

The copyright of this thesis vests in the author. No quotation from it or information derived from it is to be published without full acknowledgement of the source. The thesis is to be used for private study or non-commercial research purposes only.

Published by the University of Cape Town (UCT) in terms of the non-exclusive license granted to UCT by the author.

University of Cape Town  
Faculty of Commerce

Master of Business Science Thesis

# **A Comparison of Methods for Modelling Rates of Withdrawal from Insurance Contracts**

by

**Bradley Smith**

Supervisor: Iain MacDonald

Cape Town, 2009

## **Abstract**

Withdrawal from insurance contracts can be a significant risk for insurers. Withdrawal rates can be difficult to predict because withdrawal is influenced by a number of inter-related factors related to, *inter alia*, the sales process, characteristics of the insurance contract, characteristics of the contract holder, and economic variables. Existing methods used to model and predict withdrawal rates are initially reviewed. Two additional methods which have been proposed in the literature as means for modelling insurance risks are neural networks and Bayesian networks. These two methods are utilised in order to build models to compare their predictive ability with a commonly used method for modelling withdrawal rates, namely logistic regression. The models are evaluated on the basis of practical usage criteria before they are evaluated on the basis of Receiver Operating Characteristics metrics and by using cross-validation. Bayesian networks and logistic regression emerge over neural networks as preferable methods for modelling rates of withdrawal from insurance contracts.

---

# Contents

---

<b>Contents</b>	<b>i</b>
<b>1 The nature of withdrawal rates</b>	<b>3</b>
1.1 Definition of withdrawal and withdrawal rate . . . . .	3
1.2 Withdrawal risk . . . . .	3
1.3 Factors affecting withdrawal rates . . . . .	4
<b>2 Review of existing methods for estimating withdrawal rates</b>	<b>7</b>
2.1 Introduction . . . . .	7
2.2 Graduation of crude rates estimated from exposure data . . . . .	7
2.3 Statistical models . . . . .	7
2.3.1 Linear multiple-regression models . . . . .	8
2.3.2 Generalised linear models . . . . .	8
2.4 Financial-economic models . . . . .	9
<b>3 Proposed alternative methods for estimating withdrawal rates</b>	<b>11</b>
3.1 Introduction . . . . .	11
3.2 Bayesian networks . . . . .	12
3.2.1 Bayesian network definition . . . . .	12
3.2.2 Reasoning in a Bayesian network . . . . .	14
3.2.3 Estimating parameters in a Bayesian network . . . . .	15
3.2.4 Learning causal structure in a Bayesian network . . . . .	16
3.3 Neural Networks . . . . .	16
3.3.1 Neural network definition . . . . .	16
3.3.2 Training the neural network . . . . .	19
<b>4 Application of proposed methods</b>	<b>21</b>
4.1 Description of data . . . . .	21
4.2 Bayesian network . . . . .	22
4.2.1 Learned Bayesian network . . . . .	23

4.2.2	Proposed Bayesian network . . . . .	24
4.3	Neural network . . . . .	25
4.4	Logistic regression model . . . . .	26
<b>5</b>	<b>Comparison of methods</b>	<b>29</b>
5.1	Qualitative comparison of methods . . . . .	29
5.1.1	Theoretical considerations in comparison of methods . . . . .	29
5.1.2	Practical considerations in comparison of methods . . . . .	30
5.2	Quantitative comparison of predictive ability . . . . .	30
<b>6</b>	<b>Conclusions and recommendations</b>	<b>35</b>
6.1	Conclusions . . . . .	35
6.1.1	AUC preferred over RMSE as a metric for predictive ability	35
6.1.2	Proposed Bayesian network outperforms learned Bayesian network . . . . .	35
6.1.3	Neural network has lower predictive ability than other models but is more stable over time . . . . .	36
6.1.4	Logistic regression model has good performance, excellent run-time, but is less stable over time . . . . .	36
6.1.5	Conclusions for model selection . . . . .	36
6.2	Recommendations . . . . .	37
6.2.1	Bayesian networks . . . . .	37
6.2.2	Neural networks . . . . .	37
6.2.3	Improving efficiency and investigating optimal recalibration frequency . . . . .	37
	<b>Bibliography</b>	<b>39</b>
	<b>List of Figures</b>	<b>43</b>
	<b>List of Tables</b>	<b>44</b>

---

# Acknowledgements

---

I would particularly like to thank my supervisor Iain MacDonald for his insights, incredible attention to detail and entertaining conversations. My family also deserve a special mention for their perpetual support and encouragement.

## Chapter 1

---

# The nature of withdrawal rates

---

### 1.1 Definition of withdrawal and withdrawal rate

"Withdrawal" in the context of an insurance contract is commonly used to refer to the event of a policyholder prematurely terminating the contract with or without a termination benefit. "Withdrawal" thus encompasses both surrender (benefit available on termination) and lapsation (no benefit available on termination). A "withdrawal rate" for a defined period (usually 1 year) is the percentage of contracts within a specified group that lapse during the defined period. Thus for a group consisting of  $N_t$  members at the start of period  $t$  and for which there are  $l_t$  withdrawals during period  $t$ , the withdrawal rate  $w_t$  is simply

$$w_t = \frac{l_t}{N_t} \quad 0 \leq w_t \leq 1.$$

Prospectively the withdrawal rate is unknown and its value may be said to arise from a random variable  $W(t)$ . Thus, when referring to future withdrawal rates, the term "withdrawal rate" may more accurately be termed a "probability of withdrawal" because the actual rate is unknown until the time period over which the rate is being measured has elapsed. Therefore  $w_t = P(W(t) = \textit{withdrawal})$ . When we use a model to estimate the probability of withdrawal, the estimated probability is denoted by  $\hat{w}_t$ .

### 1.2 Withdrawal risk

"Withdrawal risk" is the risk that actual withdrawal rates experienced are more than expected and where the insurer makes a loss on the event of "withdrawal". Withdrawals may be a source of significant risk for a life insurer, especially at early durations where asset share is likely to be negative or smaller than the surrender value

because of high initial expenses that are spread over the full term of the contract. Withdrawals can cause a liquidity risk for the insurer if surrender values are offered; further if withdrawal rates are volatile, an insurer may invest in more liquid, but lower-returning assets in order to reduce the liquidity risk. There may also be second order impacts on the insurer's business such as an impaired public perception of the company based on the argument that high withdrawal rates are a reflection of poor value delivery. Depending on the reasons for withdrawal, there may be a significant adverse selective effect on mortality. Withdrawals are affected by many factors and are traditionally quite volatile and difficult to predict with a high degree of accuracy. Effective modelling of withdrawal rates, where the models are capable of accurately capturing the system dynamics leading to changes in withdrawal rates, can assist an insurer to understand its key withdrawal risk exposures under a range of scenarios. By understanding the risk factors and their interactions and modelling withdrawal rates more accurately, the insurer can reduce its risk and improve its profitability by implementing better product design, pricing, sales and distribution strategy.

### 1.3 Factors affecting withdrawal rates

Many different factors can potentially impact on the withdrawal rates. There may also be significant non-linear interactions between some of the factors. Some factors, such as interest rates, are naturally quantifiable and have data readily available for incorporation into a model. Others however, such as an insurer's public image, are less readily quantifiable and need to be indirectly modelled through the use of a suitable proxy (e.g. a consumer confidence survey can be used for estimating an insurer's public image). A fairly broad sample of factors is presented below and categorised into: "characteristics of the insurance contract", "characteristics of the policyholder", "sales process" and "macro factors".

- Characteristics of the insurance contract
  - Premium size
  - Premium size as a % of income
  - Premium frequency
  - Method of payment
  - Policy type (e.g. term assurance, endowment)
  - Commission structure
  - Surrender value / surrender penalty
  - Degree of initial underwriting
  - Original term of policy
  - Duration in force

- Characteristics of the policyholder
  - Age
  - Sex
  - Education
  - Income
  - Number of dependants
  - State of health
- Sales process
  - Distribution channels and target markets (as a proxy for other factors)
  - Who initiated the sale
  - After sales service
- Macro factors
  - Investment returns (if market-linked)
  - Degree of competition
  - Position in the business cycle
  - Interest rates
  - Inflation
  - Household indebtedness
  - Levels of unemployment
  - GDP growth
  - Company's reputation
  - Company's financial position

Significant interactions may exist between some of the factors, particularly the macro factors. It is worthwhile taking note of the relationships between the factors and looking at the deeper reasons giving rise to some of the combinations of the factors since this can help predict scenarios leading to higher withdrawal rates. A range of interactions between such factors may include:

- The position in the business cycle affects inflation which in turn affects interest rates if the central bank targets inflation using interest rates.
- Interest rates affect the level of household indebtedness.
- Multiple economic factors affect the investment returns and the company's financial position.

Chapter 2 explores current approaches to estimating withdrawal rates, some of which are able to capture the interactions between factors, albeit simplistically, whilst others ignore them altogether. Chapter 3 then goes on to propose alternative methods which deal with some of the shortcomings of existing methods.

