Neural Networks versus CHAID

A White paper from smartFOCUS
June 1999

Prepared By : Janine Okell, smartFOCUS

ABSTRACT

This paper describes two types of predictive *model*s: CHAID and Artificial neural networks. It describes the business benefit of predictive modelling and then describes the differences between these two modelling techniques. For those technically inclined, there is a section on how to build these *model*s. Also the advantages and disadvantages of each *model* are described and compared. Finally a glossary of the terms in italics is provided, this also offers alternative terms that may be used instead of the term chosen within the body of this paper.

So read on to find out why you should be using predictive modelling, if CHAID is “old hat”, or whether it is fair to condemn artificial neural networks as a “black box”.

Version: 2.0
Date last amended : 30th June 1999
Document ID: Neural Networks versus CHAID.doc

Neural Networks versus CHAID

CONTENTS

1. Why bother: the business use of these techniques

These techniques or *models* are used commercially to gain business benefit. Such benefits include:

- Targeting those customers who will purchase a particular product
- Discriminating between your most profitable customers and customers who lose you money
- Identifying customers who will remain loyal and ones who may go elsewhere
- Cross-selling products to interested customers
- Taking a relationship (consolidated) view to avoid departments working against each other, for instance when a credit card is offered to a competitor's high balance customer while another department offers the same potential customer a personal loan to clear that high balance.

These benefits are identified as the *outcome* of the modelling technique. For instance targeting those customers who will purchase a particular product can be modelled by using *independent variables* to predict the *outcome* (of purchasing a particular product).

There are numerous reasons why these techniques have become popular recently. The overwhelming reason is competitive pressure to gain information so as to provide an improved customised service and increase profit. Another reason that these techniques are now used is that it has become affordable to analyse the masses of transactional data that businesses generate and often this data is more accessible because it has been put in a *data warehouse.* Furthermore the possibility of consolidating different sources of data to create a customer view means that there is more potential for insight into customer behaviour.

2. Some terminology

Both Neural Networks and CHAID are called data mining techniques, although like all data mining techniques they have their origin in other areas. Data mining is the analysis and exploration of large quantities of data to discover non-trivial patterns and rules for commercial benefit. It is relatively new idea and has borrowed the means from statistics, artificial intelligence and computer science.

Generally the most useful "non-trivial pattern" is predictive. If a corporation can predict customer behaviour, like remaining loyal or purchasing in response to a campaign, then they will gain commercial advantage. Predictions can be made using data mining techniques and *statistical models*. Using the terms precisely, "Data Mining" identifies patterns of behaviour for some (but not all) of the data whereas "*Statistical Modelling*" produces a *model* that explains the different types of behaviour of all the data.

Data Mining is a stage of Knowledge Discovery in Databases (KDD). The stages of KDD are data preparation, data exploration, data mining, modelling, implementation and finally a report or white paper.

2.1. What is CHAID?

CHAID is both a data mining technique and a *statistical model*. It has been successfully used to identify target groups of customers for direct mail for many years. CHAID belongs to a set of *models* called decision trees. It is used for classification and prediction purposes. Classification attempts to identify what the categorical outcome will be given a set of criteria. Prediction attempts to predict the target or *outcome* given a future set of criteria or *independent variables.*

A decision tree represents a series of questions/rules, based on *independent variables*, shown as a path through the tree. Oddly, decision trees are shown going down from the root *tree node* towards the leaf *tree nodes*. A decision tree can be built using an algorithm that splits records into groups where the probability of the *outcome* differs for each group based on values of the *independent variables*. There are a variety of decision tree algorithms:

- CHAID (see below)
- *CART* = Classification And Regression Trees, developed by L Brieman et al.
- *C4.5*, based on ID3 (Interactive Dichotomiser 3) by J Ross Quinlan.

CHAID was developed by J A Hartigan from a technique called Automatic Interaction Detection (AID). CHAID uses the *chi squared* test to determine whether to branch further and if so which *independent variables* to use. Hence it's name Chi Squared Automatic Interaction Detection (CHAID).

It was developed to identify interactions for inclusion into *regression models*. CHAID easily copes with interactions, which can cause other modelling techniques difficulty. Interactions are combinations of *independent variables* that affect the outcome. For instance, profitability may be at the same level for low transactions/high balance credit customers as for high transaction/low balance credit customers; in this case of modelling profitability, these two *independent variables* (transactions and balance) should not be considered in isolation.

