23

Data Mining: Machine Learning and Statistical Techniques

Alfonso Palmer, Rafael Jiménez and Elena Gervilla
University of the Balearic Islands
Spain

1. Introduction

The interdisciplinary field of *Data Mining* (DM) arises from the confluence of statistics and machine learning (artificial intelligence). It provides a technology that helps to analyze and understand the information contained in a database, and it has been used in a large number of fields or applications. Specifically, the concept DM derives from the similarity between the search for valuable information in databases and mining valuable minerals in a mountain. The idea is that the raw material is the data to analyse, and we use a set of learning algorithms acting as diggers to search for valuable nuggets of information (Bigus, 1996).

We offer an applied vision of DM techniques, in order to provide a didactic perspective of the data analysis process of these techniques. We analyze and compare the results from applying machine learning algorithms and statistical techniques, under DM methodology, in searching for knowledge models that show the structures and regularities underlying the data analysed. In this sense, some authors have pointed out that DM consists of "the analysis of (often large) observational datasets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner" (Hand, Mannila & Smyth, 2001), or, more simply, "the search for valuable information in large volumes of data" (Weiss & Indurkhya, 1998), or "the discovery of interesting, unexpected or valuable structures in large databases" (Hand, 2007). Other authors define DM as "the exploration and analysis of large quantities of data in order to discover meaningful patterns and rules" (Berry & Linoff, 2004).

These definitions make it clear that DM is an appropriate process for detecting relationships and patterns in large databases (although we point out that it can also be applied in relatively small databases). In this sense, the concept of Knowledge Discovery in Databases (KDD) has been frequently used in the literature to define this process (Han & Kamber, 2000, 2006; Hand et al., 2001), specifying that DM is a stage of the process, and highlighting the need for a previous stage of integration and collection of data (if we start with large raw databases), and also the stage of cleaning and preparing data (data pre-processing) before building descriptive/predictive models in the DM stage (applying suitable techniques to the analysis requirements). On the other hand, several authors have used the concept of DM (instead of KDD) to refer to the complete process (Bigus, 1996; Two Crows, 1999; Paul, Guatam & Balint, 2002; Kantardzic, 2003; Ye, 2003; Larose, 2005).

In our work, we focus on the DM stage; i.e., the phase of application of suitable modelling techniques according to analysis needs. We show the analysis and comparison of several techniques to obtain knowledge models (predictive models) - we start from a pre-processed, relatively small volume of data (as we have said before, handling large databases is not a necessary requirement to apply data mining techniques). We analyze several machine learning and statistical (classical and modern) techniques. In order to choose these techniques, we took as a reference the work of the editors Michie et al. (1994), in which they review the wide repertory of classification techniques. In particular, we chose two classical machine learning techniques, Artificial Neural Networks (ANN) and Decision Trees (DT), two modern statistical techniques, k-Nearest Neighbor (k-NN) and Naive Bayes (NB), and a classical statistical technique, Logistic Regression (LR). Recently, Nisbet et al. (2009) have presented a work in which they explain and exemplify the use of these classification techniques with the *Statistica* platform, although they also use (to a lesser extent) the *SPSS Clementine* and *SAS Enterprise Miner* platforms. In our work, we exemplify the use of these techniques with the non commercial *Weka* platform.

The aim of the work that we are presenting is double. On the one hand, we present a comparison of the five aforementioned techniques, from a theoretical (methodological) and applied perspective. The applied perspective is covered from a case study, with the intention of comparing the performance of the models obtained with these techniques from one and the same database; with this applied aim, we use the Weka platform, an open code, freely distributed, *data mining* platform (developed in Java), in order to cover the second of the aims proposed in this work, which is to exemplify the advantages of Weka in order to carry out the comparative performance study from its *Explorer* and *Experimenter* modules.

2. Methodology

In this section we aim to offer an integrating view of the use of *Data Mining* (DM) methodology and techniques through a presentation of the procedures common to these techniques in the process of obtaining predictive models, and through a description of the methodological peculiarities associated to them. Specifically, we focus on a description of techniques which allow us to generate categorical response predictive models (classification models).

Table 1 presents a classification of some techniques included in DM according to the nature of the data analyzed. In this sense, we present the techniques available depending on the nature of the predictor variables and the output variable. If the output variable is continuous or categorical we find ourselves dealing with supervised learning models, whereas if there is no output variable we are dealing with unsupervised learning models.