## Chapter 2

---

# Review of existing methods for estimating withdrawal rates

---

### 2.1 Introduction

Multiple methods exist for estimating withdrawal rates which typically can be categorised as one of:

- graduation of crude rates estimated from exposure data;
- statistical models;
- models based on financial-economics.

We briefly review some of the methods falling into these categories and give the associated problems inherent in their use.

### 2.2 Graduation of crude rates estimated from exposure data

Crude rates are calculated for homogeneous groups as the number of lapses in a period over the total exposure for that period. The crude rates can be graduated using weighted least squares estimation or maximum likelihood estimation with a normal approximation to the likelihood function (Verrall, 1997).

### 2.3 Statistical models

The most common statistical models used for estimating withdrawal rates are various types of regression models.

### 2.3.1 Linear multiple-regression models

Linear multiple-regression allows numerous factors potentially affecting the withdrawal rate to be included in the model. Excluding error terms, the regression model is of the form

$$w_t = \beta_0 + \beta_1 V_1 + \dots + \beta_n V_n$$

where  $w_t$  is the withdrawal rate,  $\{\beta_i; i = 1, \dots, n\}$  are the coefficients to be estimated and  $\{V_i; i = 1, \dots, n\}$  are the explanatory variables. (Mauer and Holden, 2007) use a multiple linear regression model. The model uses five explanatory variables representing financial stress of the insurer, the price of insurance, a factor allowing for varying relative book compositions between term and whole-life insurance, the size of the insurer and the relative profitability of life insurance relative to other aspects of the business. This particular approach applied the model to estimate the lapse rate for the whole company rather than particular categories of policyholders within a single class of business, which is the more common practical application. However, such a regression model (with appropriate explanatory variables) could just as well be used within a single class of business. The estimation technique proposed by White (1980) is used, which provides a covariance matrix estimator of the ordinary least squares coefficients which is consistent even when the error term is conditionally or unconditionally heteroscedastic. Limitations of the linear multiple-regression model include:

- non-linear relationships between the explanatory variables and response variables must be explicitly specified;
- it is possible for the model to predict  $\hat{w}_t < 0$  or  $\hat{w}_t > 1$ , which is not allowed in terms of the definition of  $\hat{w}_t$ ;
- homoscedasticity (i.e. constant variance of the error terms) is assumed to hold;
- the error terms are assumed to be normally distributed.

### 2.3.2 Generalised linear models

Generalised linear models have been applied successfully to model many kinds of insurance data (de Jong and Heller, 2008). Generalised linear models allow for more complex relationships between the explanatory variables and the dependent variable to be modelled compared to linear multiple-regression models because they allow for, *inter alia*, non-normal and categorical response variables to be modelled. A variety of different link functions can be employed in the construction of a generalised linear model. Kim (2005) demonstrates the use of logistic, complementary log-log and arctan regression models to model withdrawal rates using economic variables and insurance contract variables. He concludes that the logistic and complementary log-log models are the best under a wide range of scenarios. It should be noted that

logistic models and complementary log-log models are generalised linear models with a binomial response where the respective link functions for the two are the logit link function and complementary log-log link function. Kim (2005) utilises a range of explanatory variables including interest rates, unemployment rates, economic growth rates, financial crisis indicator, policy duration and seasonal indicators. Excluding the error term, the logistic regression equation is of the form:

$$\log \frac{w_t}{1 - w_t} = \beta_0 + \beta_1 V_1 + \cdots + \beta_n V_n,$$

and the complementary log-log regression model is of the form

$$\log(-\log(1 - w_t)) = \beta_0 + \beta_1 V_1 + \cdots + \beta_n V_n.$$

The models were shown to be fairly accurate in predicting withdrawal rates (tested retrospectively), especially where the rates are generally stable over time.

The main advantages of logistic regression models and complementary log-log models include:

- they allow non-linear relationships between the explanatory variables and response variable to be modelled;
- some of the limiting assumptions of linear regression do not need to hold, namely homoscedasticity and normality of the error distribution;
- it guarantees that the response variable range constraints are met, i.e. that  $0 \leq w_t \leq 1$ .

## 2.4 Financial-economic models

If an insurance contract offers a surrender value, this represents a financial option to the policyholder and can be valued. De Giovanni (2008) presents a model for valuing such options. In order to value these, a hazard function describing the probability of a policyholder lapsing at time  $t$  is required. The hazard function is derived from a combination of expected rational consumer behaviour in the face of various financial choices (i.e. should the policyholder lapse at time  $t$  in order to maximise his expected financial gain from the contract) and irrationality on the behalf of the consumer, with the interplay between rationality and irrationality having a random element. The rational choice to lapse depends on the contract value which may depend on such factors as interest rates. Such models give rise to partial differential equations which can then be solved to give the value of the surrender option. The hazard function can be used to estimate withdrawal probabilities, but it can be quite complex to model the hazard function as a function of multiple variables and this may not be suitable for most practical modelling applications.

## Chapter 3

---

# Proposed alternative methods for estimating withdrawal rates

---

### 3.1 Introduction

A number of alternative methods have been proposed for modelling insurance risks, of which withdrawal risk is one. For example, Shapiro (2002) looks at using neural networks, fuzzy logic and genetic algorithms, Kitchens (2009) uses neural networks and Barrière et al. (2009) use Bayesian networks. Two of the more notable methods are Bayesian networks and neural networks. Both are capable of modelling non-linear interactions between various factors affecting the dependent variable, in this case withdrawal. Neural networks can be said to be a "black-box" approach in that the relationships being modelled do not necessarily have any natural interpretation; neural networks have however shown success in a wide range of applications (Palade and Jain, 2005). Bayesian networks are, however, capable of representing the causal structure in a system and thus may have more meaning to users of the model. In addition Bayesian network Theory allows for causal structure to be "learned" from data, i.e. the most likely causal structure giving rise to the observations is inferred from the data. This chapter proceeds by describing the theory needed to build a Bayesian network Model and a Neural network model. In Chapter 4 the methods are applied to build models used to estimate withdrawal rates for a book of 52137 policies. In addition a logistic regression model, which has already been used to model withdrawal rates in the literature, is built in order to make comparisons between the models. The comparison between the models and the methods in general is made in Chapter 5.

## 3.2 Bayesian networks

### 3.2.1 Bayesian network definition

Much of the following theory is based on (Korb and Nicholson, 2003). A Bayesian network is a graphical model for reasoning under uncertainty, where the nodes represent variables (discrete or continuous) and the arcs represent direct connections between them. These direct connections are often causal connections. In addition, BNs model the quantitative strength of the connections between random variables, allowing probabilistic beliefs about them to be updated automatically as new information becomes available.

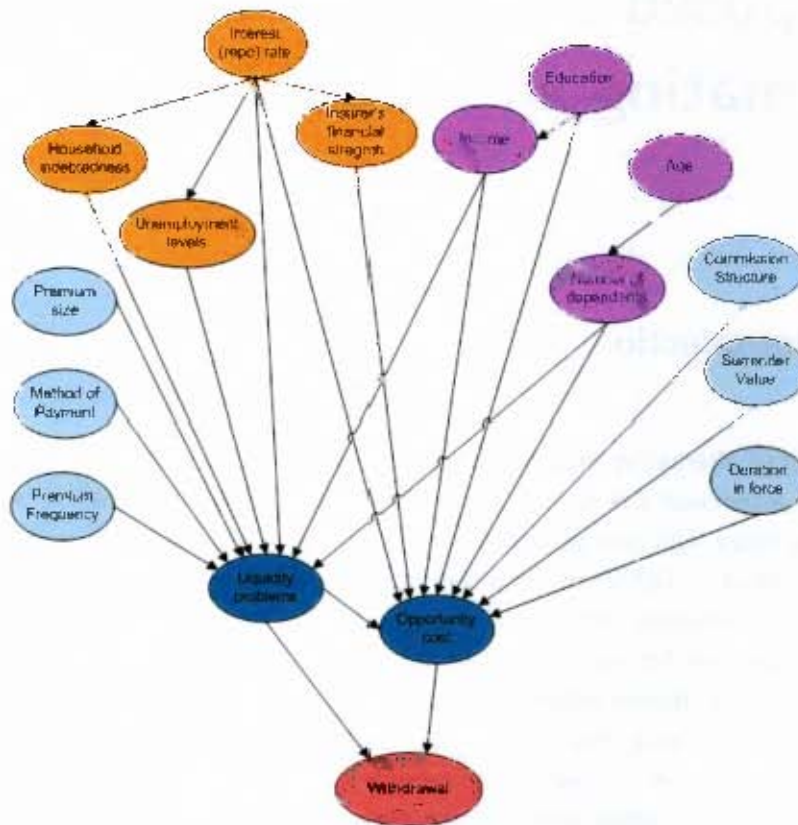


Figure 3.1: Example of a Bayesian network pertaining to withdrawal rates.

The theory of Bayesian networks (BNs) has existed for some time; Judea Pearl first used the term Bayesian network in 1985 (Pearl, 1985). Very few examples of their application in insurance work exist. One successful application is the use of BNs to model operational risk in insurance companies (Cowell et al., 2007). A brief introduction to BNs is therefore now given, before a specific application to estimating withdrawal rates is shown.

