ISBN 978-83-60810-51-4

Bayesian networks in business analytics

Dr Michael Ashcroft
Computer Science Department
Uppsala University
Uppsala, Sweden
mikeashcroft@inatas.com

***Abstract*—Bayesian networks are a popular and powerful tool in artificial intelligence. They have many applications in commercial decision support. The point of this paper is to provide an overview of the techniques involved from this perspective. We will proceed by giving a simplified mathematical overview of what Bayesian networks are and the flavors they come in. We then look at how they can be created or learnt from data and the situations that lead to the use of ensemble models. Then we look at how an application of such a technology would proceed, using the human resources example of talent retention for international firms in China, examining the full process rather than technology specific elements. Finally we look at the outputs that would be generated from such an application.**

***Index Terms*—Bayesian networks, decision assistance, business analytics, stochastic modeling**

I. Introduction

Bayesian networks are a popular and powerful tool in artificial intelligence. They have many applications in commercial decision support. The point of this paper is to provide an overview of the techniques involved from this perspective. We will proceed by giving a simplified mathematical overview of what Bayesian networks are and the flavors they come in. We then look at how they can be created or learnt from data and the situations that lead to the use of ensemble models. Then we look at how an application of such a technology would proceed, using the human resources example of talent retention for international firms in China, examining the full process rather than technology specific elements. Finally we look at the outputs that would be generated from such an application.

II. Bayesian Networks

Recall from probability theory that two random variables, X and Y, are independent if and only if $P(X,Y) = P(X)P(Y)$. Analogously, X and Y are conditionally independent given a third random variable Z if and only if $P(X,Y|Z) = P(X|Z)P(Y|Z)$, which is equivalent to:

$$P(X|Z) = P(X|Y,Z) \tag{1}$$

Also recall that the chain rule for random variables says that for n random variables, $X_1, X_2, ...X_n$, defined on the same sample space S:

$$\begin{aligned} P(X_1, X_2, ...X_n) \quad = \quad & P(X_n|X_{n-1}, X_{n-2}, ...X_1) \\ & P(X_{n-1}|X_{n-2}, ...X_1) \\ & ...P(X_2|X_1)P(X_1) \end{aligned} \tag{2}$$

Imagine we have five random variables: $\{A, B, C, D, E\}$. From the chain rule, we know that:

$$\begin{aligned} P(A,B,C,D,E) \quad = \quad & P(E|A,B,C,D) \\ & P(D|A,B,C)P(C|A,B) \\ & P(B|A)P(A) \end{aligned} \tag{3}$$

We can represent these five conditional independencies by means of a directed acyclic graph (DAG) and a set of conditional distributions, where:

- Each random variable is mapped to a node of the DAG
- Each node has associated with it the conditional distribution for its variable
- Each node has incoming edges from the nodes associated with the variables on which the node's conditional distribution is conditional

Such a representation is a Bayesian network. It satisfies the Markov conditions:

Definition A direct acyclic graph (DAG), G, with nodes N_G, a joint probability distribution, P, of random variables D_P, and a bijective mapping $f : D_P \Rightarrow N_G$ satisfies the Markov Condition if and only if for all $v \in D_P$, where $n = f(v)$, v is conditionally independent given P of the variables that are mapped to the non-descendants of n given the variables that are mapped to the parents n.

TABLE I: Conditional independencies required of random variables the DAG in Figure 1 to be a Bayesian Network

Node	Conditional Independencies
A	-
B	C and E, given A
C	B, given A
D	A and E, given B and C
E	A, B and D, given C

If we know no more than the decomposition given to us by the chain rule in equation 3, the associated Bayesian network's DAG will be complete (since each variable is conditional on all those prior to it in the decomposition order). However, imagine that we know that certain conditional independencies exist as specified in table I. From the definition of conditional independence, we know that:

- $P(C|B,A) = P(C|A)$
- $P(D|C,B,A) = P(D|C,B)$
- $P(E|D,C,B,A) = P(E|C)$

Accordingly:

$$P(A,B,C,D,E) = P(E|C)P(D|C,B) P(C|A)P(B|A)P(A) \quad (4)$$

Whenever we simplify the conditional distributions in virtue of a known conditional independence relation, we remove an edge on the DAG of our Bayesian network representation. In this case, the resulting network is given by figure 1.