Specifically, we will deal with the techniques of *Neural networks*, *Classification trees*, *k-Nearest Neighbor*, *Naive Bayes* and *Logistic regression*, since they make it possible to analyze categorical output variables in order to generate classification models.

Classification is a task that belongs to the category of supervised learning and it refers to the task of analyzing a set of pre-classified objects in order to learn a model (or function) which can be used to classify unknown data in one of several predefined classes (An, 2006).

From the perspective of supervised learning, the analysis technique estimates the model from the knowledge that it has of the behaviour of each of the entries in the selected output variable, in such a way that the supervised techniques itself supervises whether or not the model it is building adjusts to the knowledge it has of the reality. In this sense, the aim of

	Continuous response	Categorical response	No response
Continuous predictors	Linear regression Neural networks k-Nearest Neighbor	Logistic regression Neural networks Discriminant analysis k-Nearest Neighbor	Principal components Cluster analysis
Categorical predictors	Linear regression Neural networks Regression trees	Neural networks Classification trees Logistic regression Naive Bayes	Association rules

Table 1. Data mining techniques according to the nature of the data (Shmueli et al., 2007)

supervised learning is to generate knowledge based models which will help in predicting the behaviour of new data.

A common requirement in predictive modelling techniques is the use of a data sample (test data) which is independent of the one used in the construction of the model (training data), with the intention of assessing the model's generalization capacity (assessment of the model).

On the other hand, in unsupervised learning there are no known results to guide the algorithm in obtaining the model, but rather this explores the properties of the data with the aim of identifying behaviour patterns with no knowledge of these "a priori". In this way, the aim of unsupervised learning is to generate knowledge based models with a descriptive, not predictive, intention.

2.1 Artificial Neural Networks

Artificial Neural Networks (ANN) are data processing systems whose structure and functioning are inspired by biological neural networks. ANN were developed based on the following guidelines:

- Information processing occurs in simple elements called neurons.
- The neurons transmit signals through established connections.
- Each connection (communication link) has an associated weight.
- Each neuron applies an activation function (usually non linear) to the total entry of connected neurons received (sum of entries weighted according to the connection weights), thus obtaining an output value which will act as the entry value which will be transmitted to the rest of the network.

The fundamental characteristics of ANN are parallel processing, distributed memory and adaptability to the surroundings.

The processing unit is the artificial neuron, which receives the entries from the neighbouring neurons and calculates an output value, which is sent to all the remaining neurons.

As far as the representation of input and output information is concerned, we can find networks with continuous input and output data, networks with discrete or binary input and output data and networks with continuous input data and discrete output data.

An ANN is made up of the sequential order of three basic types of nodes or layers: input nodes, output nodes and intermediate nodes (hidden layer) (Figure 1). The input nodes are

in charge of receiving the initial values of the data from each case in order to transmit them to the network. The output nodes receive input and calculate the output value.

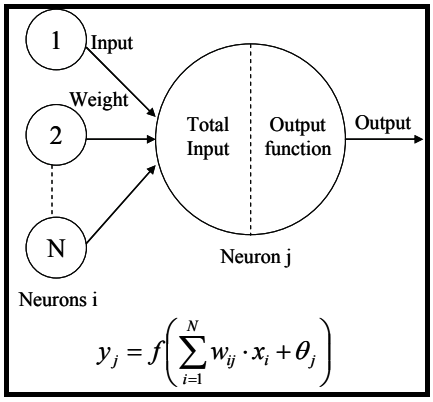

Fig. 1. Generic working of an artificial neuron and its output mathematical representation

This set of nodes used by the ANN, together with the activation function, makes it possible for the ANN to easily represent non-linear relationships, which are the most difficult as far as multivariate techniques are concerned.

The most used activation functions are: the step function, identity function, sigmoid or logistic function and the hyperbolic tangent.

There is a large selection of ANN models. A combination of the topology (number of neurons and hidden layers, and how they are connected), the learning paradigm and the learning algorithm define an ANN model (Bigus, 1996).

It can be said that an ANN has three advantages which makes it very attractive in data handling: adaptive learning through examples, robustness in handling redundant and inaccurate information and massive parallelism.

The most used method in the practical applications of ANN is the *multilayer perceptron*, which was made popular by Rumelhart et al. (1986).