The Bayesian network in Figure 3.1 has been constructed using a subset of vari-

LP	OC	$P(w_s = Withdrawal LP, OC)$	$P(w_s = NoWithdrawal LP, OC)$
severe	yes	0.9	0.1
severe	no	0.5	0.5
mild	yes	0.7	0.3
mild	no	0.3	0.7
non-existent	yes	0.5	0.5
non-existent	no	0.05	0.95

Table 3.1: Conditional probability table for node **Withdrawal**

ables listed in section 1.3. All the factors impact on two main nodes, namely *liquidity problems* and *opportunity cost*. These nodes respectively refer to the degree of financial liquidity problems experienced by the policyholder and the perceived opportunity cost of continuing to pay insurance premiums for the policy as opposed to investing the money elsewhere. These two nodes are latent nodes in that they cannot be directly observed but are clearly two important, if not the most important, decision factors in determining whether a policyholder will lapse. Note that the "emergency fund hypothesis" and the "interest rate hypothesis" are commonly held hypotheses that refer respectively to the consumer decision processes represented by lapsing because of a lack of liquidity and lapsing because of the opportunity cost of holding the insurance contract (Weiyu et al., 2003). Thus all the other factors in the Bayesian network impact on these two decision factors which in turn impact on the probability of lapse.

Each node in a Bayesian network has a conditional probability table (CPT) associated with it. The probability of observing an occurrence of the variable represented by the node can be calculated given the values of its parent nodes and the CPT. Root nodes have a CPT with prior probabilities only. Further the nodes can be discretised in any way desirable. For example, with reference to the illustrative Bayesian network given in Figure 3.1, the node *Liquidity Problems* may be discretised to take on the values from {**severe, mild, non-existent**} while *Opportunity Cost* may take values from {**yes, no**}. The CPT of **Probability Of Withdrawal** for example can be seen in table 3.1.

A particular value in the joint distribution of the Bayesian network is represented by:

$$\begin{aligned}
 P(x_1, x_2, \dots, x_n) &= P(x_1) \times P(x_2|x_1) \times \dots \times P(x_n|x_1, \dots, x_{n-1}) \\
 &= \prod_{i=1}^n P(x_i|x_1, \dots, x_{i-1})
 \end{aligned}$$

If an ordering is imposed on the variables such that that  $Parents(x_i) \subseteq \{x_1, \dots, x_{i-1}\}$  where  $Parents(x_i)$  is the set parent nodes of node  $x_i$  then

$$P(x_1, x_2, \dots, x_n) = \prod_i P(x_i|Parents(x_i))$$

In order to perform inference in a Bayesian network it is necessary for the Markov property to hold: no direct dependencies in the system being modelled exist which are not already explicitly shown via arcs. Bayesian networks which exhibit the Markov property explicitly express conditional independencies. Note that conditional independence of  $X$  and  $Y$  given  $Z$  is denoted by  $X \perp\!\!\!\perp Y|Z$ . For example, in Figure 3.1, *Withdrawal*  $\perp\!\!\!\perp$  *duration in force* | *Opportunity cost*.

### 3.2.2 Reasoning in a Bayesian network

Reasoning in a Bayesian network is the process of updating the posterior probability of certain nodes (query nodes), given new information about other nodes (evidence nodes). The process is also termed "belief updating" as one is said to be updating one's belief about the likelihood of occurrence of a value for a particular node. For example, say we have a very simple two node BN represented by  $X \rightarrow Y$ , if there is evidence about a parent node, say  $X = x$ , then the posterior probability (or belief) for  $Y$ , denoted  $Bel(Y)$ , is  $P(Y|X = x)$  and can be read straight from the CPT (Korb and Nicholson, 2003).

If there is evidence about the child node,  $Y = y$ , then

$$\begin{aligned} Bel(X = x) &= P(X = x|Y = y) \\ &= \frac{P(Y = y|X = x)P(X = x)}{P(Y = y)} \end{aligned}$$

Say now we have three nodes in a chain, such that  $X \rightarrow Y \rightarrow Z$ . If we have evidence that  $X = x$  and we want to calculate  $Bel(Z)$  then

$$Bel(Z) = P(Z|X = x) = \sum_y P(Z|Y = y)P(Y = y|X = x).$$

If we have evidence about the leaf node,  $Z = z$  and we want to calculate  $Bel(X)$ , then

$$\begin{aligned} Bel(X = x) &= P(X = x|Z = z) \\ &= \frac{P(Z = z|X = x)P(X = x)}{P(Z = z)} \\ &= \frac{\sum_y P(Z = z|Y = y, X = x)P(Y = y|X = x)P(X = x)}{P(Z = z)} \\ &= \frac{\sum_y P(Z = z|Y = y)P(Y = y|X = x)P(X = x)}{P(Z = z)} \quad (Z \perp\!\!\!\perp X|Y) \end{aligned}$$

The calculation of posterior probabilities for certain nodes given evidence about other nodes is termed inference in a Bayesian network and is also known as belief updating. Inference can be exact using specific algorithms or approximate. Where

the Bayesian network is a simple tree structure or "poly-tree" with at most one path between any pair of nodes, then Pearl's message-passing algorithm can be applied (Pearl, 1988). If a Bayesian network is multiply-connected, i.e. if more than one path exists between at least one pair of nodes in the underlying directed graph, then a clustering algorithm (such as the junction tree algorithm) can be applied in order to cluster certain nodes together and transform the BN into a poly-tree to which the message passing algorithm can be applied (Korb and Nicholson, 2003). For larger networks, exact algorithms become infeasible (exact inference in Bayesian networks is in fact NP-hard (Cooper, 1990)). To perform inference in such larger networks, approximate methods such as using stochastic sampling are required. Stochastic simulation, simply, simulates values for the root nodes based on the prior distributions and propagates the beliefs through the network multiple times in order to arrive at an updated sample posterior distribution for each of the nodes in the BN. Different sampling methods can be used. The logic sampling method simply keeps a count of all the instantiations or occurrences of values of nodes within the network and then computes the sample posterior distribution as the observed occurrences divided by the total occurrences. This can have a slow convergence if certain evidence being conditioned upon is quite unlikely to occur. Weighting the occurrences by the likelihood of the evidence helps overcome this problem. Korb and Nicholson (2003) provide pseudocode descriptions of both exact and approximate inference algorithms.

Before inference can be performed with a network, it is necessary to estimate the BN CPT parameters. Estimation of parameters will now be explained.

### 3.2.3 Estimating parameters in a Bayesian network

Different methods for learning parameters exist for the cases where data is complete (i.e. no missing values) and incomplete (i.e. missing values). Where data is complete, the Spiegelhalter and Lauritzen method can be used. This is a fairly simple method and makes use of the Dirichlet family of distributions. For each node in the Bayesian network and for each instantiation of a node's parents, an initial Dirichlet distribution with  $\tau$  states is assumed, i.e.  $D[\alpha_1, \dots, \alpha_i, \dots, \alpha_\tau]$  and the probability of being in state  $i$  is given by

$$P(X = i) = \frac{\alpha_i}{\sum_{j=1}^{\tau} \alpha_j}.$$

Assuming that each state is independent of the probability of every other state we can update the nodes' probability distributions. Note that  $\sum_{j=1}^{\tau} \alpha_j$  is said to be the equivalent sample size and the larger it is initially the slower the parameters are updated. The algorithm then proceeds to update the distribution for all the observed values. Thus for a particular instantiation of a node's parents, if state  $i$  is observed the distribution is updated to  $D[\alpha_1, \dots, \alpha_i + 1, \dots, \alpha_\tau]$ .

In reality, data is often incomplete and thus other tools must be used for parameter estimation. An exact method for parameter estimation is directly maximising

the conditional density  $P(\theta|\mathbf{Y})$ ; however this quickly becomes computationally intractable and is usually abandoned in favour of approximate methods (Korb and Nicholson, 2003). Deterministic and stochastic approximate methods exist for estimating parameters from incomplete data, but they both make the strong simplifying assumption that the missing data are independent of observed data. A deterministic method is the well-known Expectation Maximisation (EM) algorithm. The EM algorithm produces a maximum likelihood estimate  $\hat{\theta}$  of  $\theta$  which maximises  $P(e|\theta)$  where  $e$  is the evidence available. Gibbs sampling is a commonly used stochastic approach to estimating parameters. Korb and Nicholson (2003) provide descriptions of the algorithms for both of these methods.