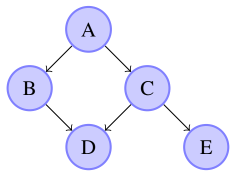

Fig. 1: A DAG with five nodes

Loosely speaking, what we have done is pull the joint probability distribution P apart by its conditional independencies. A Bayesian network is an encoding of these conditional independencies in the DAG topology coupled with the simplified conditional distributions. Note that the conditional independencies are encoded by the *absence* of edges.

The reason why Bayesian networks are useful is that this structure gives us a means of performing tractable calculations locally on the network whilst using all information of the joint distribution. It has been proven that every discrete probability distribution (and many continuous ones) can be represented by a Bayesian Network, and that every Bayesian network represents some probability distribution. Of course, if there are no conditional independencies in the joint probability distribution, representing it with a Bayesian network gains us nothing. But in practice, while independence relationship between random variables in a system we are interested in modeling are rare (and assumptions regarding such independence dangerous), conditional independencies are plentiful.

III. Discrete and continuous Bayesian networks

Bayesian networks come in a number of varieties according to the restrictions, if any, placed on the forms the conditional probability distributions can take. We will concentrate on discrete Bayesian networks, where continuous variables are discretized during preprocessing. Discrete Bayesian networks:

- Deal with continuous variables by discretization into arbitrarily many arbitrarily sized intervals. Various methods can be used for choosing the intervals, including automated clustering methods.
- Are not limited by linear and/or Gaussian noise assumptions.
- Are unrestricted by an apriori structure beyond that imposed by discretization. This is both good and bad:
  - They follow the data when it leads.
  - Cases unencountered in the learning data will take the apriori distribution, which is generally uniform. There are situations where this is undesirable (e.g. where closely related cases all strongly evince a particular structure). Methods exist that provide variance estimates which help indicate when dangerously novel cases are encountered.
- Permit efficient and accurate variance estimation on aposteriori probability distributions.
- Permit the use of exact inference algorithms.
- Permit, when combined with decision theoretic extensions, the use of exact utility maximization algorithms for generating decision policies (including on meta-models).
- Can be used as the automated basis for the production of general Bayesian networks (see below).

Other common forms include Gaussian and hybrid discrete/Gaussian networks. Automated algorithms exist for the automatic learning of, and exact inference on, such networks. These, though, require Gaussian variables to be linear combinations of their parents with Gaussian noise (potentially conditional on the values taken by their discrete in hybrid networks).

General Bayesian networks, where any conditional probability distribution in the network can be of any type, are possible. Currently, no automated learning algorithms or exact inference algorithms are known for such networks, but sampling methods do exist for inference. When such networks are desired, it has been suggested that discretized variables be used for the structural learning process [1]. After the conditional independencies are discovered by this process, bespoke conditional distributions for each variable given its parents can be fitted to the non-discretized data given domain knowledge.

Rigorous comparisons of accuracy between discrete and Gaussian networks are difficult to find. My *conjecture* is that discrete networks often offer significant advantages over their Gaussian cousins because of their non-linearity, their minimal imposition of structure on the data during learning and the current relative sophistication of the algorithms that can be perform on them. Obviously, though, the degree to which the system in question meets the assumptions involved in the Gaussian models is a key factor here. Where this is unclear but the inclusion of non-discretized continuous variables unavoidable, a good option is to custom-build a general network from the discrete conditional independence structure identified by the automated discrete learning algorithm.

IV. Learning

A. *Learning a Network from Expert Causal Knowledge*

Importantly, a causal network is a conditional independence encoding of the type described previously. Thus if we have knowledge of the causal relationships pertaining between the variables we are modeling then we can immediately produce the DAG structure of the Bayesian network. In such cases, domain experts may also directly specify the conditional distributions. Where this does not occur, we need to learn the conditional distributions from data. Discrete and Gaussian networks have efficient automated algorithms for parameter learning.

B. Learning a Network from Data

Where no expert knowledge is available we must learn the conditional independencies encoded in the network from data. The basic procedure is to perform a heuristic search on the space of possible sets of conditional independencies in order to obtain the best such set. This is complicated by the fact that multiple topologies can encode the same set of conditional independencies. To overcome this, we instead search equivalence classes of topologies/conditional independence sets [2].

