every iteration telling each other how to update beliefs based on prior beliefs, conditional distributions, and evidence.

The belief $\mathcal{B}_X(x)$ can be used to estimate the probability of $X = x$ given evidence $\mathbf{Z} = \mathbf{z}$ for all variable X:

$$\mathcal{B}_X(x) \approx \mathbb{P}(X = x|\mathbf{Z} = \mathbf{z})$$

Detailed descriptions of the algorithm can be found in [21, 27, 28]. Here we give a brief summary.

Let X be a node with children $\{Y_1, \ldots, Y_n\}$ and parents $\{U_1, \ldots, U_m\}$. Let $\lambda_{Y_j}(x)$ be the messages coming from the children of X to X, and $\pi_X(u_k)$ the messages coming from the parents of X to X. It is possible for X to send a message to itself, denoted by $\lambda_X(x)$. The belief can be calculated in the following way:

$$\mathcal{B}_X(x) = \alpha\lambda(x)\pi(x),$$

where α is the normalizing constant,

$$\lambda^{(t)}(x) = \lambda_X(x) \prod_j \lambda_{Y_j}^{(t)}(x),$$
$$\pi^{(t)}(x) = \sum_{\mathbf{u}} \mathbb{P}(X = x|\mathbf{U} = \mathbf{u}) \prod_k \pi_X^{(t)}(u_k).$$

The message sent by X to its parent U_i at the $(t+1)^{\text{th}}$ iteration is

$$\lambda_X^{(t+1)}(u_i) = \alpha \sum_x \lambda^{(t)}(x) \sum_{u_k : k \neq i} \mathbb{P}(x|u) \prod_{k \neq i} \pi_X^{(t)}(u_k),$$

and the message sent by X to its child Y_j at the $(t+1)^{\text{th}}$ iteration is

$$\pi_{Y_j}^{(t+1)}(x) = \alpha \pi^{(t)}(x) \lambda_X(x) \prod_{k \neq j} \lambda_{Y_j}^{(t)}(x).$$

The initialization of messages can be done randomly with the following boundary conditions:

1. for all evidence nodes $z_i \in \mathbf{Z}$, $\lambda(x_i) = 1$ if $x_i = z_i$, otherwise 0 and $\pi(x_i) = 1$ if $x_i = z_i$, otherwise 0;
2. for nodes without parents, $\pi(x_i) = \mathbb{P}(x_i)$, the prior probabilities;
3. for nodes without children, $\lambda(x_i) = 1$.

The messages are sent in the network simultaneously until a stable belief state is reached, if ever.

4 Modeling academic performance

Based on pre-enrollment achievement measures and demographic factors, we aim to predict how students perform in their early university studies. We build Bayesian networks for data that is available at the time of enrollment and on results of mandatory courses from the first two semesters of the mechanical engineering bachelor's program at the Budapest University of Technology and Economics. In this section, first we provide some background on the Hungarian education and university admission system, then we describe our data preparation steps and the modeling framework.

4.1 University admission system in Hungary

In Hungary, the 4 years of secondary education are followed by higher education. Schools use a Likert scale where 1 (unsatisfactory) represents fail and 5 (excellent) represents the best performance. At the end of their high school studies, students take an exit exam (matura), called 'érettségi vizsga' in Hungarian. This exam includes at least five subjects: Mathematics, History, Hungarian Language and Literature, one Foreign Language and one additional subject of the students' choice that they studied for at least two years. Students may decide to take exams in additional subjects and to take exams at advanced level.