### 3.2.4 Learning causal structure in a Bayesian network

Methods for learning Bayesian network structures from data fall into two main categories, namely constraint-based learning methods and metric learners. Metric learners select the network structure which maximises a certain metric. For example, a metric learner will select a hypothesised graph structure  $h_i$  given evidence  $e$  which maximises  $P(h_i|e)$ . Other noteworthy metrics include Minimum Description Length (MDL) and Minimum Message Length (MML). Constraint-based learners rely on Reichenback's "Principle of the Common Cause", which suggests that conditional dependencies and independencies arise from causal structure and therefore that an inverse inference from observations of dependency to causality is possible, to identify conditional independencies and infer causal structure. Conditional independencies are identified through vanishing partial correlations (on the assumption of normality), i.e.  $X \perp\!\!\!\perp Y|\mathbf{S}$  is inferred from  $\rho_{XY|\mathbf{S}} = 0$  (no correlation remains between  $X$  and  $Y$  when the set  $\mathbf{S}$  is held constant.) Constraint based learners also allow restrictions to be imposed on the discovery process. Constraints can be enforced through a number of mechanisms such as specifying arc directions, forbidden arcs and knowledge tiers (i.e. grouping of values into tiers, the order of which implies a temporal causal ordering). The best-known constraint based algorithm is the PC algorithm, see Spirtes et al. (2001). It has been suggested that metric learners are more robust and powerful than constraint-based learners; however constraint-based learners are easier to implement and are available in most Bayesian network software (Korb and Nicholson, 2003).

## 3.3 Neural Networks

### 3.3.1 Neural network definition

Neural networks (more correctly termed artificial neural networks to distinguish them from biological neural networks) are non-linear statistical modelling tools capable of learning and adapting. Various definitions exist for them depending on the context bias. Haykin (1998) gives the following description: "*A neural network is a massively*

*parallel distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:*

1. *Knowledge is acquired by the network from its environment through a learning process*
2. *Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge."*

Different learning paradigms exist for neural networks, the main paradigms being supervised and unsupervised learning. In supervised learning, a "teacher" is said to have knowledge of the environment, with that knowledge being represented by a set of input-output examples (Haykin, 1998). The neural network does not have knowledge of the environment and needs to train itself to give the desired output for a given input. Thus, given a set of input data  $x \in X$  and output data  $y \in Y$ , the goal is to find a function  $f : X \rightarrow Y$  that most accurately maps the input to the output. The training process is an iterative one that attempts to reduce the error in predicting output (i.e. minimise a cost function related to how much the mapped output data differs from the actual output data). With unsupervised learning, there are no labeled examples of functions to be learned by the network. For the application of modelling withdrawal rates we are interested in supervised learning since we have a set of inputs (e.g. interest rates, duration of policy, etc) and a set of outputs (i.e. withdrawal rates). We will therefore not pursue unsupervised learning any further.

The most common type of neural network performing supervised learning is the multilayer perceptron (MLP) neural network using the Error Back-Propagation Algorithm for training. A MLP NN can be seen in Figure 3.2. An MLP network has at least three layers of neurons: an input layer, an output layer and one or more hidden layers. An MLP forms a feedforward network where information flows from the inputs through the hidden layers to determine the output. However, during training, information must flow backwards through the network (this will be described in section 3.3.2 which deals with training). Assuming the network is already trained, information flows through the network as follows:

1. The values of the  $p$  input nodes  $\mathbf{x} = (x_1, \dots, x_p)$  are propagated through the network to each of the  $L$  neurons in the first hidden layer.  $\mathbf{v}^1 = (v_1^1, \dots, v_L^1)$  is calculated using weighted sums of the input neuron values where the weights are the synaptic weights  $\mathbf{W}$ .

$$v_j^1 = \sum_{i=p} w_{ji} x_i$$

2.  $\mathbf{v}$  is then fed into an activation function,  $\varphi$ , to determine the output,  $\mathbf{y}$  from the neurons.

$$\mathbf{y} = \varphi(\mathbf{v}).$$

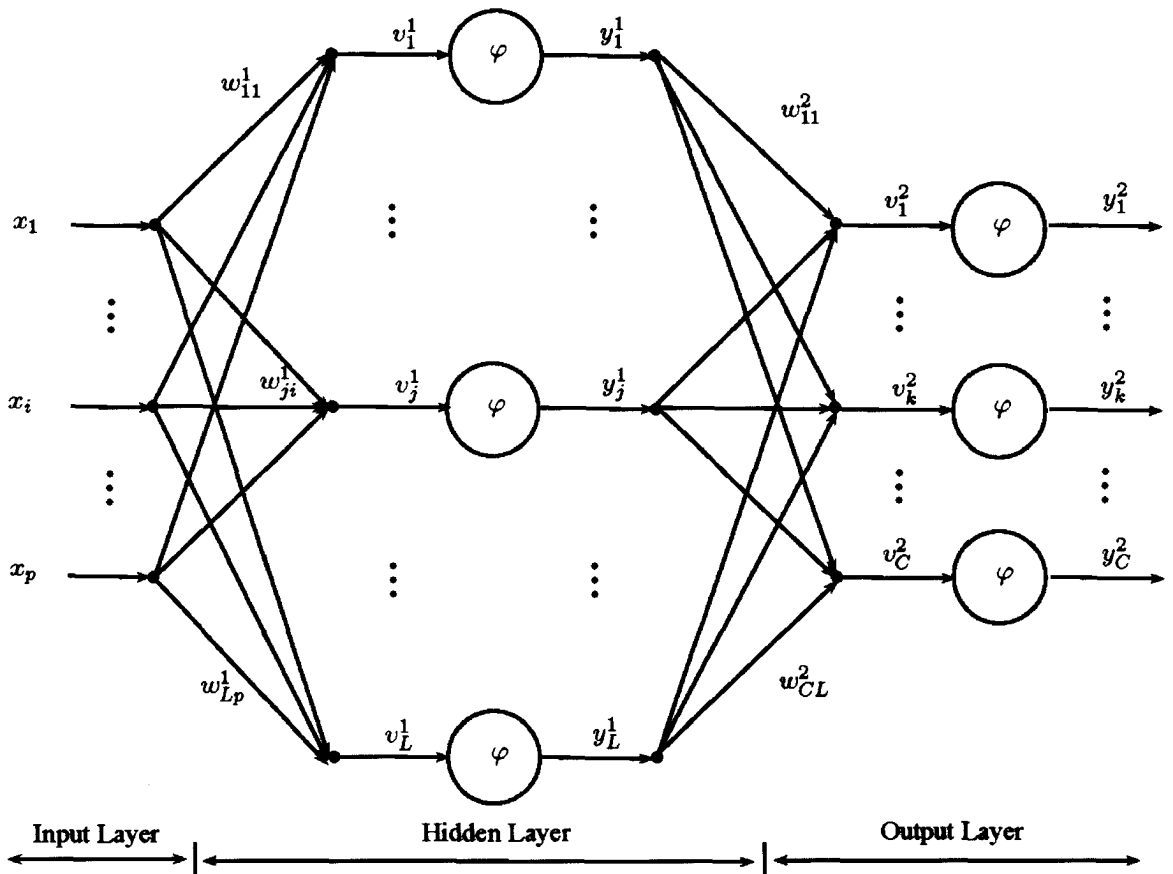


Figure 3.2: Multilayer perceptron neural network with 1 hidden layer

The activation function is a non-linear smooth (i.e. differentiable everywhere) function. Two commonly used activation functions are both sigmoidal functions. They are the logistic function and the hyperbolic tangent function, given respectively by:

$$y_i = \varphi(v_i) = (1 + e^{-v_i})^{-1}$$

and

$$y_i = \varphi(v_i) = \tanh(v_i).$$

3. The output from the neurons can then be used to calculate the inputs (as weighted sums) for the next layer of neurons which may either be another hidden layer or the output layer.

Note that the number of neurons in any layer need not be the same as in any other layer.

### 3.3.2 Training the neural network

Training consists of two passes through the NN. First a forward pass is performed to calculate the output values from the neurons as described above in section 3.3.1. Thereafter backpropagation of information through the network occurs, i.e. information flows from the output neurons via the hidden layer neurons back to the input neurons, with the synaptic weights being updated in the process according to an error correction rule. The algorithm used is called *error back-propagation*. The algorithm proceeds as follows:

For each training iteration  $n$ :

1. Calculate the error terms for the output neurons. For neuron  $j$ , this is given by

$$e_j(n) = d_j(n) - y_j(n),$$

where  $d_j(n)$  is the desired output or rather the output taken directly from the training data and  $y_j(n)$  is the output from the NN.

2. Calculate the term called the instantaneous total error energy, given by:

$$\xi(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n),$$

where the set  $C$  includes all the neurons in the output layer.

3. Calculate the average squared error energy

$$\xi_{av} = \frac{1}{N} \sum_{n=1}^N \xi(n)$$

where  $N$  is the total number of training examples. Note that one presentation of all the items in the training set to the NN is called an epoch.

4. Calculate the local gradient term  $\delta_j(n)$ . This differs depending on whether neuron  $j$  is an output neuron or a hidden neuron. If neuron  $j$  is an output neuron then

$$\delta_j(n) = e_j(n) \varphi'_j(v_j(n)).$$

If neuron  $j$  is hidden, then

$$\delta_j(n) = \varphi'_j(v_j(n)) \sum_k \delta_k(n) w_{kj}(n)$$

where  $k$  represents all the neurons in the layer in front of neuron  $j$ .