Efficient learning algorithms occur in both the Discrete and Gaussian cases. In the discrete case, an algorithm exists that includes an optimality guarantee: As the size of our learning data approaches infinity, the probability of learning the globally optimal model (with a single iteration of the algorithm) approaches 1 [3]. Where the conditions are not met, a more general hill climb algorithm produces better results. The result is that the most robust learning algorithm utilizes the inclusion boundary algorithm for its first search and then repeatedly restarts the general hill climb algorithm.

The most principled and popular fitness function is the Bayesian Dirichlet score. This calculates the aposteriori probability of a set of conditional independencies given the learning data. Accordingly the network we obtain is that which represents the most probable set of conditional independencies.

C. Meta-Models

Often a single network structure dominates alternatives. Where this is not the case, we can collect multiple high scoring networks by, for example, collecting all networks that are at least $\frac{1}{x}$ as probable as the best network, for some x. These networks can be weighted by their relative probability and inference can be performed over the entire set. Effectively, we now reason using not just our best hypothesis of the system structure, but a set of plausible hypotheses, weighted for their plausibility. This can be a very powerful method, and all inference algorithms discussed below can be run on such a meta-model.

D. Missing Data

Structural learning can be performed with missing data items in the learning set. Common algorithms for dealing with this are the Gibbs Sampler and Expectation Maximization.

V. System Analysis

A. Markov Blanket

The Markov Condition entails other conditional independencies. Because of the Markov Condition, these conditional independencies have a graph theoretic criterion called D-Separation (see [4] for detailed definition). Accordingly, when one set of random variables, Γ, is conditionally independent of another, Δ, given a third, Θ, them we will say that the nodes representing the random variables in Γ are D-Separated from Δ by Θ.

The most important case of D-Separation/Conditional Independence is:

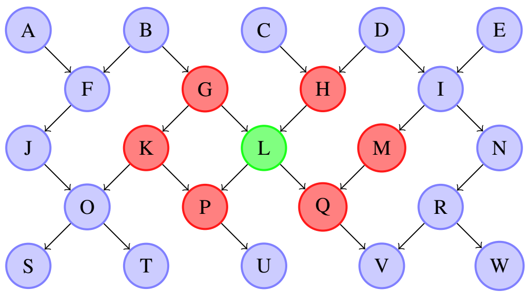

Fig. 2: The Markov Blanket of Node L

- A node is D-Separated of the entire graph given its parents, its children, and the other parents of its children.

Because of this, the parents, children and other parents of a node's children are called the *Markov Blanket* of the node.

This is important. Imagine we have a variable, α, whose probability distribution we wish to predict and whose Markov Blanket is the set of nodes, Γ. If we know the value of every node in Γ, then we know that there is no more information regarding the value taken by α. This can be generalized to look for the nodes that provide no additional information regarding the set of nodes we are interested in given the variables we are certain we will be always be able to observe the values of.

In this way, if we are confident that we can always establish the values of some of the variables our network is modeling, we can often see that some of the remaining variables are superfluous, and we need not continue to include them in the network nor collect information on them. Since, in practice, collecting data on random variables can be costly, this can be very helpful.

B. Causal Analysis

The connection between causality and conditional independence has lead to the use of Bayesian networks in causal analysis, often in conjunction with manipulation tests. See [4] for details.

VI. Inference

Inference is the practice of obtaining aposteriori probability distributions for variables of interest, given the available evidence. Once a Bayesian network has been created/learnt, we can use the network to calculate the *a posteriori* probability distributions for a subset of variables, Γ, given the observation that a second subset, Δ, has taken particular values. In the discrete and Gaussian cases we are also able to obtain accurate estimations of the variance of such *a posteriori* distributions, permitting the calculation of 'error bars' around the probability estimates.

Efficient exact algorithms exist for both the discrete and Gaussian cases. In the general case, or if a discrete network is sufficiently complex, exact inference algorithms is intractable. In such cases we turn to sampling techniques. The most important (largely for its extension in the application of particle filters in the case of general Bayes filters) is importance sampling.