Majority of higher education admission relies on secondary school performance and maturity exam results. The predictive power of these was investigated in [11], along with a description of the admission system. Students applying to universities can gather their admission score (AS) from several sources that can be summarized into three factors:

1. **study points** (SP) include secondary school grades and maturity exam results, at most two hundred points
   (a) the sum of grades of the core subjects and one subject of the students' choice (five total) from the last two academic years of study for each subject, at most a hundred points
   (b) the average result of the five mandatory maturity exams in percentage, also a hundred points at most
2. **maturity exam points** (MP)
   (a) sum of the results of the two maturity exams defined by the selected university program, in percentage, at most two hundred points
3. **additional points** (AP) include points for additional achievements and equal opportunity points, at most a hundred points (eighty before 2012)

(a) fifty points for each advanced level maturity exam that is used in calculating the maturity exam points (forty points before 2012)

(b) twenty-eight points for a B2 level and forty points for a C1 level foreign language certificates

(c) points for academic, art or sport competition placements

(d) forty points for equal opportunities

(e) points for higher-level vocational training based on results

The admission score is calculated as $AS = max\{SP + MP, 2 \cdot MP\} + AP$, thus the maximum is 500.

Universities define their own minimal admission score for each of their programs and enroll students achieving a greater score than the program's limit.

Demographic factors and the factors contributing to the admission score may help to predict how students perform in their early academic studies thus we use these values in our work. We build Bayesian networks for data that is available at enrollment and subsets of the subjects from the early semesters of the Mechanical Engineering bachelors program at the Budapest University of Technology and Economics.

4.2 Data preparation

The data provided by the Central Academic Office of the Budapest University of Technology and Economics was extracted from the Neptun educational administration system. We received anonymized data of pre-enrollment achievement measures together with course-level university performance indicators for over 30000 students admitted between 2010 and 2017 at Budapest University of Technology and Economics. We had to deal with a large amount of missing data since not every attribute was available for all students.

For the data preparation process we used the Python Programming Language (version 3.7.4) and mostly the Pandas package [29] (version 0.24.2).

First, we had to clean, transform and filter the data to fit our needs. The data contained multiple records for many students because of re-enrolled students. Re-enrollment means that a student dropped out but was accepted again for the same program. We considered the first available record when it was necessary because our goal is to predict the first grade the student will be available to achieve in the subjects based on data available before enrollment.After the data preparation steps, finally we ended up with 4616 records and 24 variables. The variables together with their possible values and description are detailed in Table 1. Out of the 4616 records, there are only 573 with no missing values. However, for various network architectures – where not all the variables are used – we can build the network on much more complete records

since the values that are not used for a given network can be ignored when dropping records that contain missing values. It is also possible to use incomplete records after imputing the missing values with a Bayesian network that was set up from the subset of the records that are complete.

To achieve better performance and increase the interpretability of the results we decided to categorize the variables. We used the quantile values for each variable creating two, three, four or five categories. In Figure 3 one can see the distribution of the Mathematics matura exam results and the quantile values. The green lines correspond to the terciles, the red lines correspond to the quartiles and the blue lines correspond to the quintile values.

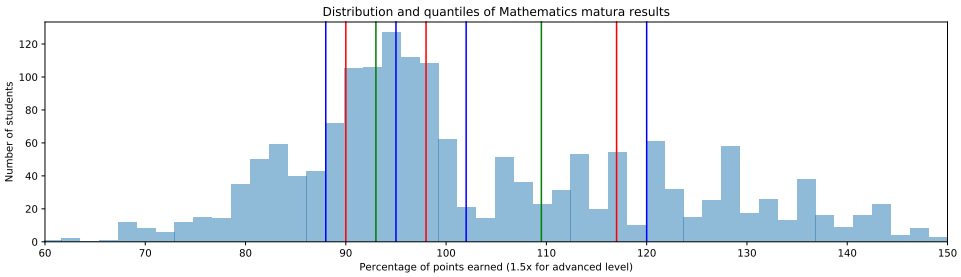


Figure 3: Distribution and quantiles of the Mathematics matura exam results

4.3 Modeling and evaluation

For the modeling part of our work we used the Pomegranate package [30], (version 0.11.1 under Python 3.7.4) that implements multiple algorithms for Bayesian network learning and evaluation.