5. Select a value for the learning parameter  $\eta$  and learning momentum parameter  $\alpha$  where  $0 \leq |\alpha| \leq 1$ . If  $\eta$  is too small, the rate of learning will be very slow, but

if it is too large then the network may become unstable (oscillatory) (Haykin, 1998). Use these parameters to calculate the weight adjustment term

$$\Delta w_{ij}(n) = \eta \sum_{t=0}^n \alpha^{n-t} \delta_j(t) y_i(t).$$

The process may be repeated for a number of epochs. There are a number of stopping criteria that can be used to determine when to terminate the training process. An example of one such stopping criterion is when the rate of change in the total average error energy over the last few iterations declines to a suitably small margin. Care must be taken however to ensure that there is not convergence to a sub-optimal local minimum.

## Chapter 4

---

# Application of proposed methods

---

### 4.1 Description of data

The data used in the application comes from a book of 52137 insurance contracts held by a South African Life insurer over a 4 year period. The contracts are of a variety of different product types. All contracts contain a death-benefit element and a subset contain an additional investment element. The database contained a large number of variables of potential interest; however many of these had a large number of values missing. Bayesian networks handle missing values with relative ease (Korb and Nicholson, 2003); however it is more difficult to deal with missing values when using a neural network or a logistic regression model. Consequently, variables with a large proportion of missing instances (more than 50%) were excluded from the analysis. An enquiry into how robust the Bayesian network is at handling data with missing values is not conducted, but could be a point of further research. The variables used are:

- Characteristics of the policyholder
  - Age
  - Sex
- Characteristics of the insurance contract
  - Payment method
  - Payment frequency
  - Premium amount
  - Whether contract has an investment portion
  - Original contract term

- Contract cash value (surrender value)
- Duration in-force
- Commission distribution method
- Macro factors
  - Prime interest rate

The period of time over which a withdrawal "rate" is analysed is one year. Bayesian networks have traditionally been developed to deal mainly with discrete categorical data, although they can be extended to deal with continuous data, see Cobb et al. (2007). In order to simplify the implementation of a Bayesian network, continuous variables are discretised. Discretisation is done in an attempt to spread the data counts fairly evenly amongst the categories, while simultaneously using heuristics to take into account the nature of the variables and their likely impact on withdrawal rates. For example, policyholders are more likely to lapse at early durations when they have less to lose by lapsing; therefore we group early durations by short intervals (1 year) and later durations in increasingly large intervals (5 years, 10 years, etc). The following discretisations are applied to the data used in the Bayesian network:

- Age:  $[0, 25)$ ,  $[25, 45)$ ,  $[45, 65)$ ,  $[65, \infty)$
- Premium amount:  $[0, 150)$ ,  $[150, 250)$ ,  $[250, \infty)$
- Benefit term:  $[0, 10)$ ,  $[10, 20)$ ,  $[20, 30)$ ,  $[30, \infty)$
- Contract cash value:  $[-\infty, 0)$ ,  $[0, 1000)$ ,  $[1000, 5000)$ ,  $[5000, \infty)$  (note that a negative cash value represents a loan to the policyholder)
- Duration:  $[0, 1)$ ,  $[1, 2)$ ,  $[2, 3)$ ,  $[3, 4)$ ,  $[4, 5)$ ,  $[5, 10)$ ,  $[10, 20)$ ,  $[20, \infty)$
- Interest rate:  $(6.5, 7.5]$ ,  $(7.5, 8.5]$ ,  $(8.5, 11.5]$ ,  $(11.5, 13.5]$ ,  $(13.5, 15.5]$

Neural networks are designed to work with continuous numerical data and the mapping of discrete categorical values to numerical values may impose a spurious ordering of the categorical data. To deal with this problem, the categories of categorical variables are translated to new binary variables. For example, the variable *Payment method* with values "Cash", "Debit-order" and "Stop-order" is mapped to 3 new variables *Cash*, *DebitOrder* and *StopOrder* which are all binary and  $Cash_i + DebitOrder_i + StopOrder_i = 1$  where  $i$  is the index of the data tuple.

## 4.2 Bayesian network

A causal structure is first extracted from the data using a constraint-based learning algorithm. The learning of the causal structure is described in section 4.2.1. A

separate model is then proposed which is designed based on domain knowledge. This is described in section 4.2.2. The software Tetrad IV was used for all aspects of the Bayesian network implementation. The software is open-source and was augmented to provide the functionality needed to perform cross-validation described in section 5.

### 4.2.1 Learned Bayesian network

The CPC algorithm is a variant of the PC algorithm which improves arc orientation accuracy. A constraint-based learning algorithm was selected over a metric learner for practical ease-of-use reasons. Constraints are put in place to ensure the direction of the causality is inferred correctly; for example age may have an impact on withdrawal rates, but clearly withdrawal rates cannot have an impact on age! Knowledge tiers are imposed for the withdrawal rates model and are as follows:

1. *< Age, Gender, ReserveBankReporate >*
2. *< ContractBenefitTerm, ContractCashValue, ContractCommissionDistributionMethod, ContractGroupName, ContractHasInvestmentAccount, ContractPaymentMethod, ContractPaymentRate, PaymentsPerYear >*
3. *< DurationInForce >*
4. *< WithdrawalValue >*

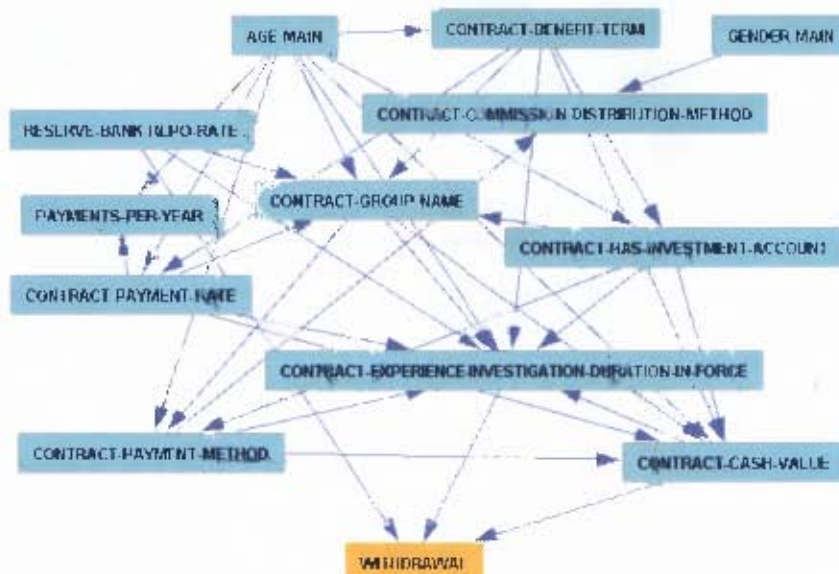


Figure 4.1: Bayesian network graph learned from data.

The Bayesian network graph structure extracted by the CPC algorithm can be seen in Figure 4.1. It is interesting to note that only three main factors are determined to affect withdrawal rates directly, namely the *ReserveBankRepoRate*, *DurationInForce* and *ContractCashValue*. All the other factors affect withdrawal rates indirectly through the aforementioned three factors. It may be argued then for simplicity's sake that a moderately complex model like a Bayesian network is not needed to accurately model withdrawal rates as only a limited number of factors may affect withdrawal rates directly. However, the actual factors may change depending on the nature of the business and in some instances the values of the key factors may not be known but probabilistically inferred through the causal structure. The relationships extracted also give insight into the business. For example *ContractCommissionDistributionMethod* is a factor which *prima facie* may be a significant determinant of withdrawal rates due to agents irresponsibly encouraging policyholders to withdraw from their policies and take out new policies in order to generate more upfront commission, but in fact has very little impact on withdrawals.

#### 4.2.2 Proposed Bayesian network

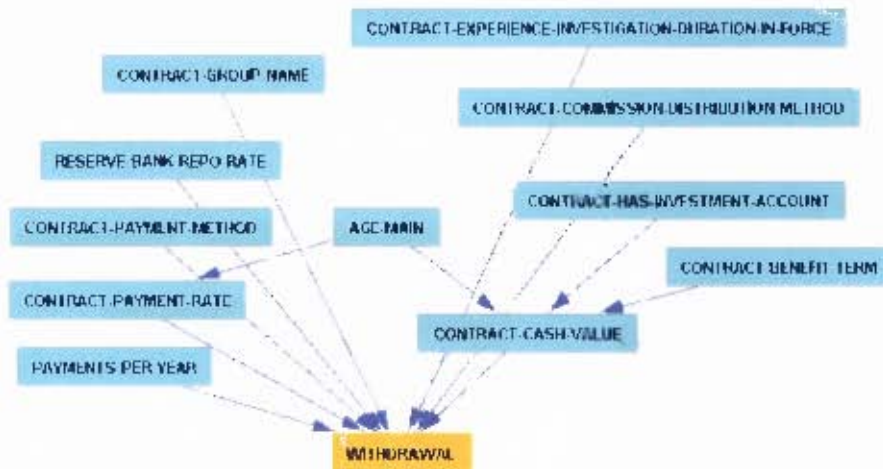


Figure 4.2: Proposed Bayesian network.