VII. Automated Decision Making

Bayesian networks can be extended with utility, decision and information nodes to produce 'Influence Diagrams'. The utilities are entered by domain experts and specify the value to the user of the system being in particular states. Variables under the user's control are designated decision variables. Additionally an information order is stipulated. This is based on the partial order in which the decisions must be made as well as the specification of information variables, which are variables not under the user's control that, if they are not currently known, will be known before the performance of a particular set of decisions. Often this is because they will be known only after the performance of earlier decisions.

So extended, inference is performed on Junction Trees whose topology respects the information order and which has both probability and utility potentials associated with the clusters and intersection sets. Transmission of information through this structure now also includes a utility maximization procedure for each decision variable. The result is the output of decision policies which specify the value to which each (relevant) decision variable ought to be set to maximize expected utility given the evidence that will obtain at the time of the decision. Details of the algorithm can be found in [5].

VIII. An Example

Talent retention has become a significant issue for international firms in China. High quality local employees often quickly switch companies and employees who are trained by their employers often seek better positions once their skill set has been enhanced. As I response, I am involved in efforts to assist a number of companies systematically hire and retain quality employees. The life cycle of such a project is the following:

1) Establish variables of interest
2) Collect data
3) Encode domain knowledge
4) Create predictive model
5) Collaborative utility estimation
6) Test model
7) Implementation of access system
8) Post-implementation

A. Establish tasks and variables of interest

The first task is to establish which variables *may* be of interest. At this point, unless collection costs for certain variables are prohibitive, it is important to have all potential relevant variables included. Generally, variable selection will be the responsibility of domain experts—in this case, we plan to talk with the human resources departments of the companies involved and these companies' employees (via anonymous surveys). Given such a list of variables, decisions such as whether a dynamic or static model might be created must be answered - though it may be that either could be suitable, and so appropriate data for both must be sought. Variable choice may be constrained significantly if we are required to use historic data rather than specify the data that is to be collected. Often the initial network must be created from historical data but new data can be specified that will come to be included at a later point.

The list of variables to be collected in our example might look something like:

- Employee Related
  - Employment history
  - Education
  - Language ability
  - Age
  - Sex
  - Relationship status
  - Demand for applicant's skills
- Position Related
  - Salary
  - Professional training at years 1,2,3,+.
  - Language training at years 1,2,3,+.
  - Overseas opportunities at years 1,2,3,+.
  - Position type
  - Hours
  - Holidays
  - Career path opportunities at years 1,2,3,+.
- Company Related
  - Chinese managerial viability (including head office)
  - Prestige
- Outcome
  - Hire
  - Length of employment
  - Average company satisfaction with employee over length of employment

As such a model is to be used only for evaluating new hires, it makes sense to model it statically. If the task were, instead, to evaluate policies towards current staff over the coming six months, it would make sense to use a dynamic model. In other words, rather than simply collect this information for an employee for their time at the company, we would seek to obtain the information for each year of their employment and see whether, at the end of that year, the employee was retained, and how satisfied the company was with them. The resulting model should not only assist in hiring employees likely to stay with the company, but also to retain them as well as promote practices that lead to high-quality employee performance. It could also be used to soft-sense and/or maximize employee satisfaction.

In fact this second model seems very useful. It is likely that anyone interested in the question 'How do we hire valuable employees who will stay with the company?' will also be interested in the question 'How do we maximise the value of our employees and ensure they remain with us?'. Further, the two models incorporate many of the same variables. Therefore we should probably expand the defined task of the model and with it the variables we will collect. (For the purposes of this paper, though, we will continue with the current example.)

Of course this 'discovery' was deliberate. It emphasises that the uses of the planned model can multiply as it takes shape.

Where these eventual uses are clearly related, as above, this is good. It can, though, necessitate reiterations of the process. If we add to or alter the assigned tasks, we must ensure that we have all potentially relevant variables the additional or emended tasks too. It is also very important not to try and create the everything model! Where uses are not clearly related and require large numbers of new variables to be tracked, it is often better to create a separate model as part of a (partially-)separate project. Although obviously relevant to employee retention, we should not include variables that track the general state of the Chinese economy. We must choose a cut off point, and here we contain ourselves to specifying current demand for the applicants skills.

It will also be necessary to establish which variables are under the control of end-users. In our case this means establishing which of the variables chosen are under the control of participating companies. Sometimes this is not clear-cut. In our case, a company can seek to control the prestige associated with working at it to some degree, but not directly or deterministically. Should companies wish to include some indication of this partial control, we would have to introduce the elements of this process that they can control. In this case, I would suggest that the prestige variable be treated as an environment/chance variable.