We used the `from_samples` function to create the networks from the data. For structure learning an exact structure searching method, a related greedy algorithm, and the Chow–Liu tree building algorithm is supported in this tool. The first two are accelerated by dynamic programming and A* search, as described in Sec. 3.3.3.

Due to the high number of variables we were not able to run the exact algorithm for the full set of variables. Instead, we built networks on pre-enrollment variables and/or on reasonable subsets of the university courses, for more details see the captions of Figure 4.

For evaluation we used two types of measures. First, we wanted to test how well the networks predict the grades of the students based on pre-enrollment or first-semester evidence. For this we calculated the average mean squared error over all predicted subjects for the network:

$$MSE(y, \hat{y}) = \frac{1}{n \cdot l} \sum_{j=1}^{l} \sum_{i=1}^{n} (y_{i,j} - \hat{y}_{i,j})^2,$$

where n is the number of data points, l is the number of predicted nodes in the network, $y_{i,j}$ is the true grade of the i^{th} student in the j^{th} subject and $\hat{y}_{i,j}$ is the prediction for that grade.

Secondly, the grades were divided into two classes: success and failure, where success means that the student obtained a passing grade. Based on this, we divided students into two groups for each semester. The students in group one are those who completed all the courses of the semester successfully, while the students in the other group failed at least one of the courses. If we consider all courses from the first semester with this transformation, we can predict if the student is able to keep up with the sample curriculum or if the student is exposed to early dropout. This transformation turns the problem into a binary classification problem and makes it easy to calculate Accuracy, Precision, Recall, F-measure and Specificity:

In binary classification it is possible to evaluate the performance of a model based on four values (see Table 2), in the columns there are the true labels of the data points, in the rows the predicted labels, and the table shows the occurrences of the four different possible modeling results.

For a better understanding the following measures can be calculated:

1. Accuracy: the percentage of labels the model predicted correctly

$$\text{accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

2. Precision: the percentage of true positive labels in the predicted positive labels

$$\text{precision} = \frac{TP}{TP + FP}$$

3. Recall: the percentage of true positive lables the model predicted positive

$$\text{recall} = \frac{TP}{TP + FN}$$

4. F-measure: the F-β score is a combination of precision and recall

$$\text{F-}\beta \text{ score} = (1 + \beta^2)\frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}}$$

5. Specificity: the percentage of true negative labels the model predicted negative

$$\text{specificity} = \frac{TN}{TN + FP}$$

We decided to calculate Accuracy and Specificity. Accuracy has a very intuitive

interpretation, and it is a broadly accepted and used measure for the performance of binary classification models.

Specificity (or true negative rate) is meaningful in our case because it measures what proportion of students, who actually fall behind, is correctly detected as at-risk students. These are the students who are in the possible need of remediation, and their identification based on pre-enrollment variables is crucial.

In Figure 4, we can see examples of the built networks in different scenarios. The applied structure learning algorithm, the calculated metrics, and other attributes are detailed in Table 3. The predictions are based on the loopy belief propagation with the default parameter setting of the implementation. The Bayesian networks of Figure 4a and Figure 4b are built on all available variables, therefore these two are directly comparable. Their structures are similar, and the performance of the Chow–Liu version is comparable to the greedy version, which is rather surprising. However, the Chow–Liu version is the only network where the degree of language exam is not independent of everything else.

Networks are presented in a way that green edges go between evidence variables, blue edges go between predicted variables and red edges indicate time difference between variables. These do not always follow the natural causality of the variables, but such a constraint may reduce the predictive power of the networks.

The MSEs of models (a) and (b) show that prediction is better in the short run when we predict first-semester results based on the pre-enrollment variables, than in case of predicting second-semester results. We can observe a similar phenomenon in case of models (d) and (e), where we predict the second-semester results based on pre-enrollment variables, and first-semester results, respectively.