CHAID can be used alone or can be used to identify *independent variables* or *sub-populations* for further modelling using different techniques, like regression, artificial neural networks or *genetic algorithms*.

2.2. What is a neural network?

Actually a neural network is a brain, what we mean by neural network is an Artificial Neural Network (ANN). It is a misconception to think that an ANN is a "brain in a box". An ANN is a simple *model* of nodes and interconnections, which have a similar format to the neural interconnections in the brain. It is not a fake, but currently an ANN has the brain capacity of an earthworm, as such it can *model* only very basic *outcomes*. For instance whether a robot has reached the edge of a car (or the earthworm has come to the surface) and should turn around or not. To the author, it seems most unlikely that an ANN could be developed to perform the extremely complex actions of a human brain. Yet this similarity to a brain is one of the most compelling attractions of this data mining technique.

There are two more common misconceptions regarding ANN. One is that ANNs are believed to be synonymous with data mining, this is incorrect. ANNs are simply one type of data mining technique. The second is that all ANNs learn to identify new objects like humans can. This misconception may be due to their ability to undergo unsupervised as well as supervised learning. Unsupervised learning is akin to "clustering" in that there are no correct answers, the ANN will attempt to identify patterns of data based on the input data only. For ANNs, unsupervised learning has had few successes. Within supervised learning ANNs, there are known answers, or *outcomes*, as is the case in regression or CHAID analysis.

For supervised learning ANNs, it should be clear that the terms "learning" or "training" are used in the same way that a statistician would say that a *regression model* is "estimated" or a CHAID model is "built".

An ANN is used for estimation and prediction purposes. Estimation unlike classification allows continuous outcomes to be estimated. Similarly *regression models* are used for estimation and prediction purposes. ANNs are more complex than *regression models* and as such have the potential to out-perform them. Yet in practice there are several drawbacks that limit the success of ANNs, these are detailed later.

ANNs originated from attempts to *model* biological processes using computers, it is a type of "machine learning" and differs slightly from Artificial Intelligence (AI) because AI uses rules and machine learning uses data. Generally machine learning has had more successes than AI.

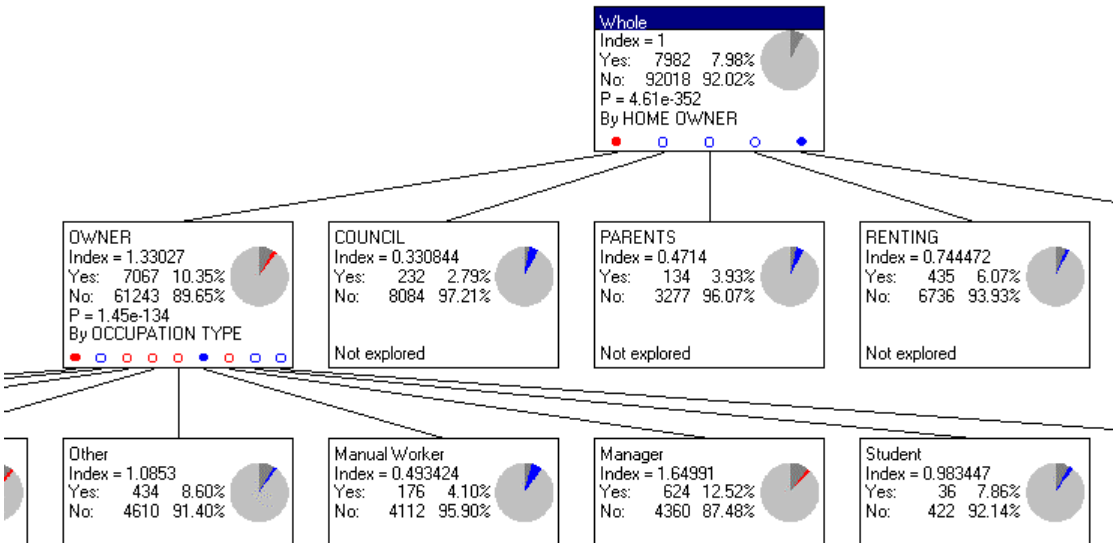
3. How do you develop these models

3.1. How to build a CHAID model

A CHAID *model* uses the CHAID algorithm to split records into groups with the same probability of the *outcome*, based on values of *independent variables.* The algorithm starts at a root *tree node*, dividing into child *tree nodes* until leaf *tree nodes* terminate branching. Branching may be *binary, ternary* or more. The splits are determined using the *Chi Squared* Test. This test is undertaken on a cross-tabulation between *outcome* and each of the *independent variables*. The result of the test is a "p-value". The p-value is the probability that the relationship is spurious, in statistical jargon this is the probability that the Null Hypothesis is correct. The p-values for each cross-tabulation of all the *independent variables* are then ranked, and if the best (the smallest value) is below a specific threshold then that *independent variable* is chosen to split the root *tree node*.