A simpler Bayesian network, which can be seen in Figure 4.2, is proposed (designed using domain knowledge) in order to compare it to the learned graph. The graph only has three nodes with parents and thus only three conditional probability tables need to be calculated (excluding prior distributions for other nodes). This makes the Bayesian network much quicker to calibrate and use for prediction purposes. The contrast between the proposed and learned Bayesian networks can help

determine whether the search algorithm used to learn the causal structure is able to extract a more accurate causal structure with better predictive power and robustness under changing conditions. Latent variables are excluded from the analysis due to practical problems involved in calibrating the parameters of a model with latent variables to data.

### 4.3 Neural network

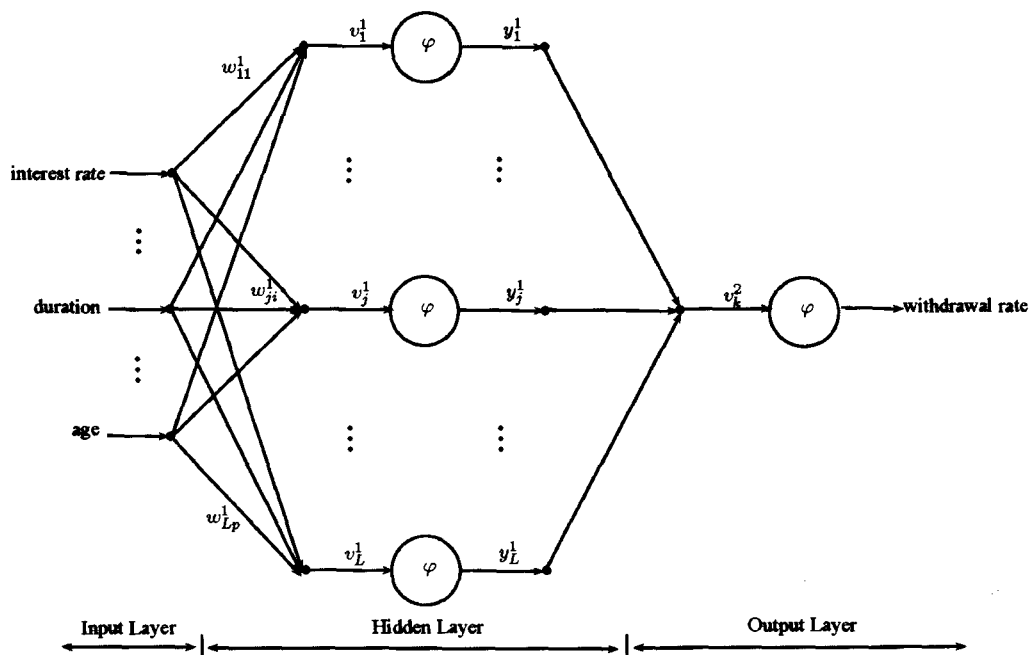


Figure 4.3: MLP NN for predicting withdrawal rates. Only a subset of the inputs are shown.

A multi-layer perceptron neural network with 1 hidden layer is used to model the withdrawal experience. The neural network can be seen in Figure 4.3. It was implemented in the open-source software package **R** (R Development Core Team, 2009) using the 'neural' package (Nagy, 2007). The input nodes are:

- *Age*
- *Male*
- *Female*
- *PaymentMethodStopOrder*
- *PaymentMethodDebitOrder*

- *PaymentMethodCash*
- *ContractPaymentRate* (Premium amount)
- *ContractHasInvestmentAccount*
- *ContractTerm*
- *ContractCashValue*
- *DurationInForce*
- *UpfrontCommission*
- *AsAndWhenCommission*
- *ReserveBankRepoRate* (interest rate)

The hidden layer is configured to utilise 30 neurons. The activation function  $y_i = \varphi(v_i) = (1 + e^{-v_i})^{-1}$  is used since the function produces an output value  $y_i$  such that  $0 \leq y_i \leq 1$  as is required for a probability value. The learning rate parameter  $\alpha$  is set as  $\alpha = 0.2$  and the training algorithm runs in sequential mode of back-propagation as opposed to batch-mode. The training of a neural network and subsequent accuracy of the trained neural network can vary significantly with the aforementioned parameters. It is possible to implement a formal strategy for optimising the values of these parameters; genetic algorithms in particular have been employed successfully as demonstrated by Ribert et al. (2006). Sukthomya and Tannock (2005) use Taguchi's design of experiments approach to optimise the parameters of a neural network. Varying the parameters by trial and error revealed that the performance of our network is not significantly affected by the parameters and thus pseudo-optimal values were allocated accordingly.

#### 4.4 Logistic regression model

A logistic regression model was built using backwards stepwise regression based on AIC. Thus a full logistic regression model is initially built with all the available variables and then variables that are not significant are iteratively removed in order to improve the AIC scoring of the model. The logit link function is used and is given by

$$g(w_t) = \log \frac{w_t}{1 - w_t}.$$

This results in the regression equation given by:

$$g(w_t) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k$$

and therefore the modelled withdrawal rate  $w_t$  is given by:

$$w_t = \frac{1}{1 + e^{-g(w_t)}}.$$

The variables selected through backwards stepwise regression on the full data set can be seen in Table 4.1 (note: the variables are organised into factor and non-factor variables, where a factor variable is categorical in nature).

Non-factor variables	factor variables
<i>PaymentsPerYear</i>	<i>ContractCommissionDistributionMethod</i>
<i>ContractCashValue</i>	<i>ContractGroupName</i>
<i>ContractBenefitTerm</i>	<i>ContractPaymentMethod</i>
<i>ReserveBankRepoRate</i>	<i>ContractHasInvestmentAccount</i>
<i>DurationInForce</i>	

Table 4.1: Variables selected through backwards stepwise regression on full data set.

The logistic regression model was implemented in R (R Development Core Team, 2009).

## Chapter 5

---

# Comparison of methods

---

A number of key factors are important in determining a method's suitability for the task at hand. In section 5.1 we compare and contrast the various methods using a number of criteria before we quantitatively compare the models' predictive abilities in section 5.2.

### 5.1 Qualitative comparison of methods

There are both theoretical and practical factors to take into consideration when selecting a method for modelling withdrawal rates.

#### 5.1.1 Theoretical considerations in comparison of methods

Multicollinearity, i.e. correlation between the explanatory variables, can cause problems with logistic regression in that it can result in incorrect signs and magnitudes of regression coefficient estimates. Bayesian networks search algorithms use correlations between variables to search for causal structure and then explicitly model the dependencies between variables through conditional probability tables. Further, Bayesian networks can help the modeller to understand the relationships that give rise to the correlations and thus have a better understanding of the system being modelled. Carpio and Hermosilla (2002) have shown that neural networks are significantly better at dealing with multicollinearity than logistic regression models.

If the modelling of withdrawal rates has the dual purpose of prediction and inference then Bayesian networks and logistic regression emerge as being preferable to neural networks. All three methods can be used for prediction, however a neural network is a "black-box" in that it is extremely difficult to make any inferences about the system being modelled from the resulting model. In contrast, logistic regression

models give regression coefficients for each of the variables indicating the magnitude of their impact on the response variable as well statistical significance results. Bayesian networks similarly provide a graphical result of the causal relationships in the system being modelled as well the impact of the explanatory variables on the response variable through the conditional probability tables. In addition Bayesian networks can be used to perform "what-if" scenario analysis given evidence about a subset of the explanatory variables.

### 5.1.2 Practical considerations in comparison of methods

When one is working on large data sets, the computational run-time required to build and calibrate the models and then subsequently use them for prediction can become an important consideration. The time required to build, calibrate and predict using the three compared methods on the same size data set is given in section 5.2. This is only an empirical example that illustrates the more general observation that Bayesian networks and neural networks take longer to run than logistic regression; however the run-time of Bayesian networks and neural networks can vary significantly when varying standard parameters. Zhang and Bivens (2007) compare the response times for a Bayesian network and a neural network in a particular application and find that Neural networks are noticeably quicker to run than Bayesian networks, particularly as the data set size increases.

All methods can be modified to handle missing values. See (Sharpe and Solly, 1995) and (Viharos et al., 2002) for neural networks, (Fung and Wrobel, 1989) for logistic regression and (Korb and Nicholson, 2003) for Bayesian networks. However, the ease and success with which the missing values are dealt with by the various methods vary and should be considered in the light of data with many missing values.

Neural networks and logistic regression models deal naturally with continuous data under arbitrary distributions and are able to handle discrete variables. Bayesian networks work optimally with discrete data and thus require continuous data to be discretised (most Bayesian network methods involving continuous variables require the limiting assumption that the variables are normally distributed (Korb and Nicholson, 2003)). Deciding on the discretisations can be quite arbitrary and may require much additional work to decide on optimal discretisations.

## 5.2 Quantitative comparison of predictive ability

The problem of determining whether a contract holder will withdraw from the contract or not is essentially a classification problem and the three methods being compared are classification methods. Thus a model is used to determine the probability that a contract will be classified as "withdrawn" in a period. Receiver operating characteristics (ROC) graphs have been shown to be effective in evaluating and comparing classification methods (Fawcett, 2006). We therefore use ROC as a framework for

comparing the three methods in terms of their predictive ability and their robustness under changing conditions.