An easy to see pattern is that the Scholarship Index (24) is connected to all first semester subjects in the networks containing this variable. This is expected, due to the fact that it is calculated from the results of these subjects.

Node	Variable	Values (categories)	Description
1	Age	integers from 17 to 35 (cut at age 20)	the age in the year of enrollment
2	Place of preliminary studies	string (Budapest, major city, other)	indicates where the student studied before enrollment
7	Level of preliminary studies	secondary school or higher education	the level of study before enrollment
3	Mathematics grade	float (cut at 4.5)	the average of the student's grades in mathematics that were used in SP calculation
5	Humanities grade	float (4 categories using quantiles)	the average of the student's grades in literature, history and foreign language that were used in SP calculation
6	Other grade	float (5 and not 5)	the average of the student's grades in the fifth subject that was used in SP calculation
4	Degree of language exam	intermediate or advanced	the degree of the highest language exam of the student
8	Mathematics matura	float between 0 and 150 (5 categories using quantiles)	the percentage of points scored on the Mathematics matura, multiplied by 1.5 for advanced level exams
9	Other matura	float between 0 and 150 (5 categories using quantiles)	the percentage of points scored on the other matura used for MP calculation multiplied by 1.5 for advanced level exams
10–22	13 subject variables	1,2,3,4,5	the grade of the first enrollment of the student in the subject (fundamental mechanical engineering subjects)
24	Scholarship index	float (3 categories using quantiles)	$\sum_i \frac{\text{grade}_i \cdot \text{credit}_i}{30}$, where sum is taken over completed courses
23	Unsuccessful credits	positive integer (0, more than 5, other)	the amount of university credits that the student took but did not complete

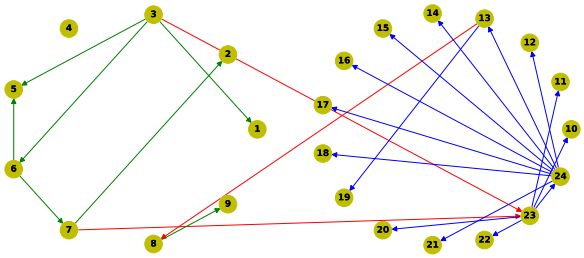
Table 1: The selected variables from the dataset

$\frac{\text{True label}}{\text{Prediction}}$	True	False
True	True positive (TP)	False positive (FP)
False	False negative (FN)	True negative (TN)

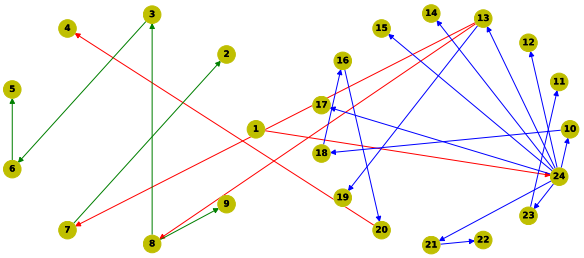
Table 2: Binary classification evaluation values

Attribute	a	b	c	d	e
Number of variables	24	24	17	16	15
Number of complete records	580	580	661	589	2338
Algorithm	greedy	Chow–Liu	exact	exact	exact
MSE (1)	1.743	1.803	1.909	-	-
MSE (2)	2.121	2.231	-	2.34	1.517
Accuracy (1)	0.688	0.681	0.716	-	-
Specificity (1)	0.75	0.841	0.828	-	-
Accuracy (2)	0.6	0.57	-	0.55	0.644
Specificity (2)	0.725	0.95	-	1.0	0.378

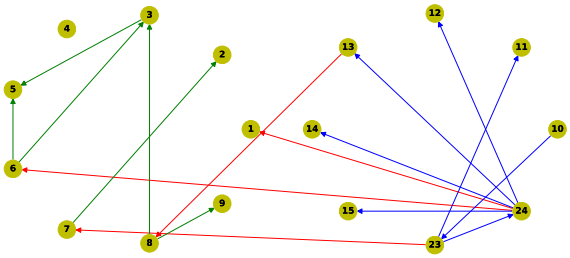
Table 3: Attributes of the built networks, (1) and (2) correspond to the predicted semester