This testing and splitting is continued for each *tree node*, building a tree. As the branches gets longer there are fewer *independent variables* available because the rest have already been used to further up that branch. The splitting stops when the best p-value is not below the specific threshold. The leaf *tree nodes* of the tree are *tree nodes* that did not have any splits with p-values below the specific threshold or all *independent variables* are used.

Example of a CHAID decision tree



This outlines a purely automated approach to building a tree. The best trees are built when a *model* builder checks each split and makes rational decisions (using *background domain knowledge*) as to the appropriateness of splitting on a particular variable at a specific point. The *model* builder can spot splits using *independent variables* that raises questions as to quality, hence avoiding problems of building an invalid tree, or *model,* on poor input data. A *model* builder may also decide to stop splitting at a higher level than the automated approach would stop, in order to produce a simpler *model*.

The other decision trees of *CART* or *C4.5* uses different techniques to split and create trees.

3.2. How to build an ANN

The basic structure of an ANN is a set of input nodes, one or more hidden layers of nodes and a set of output nodes. The input nodes and output nodes must have values between 0 and 1. The interconnections between the nodes of each hidden layer has an associated weight. To train (or as we would say in statistics, to *model*) an ANN, each case (record) is presented and the values of the output nodes are calculated. Each node value is calculated by multiplying the weight by value of each node feeding that node and then summing these.

For the more useful supervised learning ANNs, the actual output (or *outcome*) is then compared to this calculated, or derived, output. The set of actual output nodes will have a value from 0 to 1. For a categorical outcome then one node will have a value of 1 and the rest of the output nodes will have values of 0. The set of derived output nodes will have values from 0 to 1. For all the output nodes of a presentation of a case, *p* the actual output value is subtracted from the derived output and added to the total averaged error.
In algebraic expressions this may be described as follows:

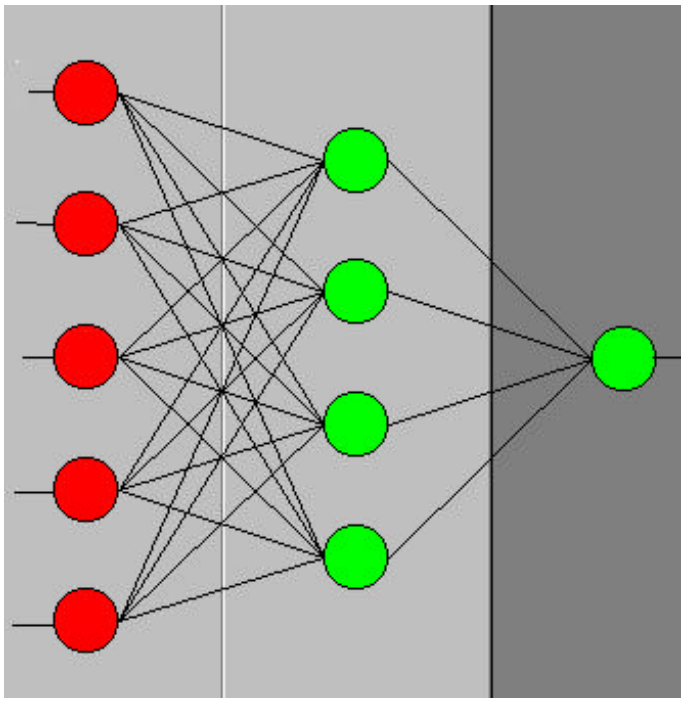
total averaged error = $E = 1/m$ (Sum of Ep)

m = number in training sample

For each presentation of a case, *p*
$Ep = 1/n$ (Sum of squared {actual output – derived output}) for each output node.

n = number of output nodes

Example of an Artificial Neural Network



Input Layer Hidden Layer Output Layer

A 3-layer feed forward ANN with 5 input and 1output nodes. The hidden layer has 4 nodes.