B. *Encode domain knowledge: Pre-learning*

Expert knowledge can be encoded in the network. This will take the form of specifying relationships between variables (edges in the network) that are required or prohibited, and concrete or defeasible parameters for the conditional probability distributions that relate the variables. In certain circumstances the network will be entirely created from expert knowledge in this fashion.

In our example, the current demand for the applicants skills is something that is unlikely to have been tracked previously, so is something that can only be incorporated into the intial model by encoding domain knowledge about its relationship to other variables. As data is collected, this initial specification will evolve.

In practice, where data exists it is often be better to leave the learning process unconstrained and add known relationships only insofar as they were to weak in the learning data for the learnt model to pick up.

C. *Collect data*

Where the network is not entirely based on expert knowledge, data will be required to learn the model from. Even when only using expert knowledge, it is useful to have data to test the validity of the model.

In our case, the data is internal to the company and it is likely that participating companies will need little assistance in collecting it. It will be essential, though, to ensure data security and confidentiality. Participating companies will not want their data to be viewed by other participants or outside entities, nor will individuals wish for their personal records to be reconstructible from the finished product.

D. *Create network(s)*

Where the network is not entirely based on expert knowledge, the model or meta-model will be learnt from the data collected. During this process a number of methods permit us to test whether we have sufficient data. If we do not, we must obtain more (or switch to a less data intensive method).

Further, during learning redundant variables will be found and eliminated: As explained above, the topology of a Bayesian network indicates which variables provide no information regarding the state of the variables of interest to us given the variables we can be certain of always knowing the values of. In our case we might find that the individual's gender is correlated with our variables of interest, but only insofar as it is related to a persons level in their company (perhaps men are more common in managerial positions). Given we know someones position in the company, their sex contains no additional information regarding whether they will be a satisfactory employee nor whether they are likely to leave the company. As its is the employees companies who will be using the models, employees' position in their company can always be established and the gender of the employee is redundant.

We will also determine whether a single network dominates the possible hypotheses regarding the system being modeled, or whether we should utilize a meta-model made up of multiple networks as explained above.

E. *Encode domain knowledge: Post-learning*

In certain circumstances additional domain knowledge will be encoded in the model post learning. This may include specifying utility values or transforming the network from a discrete network to a general Bayesian network.

In our case the utilities involved will be different for different end users and even for the same end-users in different circumstances. This is often the case. Accordingly, the utility values will be alterable just as the value of any other variable and it is to a large degree the responsibility of the end user to specify their own. Further, the advantages of efficient, exact decision theoretic algorithms would make the use of discrete networks ideal - though some parameter 'smoothing' might be useful to minimize the problems associated with novel cases discussed above. If there are particular relationships that participating companies have experienced which are not present in the model, these should be tested for and, assuming they are found to be present, manually added.

F. *Test model*

Routine testing is part of the modeling process. Ideally, though, at this point the model should be used and evaluated in a pilot program where it will be incorporated into the access system (see below), encounter new real life cases and be used by real end-users. The guidance the system offers should be useful (not trivial). End users should find the access system easy to use, and the 'reasoning' behind decisions understandable. Difficulties found here can be incorporated into training.

Whether this step is necessary depends upon the size of the eventual user base. When the user base is restricted to a handful of individuals, no such testing is possible. In our case, the system will be used by human resource staff of numerous companies and a pilot program would be essential.

G. Implementation of access system

The access system is the software—and, in certain cases though not in ours, hardware - that will be use to enter data into and query the finished model or meta-model for decision assistance and predictions. It must be deployed and end-users trained in its use. For us this means installing, in all participating companies, the finished software application, which would be a non-technical wrapper that permits all required interactions with the network, as well as providing training courses to end-users.

It will be essential for end users to be able to enter new data, and for the model to react to this data. In our case, data regarding new hires will occur automatically as end-users work with the program. But data regarding policies to employees, satisfaction with and of employees and employee retention will need to be specifically entered. It will be necessary to ensure that the finished application permits such data-entry, and would be advantageous if it alerted designated individuals if such data-entry does not occur.