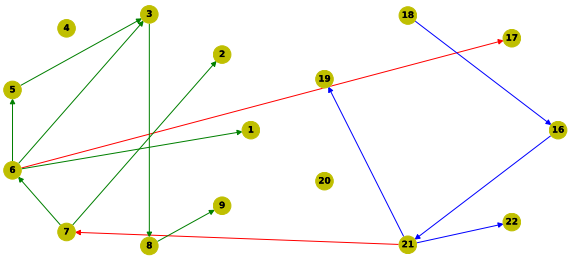
(a) Network for all variables (greedy version)



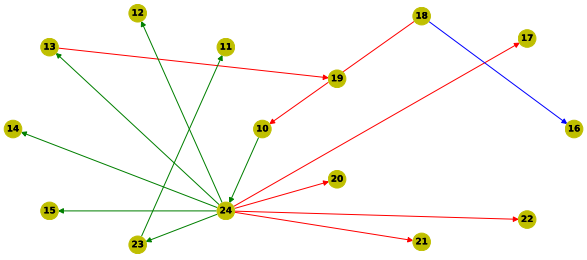
(b) Network for all variables (Chow–Liu version)



(c) Network for pre-enrollment data and first-semester subject results



(d) Network for pre-enrollment data and second-semester subject results



(e) Network for first- and second-semester subject results

Figure 4: Examples of the learned Bayesian networks

5 Summary

In this work, we reviewed part of the related literature on the topic of academic performance prediction. We presented the theoretical background of Bayesian networks, their learning and how inference can be done using learned networks.

We created Bayesian networks for the presented variables and evaluated them using mean squared error, accuracy and specificity.

The calculated measures suggest that the used pre-enrollment variables have limited prediction power. However, it is worth mentioning that the small sample size and the unpredictable nature of the loopy belief propagation may have altered the results.

On the other hand, the relatively high specificity scores indicate that the built models are suitable to identify students at risk at the time of their enrollment. The prediction power of our networks is comparable to similar studies working with other Bayesian tools, e.g. [10].

As a possible future research, we aim to identify other suitable pre-enrollment attributes, and investigate the applicability of other models that respect the time order of the educational domain.

6 Acknowledgment

I would like to express my gratitude to my supervisors, Roland Molontay and Máté Baranyi, who supported me through the creation of this work. Without their tireless help and comprehensive advice it could not have reached its current form.

References

[1] International Educational Data Mining Society. *Journal of Educational Data Mining*. URL: http://educationaldatamining.org/.

[2] Zacharoula Papamitsiou and Anastasios A Economides. "Learning analytics and educational data mining in practice: A systematic literature review of empirical evidence". In: *Journal of Educational Technology & Society* 17.4 (2014), pp. 49–64.

[3] Cristóbal Romero and Sebastián Ventura. "Educational data mining: a review of the state of the art". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 40.6 (2010), pp. 601–618.

[4] Alejandro Peña-Ayala. "Educational data mining: A survey and a data mining-based analysis of recent works". In: *Expert systems with applications* 41.4 (2014), pp. 1432–1462.

[5] Michalis Xenos. "Prediction and assessment of student behaviour in open and distance education in computers using Bayesian networks". In: *Computers & Education* 43.4 (2004), pp. 345–359.

[6] Patricio García et al. "Evaluating Bayesian networks' precision for detecting students' learning styles". In: *Computers & Education* 49.3 (2007), pp. 794–808.

[7] Zachary A Pardos et al. "Using fine-grained skill models to fit student performance with Bayesian networks". In: *Handbook of educational data mining* 417 (2010).