The weights of the interconnections are iteratively changed as the training set is continually presented to the neural network. For example if the derived output was much higher than the actual output then the nodes in the hidden layers with the highest values will have their weights changed more than others. This continues until (hopefully) the absolute minimum total error is found.

There are many types of ANN, but the most useful one is called “Multi-Layer Feed Forward (MLFF), these use the “back propagation of errors” method to change the weights of the interconnections. The search technique most often used here is the “method of gradient steepest descent” algorithm. On occasions a *sub-optimal* solution is found because a local minimum is found instead of the absolute minimum.

Two very technical notes:

1. A method of improving this search technique to avoid finding sub-optimal solutions and increase the speed is called “simulated annealing”.
2. The use of *genetic algorithms* as an optimised search technique to find the best solution is being investigated but currently *genetic algorithms* have not proved better than the “method of gradient steepest descent” and “simulated annealing”.

The solution is then a set of weights, which have no meaning in themselves. This is why an ANN is often described as a “black-box”.

4. The strengths and weaknesses

In this section the individual strengths and weaknesses of both techniques are detailed and then there is a direct comparison of both techniques in terms of:

- Clarity and explicability.
- Implementation/integration.
- Data requirements.
- Accuracy of model.
- Construction of model.
- Costs.

4.1. Strengths of CHAID

Explicability. The form of a CHAID tree is intuitive, it can be expressed as a set of explicit rules in english. This means that the business user can confirm the rationale of the *model* and if necessary, modify the tree or direct it’s architecture from their own experience or their *background domain knowledge*. Input data quality problems can be spotted, hence problems of building an invalid *model* on poor input data can be avoided. Also the most important predictors (or *independent variables)* can easily be identified and understood.

Ease of Implementation/integration. The rules from the path down the tree can be expressed in logical format (for instance SQL) that can be implemented directly with little risk of IT coding errors. Such rules work well with relational databases. Also a CHAID *model* can be used in conjunction with more complex *models*. For instance, a CHAID *model* could identify who may be at risk of leaving and then a more complex profit *model* could be used to determine whether the customer is worth keeping.

Ease of construction. CHAID *models* can handle categorical (like marital status) and banded continuous *independent variables* (like income). In particular if the *independent variables* are categorical with high cardinality (implicit "containing" relationships) CHAID should perform even better. A CHAID *model* automatically prevents *over-fitting* and handles missing data. CHAID does not require much computational power. The analytical skills required to build a CHAID *model* are not exceptional, often a general marketing analyst could build a CHAID model.

4.2. Strengths of ANNs

Wide applicability. ANNs can be applied to both directed (supervised) and undirected (unsupervised) data mining. ANNs can handle both categorical (e.g. marital status) and continuous (e.g. income) *independent variables* without banding.

A complex solution. ANNs can produce a *model* even in situations that are very complex because an ANN produces non-linear models. Theoretically, an ANN should be more accurate than other techniques, although several tests have shown that ANNs do not significantly out-perform logistic *regression modelling*.

4.3. Weakness of CHAID

Data Volumes. CHAID needs rather large volumes of data to ensure that the number of observations in the leaf *tree nodes* large enough to be significant.

Continuous variables must be banded. Continuous *independent variables*, like income, must be banded into categorical-like classes prior to use in CHAID.

4.4. Weakness of neural networks

Pre-processing the data. The *independent variables* must be converted into the range from 0 to 1, this is done using "transformations" which can be inaccurate when the *independent variables* are skewed with a few outliers.

Continuous output. The output from an ANN is usually continuous which may be difficult to transform to a discrete categorical *outcome*.

Setting up the model parameters. Several parameters must be set up, for instance the number of hidden layers and the number of nodes per hidden layer. These parameters affect the *model* built. The results of small differences in these parameters can be the difference between a very predictive *model* and a poor *model.*

Lack of clarity. The results of an ANN cannot be explained, it is a "black-box", a set of weights with no inherent meaning. Recently, some explanation of an ANN may be obtained by using additional techniques to visualise the networks and to produce rules from prototypes (using sensitivity analysis) that attempt to describe an ANN. Explicability is a legal requirement for credit product application *models* in the US, this means that ANNs cannot be used to build credit risk *models*. The lack of clarity means that unfair prejudice cannot be ruled out from the credit decision.

Sub-optimal solutions. ANNs can produce *models* that are *sub-optimal.*

Over-fitted models. To build an ANN *model* requires an experienced statistician or expert ANN user to ensure that the *model* is not *over-fitted.*

Time-consuming and costly. This method can be very time-consuming because of the number of re-presentations of the data that is required during training. Also if there is a large number of predictive variables then the time taken to find a solution are further lengthened. The skill and effort required to build an ANN plus the time involved means that this technique is costly.