Bayesian networks can be set to automatically adapt the parameters of their conditional probability distributions to new data. On the assumption that the underlying system is stable, this generally suffices. If this assumption is questionable, an ongoing structural learning process should continue to model the relationships found in recent data to ensure that no abrupt alterations of the system have occurred and hence that the network remains valid for the domain. Our system is certainly subject to shocks from outside the variables we have included - for example the Chinese economy could enter a prolonged downturn - and so ideally such a process should be included as part of the final application and automatically operate as data becomes available. Minimally, it should be possible for humans to periodically implement such a process.

H. Post-implementation

The access system will require ongoing support and maintenance. As new individuals become end-users they will need training. Finally, to ensure the model stays accurate in changing circumstances, ongoing data acquisition and collation will be required.

In our case, this means providing software support and training for novice end-users such as new HR employees. The low level data acquisition program should be able to be implemented by participating companies.

IX. Outputs

So what might we expect to obtain from such a system? Let us imagine a 26 year old, unmarried male without children. He has an undergraduate engineering degree from a high-quality Chinese institution and is professionally competent in English. His last job was low-level management, and he specifies an expected salary of RMB7000/month. He has held three jobs in the last fours years.

1) Should the prospective employee be hired.
2) If he is hired, what sort of contract and conditions should he be given to maximise the expected value of the hire, measured in terms of satisfaction with his contribution and retention whilst minimizing costs.

For example he might be offered a number of attractions at a specified future time - perhaps an overseas placement opportunity after two years, or ongoing professional education paid for by the company from year 1-3, etc. It may be that he should *not* be offered additional language training, since this is unlikely to increase either his or the company's satisfaction but, if successful, will greatly increase the risk that he will be leave (because headhunted!).

A more sophisticated situation is where certain characteristics remain unknown (perhaps at a resume sifting stage). The company may decide not to trust applicants' claims about their language ability, or plan to have applicants take additional tests. Decision policies will be given that specify whether an applicant should be hired and, if so, the details of the contract for each of the possible values of the unknown variables. For example, the applicant in question might be hired only if he performs outstandingly on the test (since there might be a high-risk of him leaving), but in such a case be offered a lucrative salary and numerous inducements to stay (since he would be a valuable employee if he could be retained). A example decision policy representing the above, and assuming that an appropriate test outcome variable was included, is given in table II.

TABLE II: Decision policy for applicant, given test result

Test Score	Hire	Salary	Overseas placement	Professional Training	Language Training
< 60	No	-	-	-	-
60-70	No	-	-	-	-
70-80	No	-	-	-	-
80-90	No	-	-	-	-
90-100	Yes	14,000	After two years	After three years	No

The network would also be able to produce aposteriori predictions for the variables of interest, given current knowledge and on the assumption that the decision policies specified are followed. It could also be run with the decision variables treated as chance variables or set to other options to see the aposteriori given just current knowledge, or to test alternative decision policies.

If we imagine the 'Average company satisfaction with employee over length of employment' variable takes values from 1 to 10 (representing some suitable function from yearly reviews) and the 'Length of employment' variable takes the values 1 to 5 and >5 (representing the employee leaves the company before that many years), then we might be given the distributions represented in tables III and IV.

References

[1] S. Monti and G. Cooper, "Technical report: Learning hybrid bayesian networks from data," 1997.

TABLE III: A posteriori probabilities for variable 'Average company satisfaction with employee over length of employment'

Value	Apriori Probability
1	≈ 0
2	≈ 0
3	.01
4	.02
5	.04
6	.08
7	.10
8	.43
9	.22
10	.10

TABLE IV: A posteriori probabilities for variable 'Length of employment'

Value	Apriori Probability
1	.08
2	.07
3	.05
4	.01
5	.01
>5	.78

[2] D. Chickering, "Learning equivalence classes of bayesian-network structures," *Journal of Machine Learning Research*, no. 2, pp. 445–498, 2002.
[3] D. Chickering, "Optimal structure identification with greedy search," *Journal of Machine Learning Research*, no. 3, pp. 507–554, 2002.
[4] R. Neopolitan, *Learning Bayesian Networks*. Pearson Prentice Hall, 2004.
[5] F. V. Jensen and T. D. Nielsen, *Bayesian Networks and Decision Graphs*. Springer, 2nd ed., 2007.