[8] Ahmad Slim et al. "Predicting student success based on prior performance". In: *2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*. IEEE. 2014, pp. 410–415.

[9] Erman Yukselturk, Serhat Ozekes, and Yalın Kılıç Türel. "Predicting dropout student: an application of data mining methods in an online education program". In: *European Journal of Open, Distance and e-learning* 17.1 (2014), pp. 118–133.

[10] Dorina Kabakchieva. "Predicting student performance by using data mining methods for classification". In: *Cybernetics and Information Technologies* 13.1 (2013), pp. 61–72.

[11] Marcell Nagy and Roland Molontay. "Predicting Dropout in Higher Education Based on Secondary School Performance". In: *2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES)*. IEEE. 2018, pp. 000389–000394.

[12] Marcell Nagy, Roland Molontay, and Mihály Szabó. "A Web Application for Predicting Academic Performance Identifying the Contributing Factors". In: *2019 47th SEFI Annual Conference*. SEFI. 2019.

[13] Steffen L Lauritzen. *Graphical models*. Vol. 17. Clarendon Press, 1996.

[14] Peter Spirtes et al. *Causation, prediction, and search*. MIT press, 2000.

[15] Marloes Maathuis et al. *Handbook of Graphical Models*. CRC Press, 2018.

[16] Nir Friedman and Moises Goldszmidt. "Building classifiers using Bayesian networks". In: *Proceedings of the national conference on artificial intelligence*. 1996, pp. 1277–1284.

[17] David Maxwell Chickering. "Optimal structure identification with greedy search". In: *Journal of machine learning research* 3.Nov (2002), pp. 507–554.

[18] Emilija Perković, Markus Kalisch, and Maloes H Maathuis. "Interpreting and using CPDAGs with background knowledge". In: *arXiv preprint arXiv:1707.02171* (2017).

[19] Markus Kalisch and Peter Bühlmann. "Estimating high-dimensional directed acyclic graphs with the PC-algorithm". In: *Journal of Machine Learning Research* 8.Mar (2007), pp. 613–636.

[20] Bojan Mihaljević, Concha Bielza, and Pedro Larrañaga. "Learning Bayesian network classifiers with completed partially directed acyclic graphs". In: *International Conference on Probabilistic Graphical Models*. 2018, pp. 272–283.

[21] Richard E Neapolitan et al. *Learning Bayesian networks*. Vol. 38. Pearson Prentice Hall Upper Saddle River, NJ, 2004.

[22] Thomas Dyhre Nielsen and Finn Verner Jensen. *Bayesian networks and decision graphs*. Springer Science & Business Media, 2009.

[23] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 2014.

[24] Ross D Shachter. "Bayes-ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams)". In: *arXiv preprint arXiv:1301.7412* (2013).

[25] Changhe Yuan, Brandon Malone, and Xiaojian Wu. "Learning optimal Bayesian networks using A* search". In: *Twenty-Second International Joint Conference on Artificial Intelligence*. 2011.

[26] Gregory F Cooper. "The computational complexity of probabilistic inference using Bayesian belief networks". In: *Artificial intelligence* 42.2-3 (1990), pp. 393–405.

[27] Kevin P Murphy, Yair Weiss, and Michael I Jordan. "Loopy belief propagation for approximate inference: An empirical study". In: *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc. 1999, pp. 467–475.

[28] Amen Ajroud et al. "Loopy belief propagation in Bayesian networks: origin and possibilistic perspectives". In: *arXiv preprint arXiv:1206.0976* (2012).

[29] Wes McKinney. "Data Structures for Statistical Computing in Python". In: *Proceedings of the 9th Python in Science Conference*. Ed. by Stéfan van der Walt and Jarrod Millman. 2010, pp. 51–56.

[30] Jacob Schreiber. "Pomegranate: fast and flexible probabilistic modeling in python". In: *Journal of Machine Learning Research* 18.164 (2018), pp. 1–6.