Slow in implementation. Most operational systems and relational database systems are optimised to perform data movements fast but are slow at computations. An ANN requires many computations hence the performance of an operational system with an embedded ANN will be relatively slow. Also to code, or embed, an ANN developed by a specialised ANN package may not be straightforward.

4.5. CHAID and ANNs in comparison

4.5.1. Clarity and explicability.

The form of a CHAID *model* can be understood, it is a set of rules, whereas an ANN is obscure, the weights have no intuitive meaning. It is possible to apply *background domain knowledge* to a CHAID *model* because it should be easy to explain and thus involve a *domain expert* or business user.

4.5.2. Implementation/integration.

It is much easier for a CHAID *model* to be implemented than an ANN. Moreover, the risks of miss-coding by an IT department are slim for a CHAID *model* and rather higher for an ANN. Furthermore, the performance of an implemented CHAID *model* will be significantly faster than an implemented ANN.

4.5.3. Data requirements.

More data must be provided for a CHAID *model* to ensure that there is critical mass in the leaf nodes following many branches. The data for both techniques will require some pre-processing. ANNs require that all the possible values of *independent variables* are transformed into O/1 input nodes. Before using CHAID, any continuous *independent variables* must be banded.

4.5.4. Accuracy of model.

ANNS should provide more accurate (powerful/predictive) *models*, especially for complex problems. Yet there is a risk of finding *sub-optimal* solutions and *over-fitting*.

4.5.5. Construction of model.

CHAID is easier and quicker to construct, whereas ANNs must have many parameters set and require more skilled manipulation to ensure *over-fitting* does not occur. It is a lot harder to apply *background domain knowledge* using ANNs whereas it is a lot easier to see mistakes and *over-fitting* in a CHAID tree.

4.5.6. Costs.

Building an ANN will be more costly then a building a CHAID *model*. Building an ANN will take more time and a higher level of skill will be required for building an ANN than for a CHAID *model.*

5. Applications

Both CHAID and ANNs can be used to create predictive models. Such *models* include attrition, churn, propensity and customer lifetime value. Yet in general the application of ANNs is wider than CHAID because ANNs can be applied to both directed (supervised) and undirected (unsupervised) data mining. ANNs can handle both categorical (e.g. marital status) and continuous (e.g. income) *independent variables* but these have to be transformed to 0/1 input variables. When all or most of the *independent variables* are continuous, ANNs should perform better than CHAID. When all or most of the *independent variables* are categorical with high cardinality (implicit "containing" relationships) CHAID should perform better than ANNs.

In addition to the more common predictive *models* of marketing, both ANNs and CHAID can be used to attempt to solve *sequence prediction* problems, for example predicting share prices in the stock markets, but the effort required to pre-process time series data is large.

ANNs may be used to solve estimation problems (with continuous outcomes). Whereas CHAID can provide good solutions to classification problems, and can also be used for exploratory analysis (perhaps prior to another modelling technique) and to provide descriptive rules.

Both techniques, in line with all modelling techniques have these prerequisites:

- Ensure that the *independent variables* are indeed independent of the outcome to be predicted. In other words, ensure that the outcome is not directly determined by the *independent variables*. Also the *independent variables* must be available at the decision point. Such data is often demographic data, like age, gender, occupation, etc.
- Identify *outcome* or objective precisely.
- The *model* builder must know the data and understand relationships between fields - they must have *background domain knowledge*.
- Derived fields may have to be built. This is often the case for time series (or sequence) data that is to be used for sequence modelling (for instance the stock market).

6. Conclusions

CHAID *models* are easier to build and implement than ANNs and also are less costly than ANNs. Theoretically ANNs should provide *models* that are better than CHAID in terms of power and accuracy. That means they should be more powerful at discriminating between groups that fit the target (for instance, churn) and that they should get the prediction correct more often. Currently, this is not the case, perhaps because of the problems of *over-fitting* and *sub-optimal* solutions.

How exactly a CHAID *model* is working can be easily seen, it is intuitive, but an ANN has been described as a “black-box” because it is not possible to explain how each outcome is determined. For this reason, the US credit industry is prohibited from using ANNs to determine credit risk, because the lack of clarity means that unfair prejudice cannot be ruled out from the credit decision.