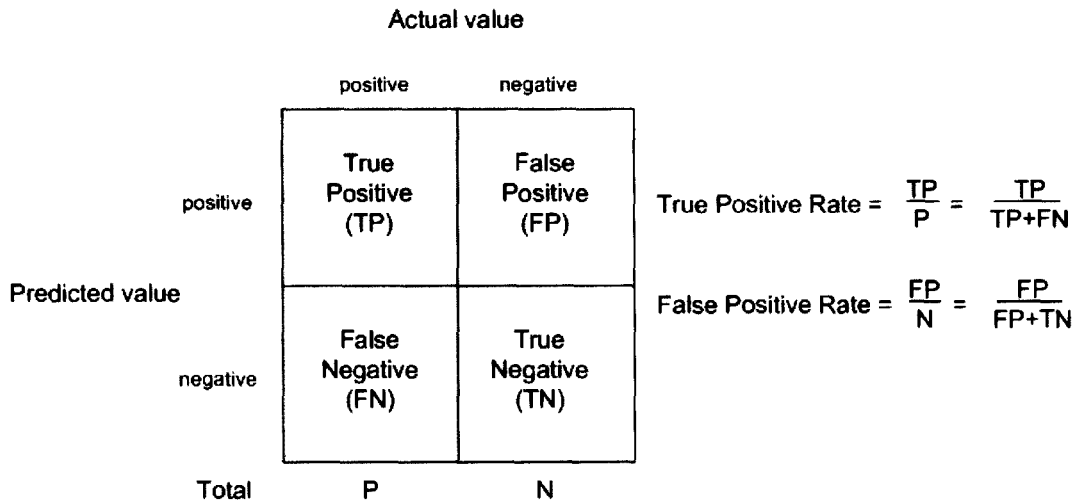


Figure 5.1: Confusion matrix used in ROC analysis

A *discrete* classifier gives a classification (e.g. either a policy will withdraw in the analysis period (Yes) or a policy will not withdraw (No)) as its output. A *scoring* classifier uses a score, or in many instances such as withdrawal rates, a probability and an associated threshold level to classify an instance. For example if the threshold for the probability of withdrawal for classifying an instance as a withdrawal is 0.5 then an instance that is deemed to have a probability of withdrawal of 0.65 is classified as "Withdrawal". In the instance where there are only two classes to which an instance can be classified a confusion matrix as seen in Figure 5.1 is applicable. Thus there are four possible outcomes of a classification, namely:

1. True positive (TP): an instance is classified "Yes" (e.g. "Withdrawal") when the actual outcome is "True" e.g. "Withdrawal".
2. True negative (TN): an instance is classified "No" (e.g. "No Withdrawal") when the actual outcome is "False" e.g. "No Withdrawal".
3. False positive (FP): an instance is classified "Yes" (e.g. "Withdrawal") when the actual outcome is "False" e.g. "No Withdrawal".
4. False negative (FN): an instance is classified "No" (e.g. "No Withdrawal") when the actual outcome is "True" e.g. "Withdrawal".

A ROC graph plots true positive rates on the y-axis versus false positive rates on the x-axis for a classifier. The classification output used to plot the graph arises from using the classifier trained on a training data set to classify instances of a training data set. Clearly a classifier which results in a 100% true positive classification and

a 0% false negative classification is optimal. This results in a point (0,1) on the ROC graph. Further, any point which results in classification which is to the "south-east" of the line  $y = x$  should be disregarded in favour of a classifier which simply inverts the classification (Fawcett, 2006). Plotting a ROC graph for discrete classifiers results from plotting a few fixed points from the classification output. For scoring classifiers, the threshold for classification is usually not known and thus a ROC curve is plotted which is a curve resulting from plotting all the points arising from altering the threshold rate between 0 and 1. ROC curves from the various models can be seen in Figure 5.3. Various scalar metrics derived from the confusion matrix can be used to evaluate and compare the methods. We utilise two metrics: area under curve (AUC) and root mean square error (RMSE). AUC is more commonly used to compare the predictive ability of models and has a sounder theoretical foundation for this purpose than RMSE; however it is useful to include another metric for further insight. AUC is derived from the ROC curve and is the area under the ROC curve. The higher the AUC value the more likely a classifier is to result in a higher percentage of true positive classifications and a lower for a given percentage of false negative classifications at any given threshold level. RMSE simply measures the mean of the square distance between the classification value (or probability of withdrawal in this case) and the actual value (indicated by a 0 for no withdrawal and a 1 for withdrawal). Thus the more accurate the classification the lower the RMSE.

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_i (y_i - \hat{y}_i)^2},$$

where  $y_i$  is the actual output value and  $\hat{y}_i$  is the predicted value.

We are also interested in how the various methods' predictive ability changes over time as conditions change. To test this we perform a cross-validation on the data, iteratively calculating AUC and RMSE on sliding windows (tranches) of subsets of the main data set. An illustrative diagram showing the segmentation of the data for three cross validations is shown in Figure 5.2.

By calculating AUC and RMSE for  $R$  tranches we can calculate average values and sample variances. The sample variance enables us to compare consistency in prediction accuracy over time between the methods. We define the mean AUC as:

$$\bar{\text{AUC}} = \frac{1}{R} \sum_{r=1}^R \text{AUC}_r$$

and the variance of AUC as:

$$\sigma_{auc}^2 = \frac{1}{R-1} \sum_{r=1}^R (\text{AUC}_r - \bar{\text{AUC}})^2.$$

A mean and variance is calculated similarly for RMSE. The ROC graphs created by combining the predictions from all the tranches together are shown in Figure 5.3.

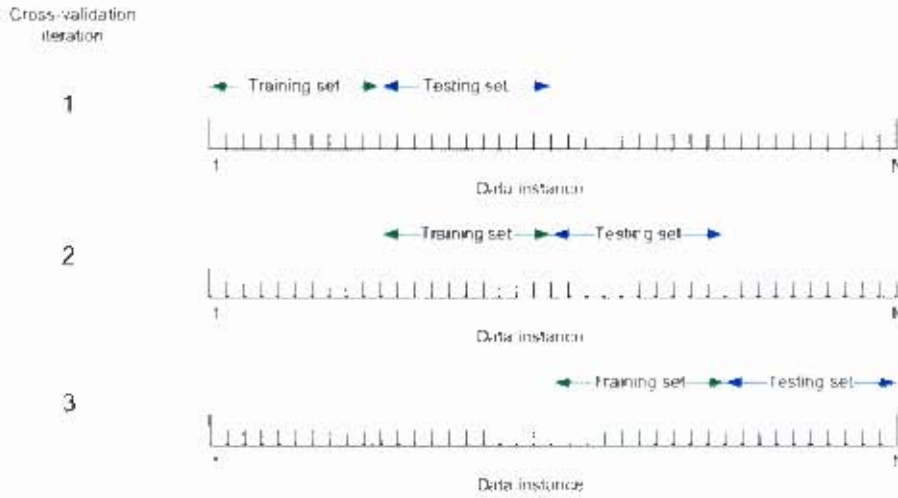


Figure 5.2: Example of segmentation of data into training and testing sets for cross-validation with 3 iterations.

The run-time taken to calibrate the models and subsequently use them for prediction is an important practical consideration. In our analysis we thus include the times taken to run the analyses on a machine with CPU Intel Core2 Duo 2.53GHz and 4GB RAM.

The results from using  $H = 10$  can be seen in table 5.1 and the ROC graphs can be seen in Figure 5.3.

	AUC	$\sigma_{AUC}^2$	RMSE	$\sigma_{RMSE}^2$	Run-time
Learned Bayesian network	0.561233	0.005514	0.357593	0.002652	2h46min
Proposed Bayesian network	0.672244	0.007874	0.356453	0.002978	56min
Neural network	0.593906	0.000672	0.350315	0.002613	18h55min
Logistic Regression Model	0.634338	0.010034	0.354240	0.002697	12min

Table 5.1: Table of results showing evaluation metrics

## Chapter 6

---

# Conclusions and recommendations

---

## 6.1 Conclusions

### 6.1.1 AUC preferred over RMSE as a metric for predictive ability

RMSE indicates that there is a very small difference between the models in terms of their predictive ability and stability over time. AUC, however, seems to indicate that there are significant differences. AUC is more commonly used than RMSE to assess a model's predictive ability and therefore inference is done using the results from the AUC analysis.

### 6.1.2 Proposed Bayesian network outperforms learned Bayesian network

The AUC for the proposed Bayesian network is 20% higher than for the learned Bayesian network indicating a significantly higher predictive ability. This combined with the nearly 3-fold reduction in run-time would indicate that the proposed Bayesian network is preferable to the learned network. The proposed Bayesian network does however have a 42% higher variance indicating that it is less stable in its predictive ability over time. In practice this may be overcome by analysing the factors leading to withdrawal and adjusting the model over time to reflect expected changes in reality. The learned Bayesian network is reflective of only a single algorithm's causal discovery ability and an investigation into the predictive ability of Bayesian networks extracted by a number of different algorithms would be needed before making firm conclusions.