In terms of predictive modelling, CHAID wins against ANNs, yet in the future if ANNs become easier to build and the methods of producing rules that explain an ANN improve, ANNs may become the winners. An interesting development is the combination of these two techniques to create “Neural trees”. These could use the CHAID method to identify *sub-populations* upon which ANNs could be built to predict a particular target.

7. Glossary

Background domain knowledge*.* This is the experience and understanding of the data, inter-relationships and quality, which various users/collectors of the data will have attained while working in the business. A person with *background domain knowledge* will be a *domain expert.*

Binary. Two way. A binary split is a split into two.

C4.5. This is a decision tree. C4.5 uses the reduction of a measure of diversity to firstly build a tree and then undergoes pruning. This is unlike CHAID that does not use pruning but stops building the tree before *over-fitting* occurs. C4.5 can use both categorical variable and continuous variables.

CART. This is a decision tree. CART can only produce a tree with binary splits. CART uses the reduction of a measure of diversity to firstly build a tree and then undergoes pruning. This is unlike CHAID that does not use pruning but stops building the tree before *over-fitting* occurs. CART can use both categorical variable and continuous variables.

Chi squared. This is a statistical distribution that is used in the chi squared test. The chi squared test is a statistical test on a cross-tabulation that measures the significance of the relationship between the two variables of the cross-tabulation.

Data warehouse. Summarised company wide data stored separately from operational data so that it is accessible for strategic and tactical purposes. As such accessing data in a data warehouse will not impinge on operational systems. The data will be time-stamped (in monthly/weekly blocks), hence figures will be non-volatile because they will not change with day to day transactions. Alternative terms include: data mart (actually a smaller scale data warehouse, for instance a marketing data mart)

Demographic data*.* This is data about a person, for instance age, gender, occupation, postal areas etc.

Domain expert*.* This is a user or collector of the data. They will have attained experience and understanding of the data, inter-relationships and the data quality while working in the business. As such a *domain expert* has *background domain knowledge.*

Genetic Algorithms. An optimised search technique that uses selection, cross-over and mutation operators to evolve successive generations of solutions. Only the most predictive solutions survive, until the functions converge on an optimal solution. Genetic algorithms, like ANNs, were developed in an attempt to *model* biological processes; both are processes called “machine learning” and differs slightly from Artificial Intelligence (AI) because AI uses rules and machine learning uses data.

Independent variables*.* The inputs into a model. For a predictive customer lifetime value *model* this may be *demographic data* and perhaps previous transaction data. Alternative terms include: predictive variables or inputs.

***Model**.* This is a representation of reality. See also *Statistical Modelling.*

Outcome The output from a model. For a predictive churn *model* this is the predicted churned/did not churn variable. Alternative terms include: target, dependent variable, predicted variable, output or objective.

Over-fitting. This is when a *model* attempts to interpret the input data, or *independent variables,* to such an extent that random relationships which occur only in that dataset are modelled.

***Regression models**.* This is a *model* built using a statistical method called regression. The *model* is built by finding a line (or plane if there is more than one *independent variable*) of best fit through the data using the least squared difference method. Alternative terms and related techniques include: log-linear *models*, logistic *models* and logit *models*.

***Sequence prediction**.* These problems are about predicting what will happen next. The best known example is to predict share prices on the stock markets. Alternative terms include: time series analysis.

Statistical Modelling. This is a *model* built using statistics. These include *regression models* and CHAID *models*.

***Sub-optimal**.* This means not the best. Sub-optimal solutions may be found instead of the best solution. An ANN finds a sub-optimal solution when a local minimum instead of a global minimum is found in the attempt to locate a minimum point in multi-dimensional space.

***Sub-populations**.* These are specific groups from a larger population or database, for instance the age range 50-65 would be a sub-population. Independent *Statistical Modelling* may be undertaken on each separate sub-population.

Ternary. Three way. A ternary split is a split into three.

Tree Nodes. These are groups identified by particular values of *independent variables.* For instance, a tree node may be for young people (less than 30), married with an income over £30,000. These tree nodes make up the decision tree, nodes are split into other tree nodes by branching on values of an *independent variable.* The root tree node is (strangely) at the top of the tree and represents all the data. Leaf tree nodes are at the end of the tree and all together they represent all the data segmented into groups. Note that ANNs also use node as a term, in this document I have discriminated nodes used in CHAID decision trees by describing them as tree nodes.