### **6.1.3 Neural network has lower predictive ability than other models but is more stable over time**

The neural network generated a lower AUC score than the logistic regression model and proposed Bayesian network indicating a lower predictive ability than the two other models. The variance of the AUC was, however, significantly lower than that of the logistic regression model and proposed Bayesian network; 14.9 times and 11.7 times lower respectively. This indicates that there may be a tradeoff with the neural network between lower predictive ability for more robustness and stability in its predictive ability over time. The training time of the neural network was much higher than all the other models: 94.6 times longer than the logistic regression model, 6.8 times longer than the learned Bayesian network and 20.3 times longer than the proposed Bayesian network. This difference in time will be amplified on large data sets. Thus the neural network would be recommended where training time is not a constraint and where stability over time is desired.

### **6.1.4 Logistic regression model has good performance, excellent run-time, but is less stable over time**

The logistic regression model performs well in its predictive ability; coming second only to the proposed Bayesian network, although the variability of its AUC score is the highest of all the models. It does have the shortest training time which can be reduced even further through techniques such as least angle regression (Hastie et al., 2009).

### **6.1.5 Conclusions for model selection**

The neural network had lower predictive ability than the proposed Bayesian network and the logistic regression model, had significantly higher training and classification time and it is a "black box" approach that does not enable one to perform inference based on the selected model. Bayesian networks and logistic regression are thus recommended in preference to neural networks.

It is less clear whether to recommend a Bayesian network or a logistic regression model. Both methods are good for prediction and can be used for inference although a Bayesian network may reveal a deeper causal structure. A logistic regression model may however be significantly quicker to train and use for prediction. Choosing a method for modelling withdrawal rates should therefore be looked at in light of the modelling activity's constraints.

## **6.2 Recommendations**

### **6.2.1 Bayesian networks**

Continuous variables in the data were discretised by choosing intervals which resulted in a fairly even distribution of the counts for each category and then adjusting the number of categories and the intervals for heuristics based on domain knowledge. Further research may attempt to model the continuous variables as continuous models within a Bayesian network or may focus on formalising a framework for the discretisation of continuous variables, particularly in an insurance context.

Bayesian networks are able to deal with data that has missing values fairly easily. An analysis of just how robust they are as the proportion of missing data values increases may be a key decision factor in practice where data is of poor quality.

The CPC algorithm was used to search for a causal structure. There are numerous other causal discovery algorithms which could be compared and contrasted in their ability to detect causal structure in insurance data.

### **6.2.2 Neural networks**

A few of the parameters for the neural network were chosen through trial and error. Improvements could be made through a formal method, such as a genetic algorithm, for selecting values for these parameters.

### **6.2.3 Improving efficiency and investigating optimal recalibration frequency**

The time required to calibrate and subsequently use the Bayesian networks and Neural networks can become untenable for large datasets given limited computational resources. Specific features of the problems may be utilised to alter the algorithms to improve the efficiency. Multicore, multimachine and hardware implementations of the methods may be investigated. The marginal benefit in improved prediction accuracy from increasing the frequency at which the models are recalibrated may also be investigated.

---

# Bibliography

---

- O. Barrière, E. Lutton, and P. Wuillemin. Bayesian network structure learning using cooperative coevolution. In *GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 755–762, New York, NY, USA, 2009. ACM. [cited at p. 11]
- K.J.E. Carpio and A.Y. Hermosilla. On multicollinearity and artificial neural networks. *Complexity International*, 10, 2002. [cited at p. 29]
- B.R. Cobb, R. Rumi, and A. Salmeron. *Advances in Probabilistic Graphical Models*, volume 214/2007 of *Studies in Fuzziness and Soft Computing*. Springer Berlin, 2007. [cited at p. 22]
- G.F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405, 1990. [cited at p. 15]
- R. G. Cowell, R. J. Verrall, and Y. K. Yoon. Modeling operational risk with Bayesian networks. *Journal of Risk and Insurance*, 74(4):795–827, 2007. [cited at p. 12]
- D. De Giovanni. Lapse rate modeling: A rational expectation approach. *Scandinavian Actuarial Journal*, 2008. [cited at p. 9]
- P. de Jong and G. Heller. *Generalized Linear Models for Insurance Data*. International Series on Actuarial Science. Cambridge University Press, March 2008. [cited at p. 8]
- T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, June 2006. [cited at p. 30, 32]
- K.Y. Fung and B.A. Wrobel. The treatment of missing values in logistic regression. *Biometrical Journal*, 31(1):35–47, 1989. [cited at p. 30]
- T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer series in statistics. Springer, 2nd edition, 2009. [cited at p. 36]
- S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 2nd edition, July 1998. [cited at p. 16, 17, 20]
- C. Kim. Modeling surrender and lapse rates with economic variables. *North American Actuarial Journal*, 28.1:54–64, 2005. [cited at p. 8, 9]

- F. Kitchens. Financial implications of artificial neural networks in automobile insurance underwriting. *International Journal of Electronic Finance*, 3(3):311 – 319, 2009. [cited at p. 11]
- B. Korb and A. Nicholson. *Bayesian Artificial Intelligence*. Series in Computer Science and Data Analysis. Chapman and Hall/CRC, 2003. [cited at p. 12, 14, 15, 16, 21, 30]
- L. Mauer and N. Holden. Determinants of the lapse rate in life insurance operating companies. *Review of Business*, 28.1:54–64, 2007. [cited at p. 8]
- A. Nagy. *neural: Neural Networks*, 2007. R package version 1.4.2. [cited at p. 25]
- V. Palade and L. Jain. Practical applications of neural networks. *Neural Computing and Applications*, 14(2):95–96, July 2005. [cited at p. 11]
- J. Pearl. Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proceedings of the 7th Conference of the Cognitive Science Society*, pages pp. 329–334, University of California, Irvine, CA, 1985. University of California. [cited at p. 12]
- J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988. [cited at p. 15]
- R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. URL <http://www.R-project.org>. ISBN 3-900051-07-0. [cited at p. 25, 27]
- A. Ribert, E. Stocker, Y. Lecourtier, and A. Ennaji. Optimizing a neural network architecture with an adaptive parameter genetic algorithm. *Lecture Notes in Computer Science*, 1240/1997:527–535, 2006. [cited at p. 26]
- A. Shapiro. The merging of neural networks, fuzzy logic, and genetic algorithms. *Insurance: Mathematics and Economics*, 31(1):115–131, August 2002. [cited at p. 11]
- P. K. Sharpe and R. J. Solly. Dealing with missing values in neural network-based diagnostic systems. *Neural Computing and Applications*, 3(2), 1995. [cited at p. 30]
- P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. The MIT Press, 2nd edition, 2001. [cited at p. 16]
- W. Sukthomya and J. Tannock. The optimisation of neural network parameters using taguchia’s design of experiments approach: an application in manufacturing process modelling. *Neural Computing and Applications*, 14.4:337–344, 2005. [cited at p. 26]
- R.J. Verrall. A unified framework for graduation. *Insurance: Mathematics and Economics*, 20(2):147 – 147, 1997. ISSN 0167-6687. doi: DOI:10.1016/S0167-6687(97)80654-3. [cited at p. 7]
- Z.J. Viharos, L. Monostori, and T. Vincze. Training and application of artificial neural networks with incomplete data. In *IEA/AIE '02: Proceedings of the 15th international conference on Industrial and engineering applications of artificial intelligence and expert systems*, pages 649–659, London, UK, 2002. Springer-Verlag. ISBN 3-540-43781-9. [cited at p. 30]

- K. Weiyu, C. Tsai, and W. Chen. An empirical study of the lapse rate: The cointegration approach. *The Journal of Risk and Insurance*, 70(3):489–508, 2003. [cited at p. 13]
- H. White. A heteroskedasticity-consistent covariance matrix estimator and direct test for heteroskedasticity. *Econometrica*, 48:817–838, 1980. [cited at p. 8]
- R. Zhang and A.J. Bivens. Comparing the use of Bayesian networks and neural networks in response time modeling for service-oriented systems. In *SOCP '07: Proceedings of the 2007 workshop on Service-oriented computing performance: aspects, issues, and approaches*, pages 67–74, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-717-9. doi: <http://doi.acm.org/10.1145/1272457.1272467>. [cited at p. 30]

---

# List of Figures

---

3.1	Example of a Bayesian network pertaining to withdrawal rates. . . . .	12
3.2	Multilayer perceptron neural network with 1 hidden layer . . . . .	18
4.1	Bayesian network graph learned from data. . . . .	23
4.2	Proposed Bayesian network. . . . .	24
4.3	MLP NN for predicting withdrawal rates. Only a subset of the inputs are shown. . . . .	25
5.1	Confusion matrix used in ROC analysis . . . . .	31
5.2	Example of segmentation of data into training and testing sets for cross-validation with 3 iterations. . . . .	33
5.3	ROC graphs for models. . . . .	34

---

# List of Tables

---

3.1	Conditional probability table for node <b>Withdrawal</b> . . . . .	13
4.1	Variables selected through backwards stepwise regression on full data set. . . . .	27
5.1	Table of results showing evaluation metrics . . . . .	33