A multilayer perceptron type of ANN starts with an input layer in which each node or neuron corresponds to a predictor variable. These input neurons connect with each of the neurons making up the hidden layer. The nodes in the hidden layer in turn connect with the neurons in another hidden layer. The output layer is made up of one (binary prediction) or more output neurons. In this sort of architecture, the information is always transmitted from the input layer towards the output layer.

The popularity of the multilayer perceptron is mainly due to the fact that it is capable of acting as a universal function approximator. More specifically, a "backpropagation" network which contains at least one hidden layer with enough non-linear units can learn any sort of function or continuous relationship between a group of input variables (discrete and/or continuous) and an output variable (discrete or continuous). This property makes multilayer perceptron networks general, flexible and non-linear tools. A complete description of the mathematical foundations associated with the training stage and the functioning stage of the *backpropagation* algorithm in multilayer perceptron architecture can be found in Rumelhart et al. (1986).

The usefulness of the multilayer perceptron, lies in its ability to learn virtually any relationship between a set of input and output variables. On the other hand, if we use techniques derived from classical statistics such as linear discriminant analysis, this does not have the capacity of calculating non-linear functions and, therefore, will show a lower performance compared to the multilayer perceptron in classification tasks that involve complex non-linear relationships.

When the network is used to classify normally the output layer has as many nodes as the number of classes and the node of the output layer with the highest value offers the estimate of the class which the network makes for a certain input. In the special case of two classes it is common to have a node in the output layer, and the classification between the two classes is carried out by applying a cut off point to the node value.

If one of the virtues of ANN is that they allow modelling any sort of functional relationship (linear or non linear) between variables and, therefore, act as universal function approximators, another of the outstanding advantages of this technique, compared to classical modelling techniques, is that it does not impose any sort of restriction with respect to the starting data (type of functional relationship between variables), neither does it usually start from specific assumptions (like the type of distribution the data follow). Another virtue of the technique lies in its capacity to estimate good models even despite the existence of noise in the information analyzed, as occurs when there is a presence of omitted values or outlier values in the distribution of the variables. Hence, it is a robust technique when dealing with problems of noise in the information presented; however, this does not mean that the cleaning criteria of the data matrix should be relaxed.

Nevertheless, its extreme flexibility lies in the need to have sufficient training data and that it requires more time for its execution than other techniques (Shmueli et al., 2007). It is worth pointing out that in ANN, as well as the set of training data to build the model and the set of independent data (test data) in order to assess its generalization capacity, a third set of independent data (validation set) is used to avoid overfitting the model (during the learning process) which can cause an excessive number of parameters or weights regarding the problem (Hastie et al., 2001, p. 356).

Despite the advantages presented concerning the technique, on the other hand, one of the most important criticisms that have been raised against the use of ANN focuses on the fact that a knowledge of the weights within the network does not in general help the interpretation of the underlying process that the prediction of a certain output value generates. To put it another way, the reproaches against the use of this technique are limited to the difficulty in understanding the nature of the internal representations generated by the network in response to a certain problem. Despite this, this perception of ANN as a complex "black box" is not completely true. In this sense, different attempts at interpreting the weights of the neuronal network have arisen, of which the most widely used is the so-called sensitivity analysis (Montaño & Palmer, 2003), implemented in ANN programmes as recently presented by Palmer et al. (2001), under the name of *Sensitivity Neural Network 1.0*.

2.2 Decision Trees

Decision trees (DT) are sequential partitions of a set of data that maximise the differences of a dependent variable (response or output variable). They offer a concise way of defining groups that are consistent in their attributes but which vary in terms of the dependent variable.

DT are made up of nodes (input variables), branches (groups of entries in the input variables) and leaves or leaf nodes (values of the output variable).
The construction of a DT is based on the principle of "divide and conquer": through a supervised learning algorithm, successive divisions of the multivariable space are carried out in order to maximise the distance between groups in each division (that is, carry out partitions that discriminate). The division process finalizes when all the entries of a branch have the same value in the output variable (pure leaf node), giving rise to the complete model (maximum specified). The further down the input variables are in the tree, the less important they are in the output classification (and the less generalization they allow, due to the decrease in the number of inputs in the descending branches).
To avoid overfitting the model, the tree can be pruned by eliminating the branches with few or scarcely significant entries. As a result, if we start from the complete model, after the tree pruning this will gain in generalization capacity (assessed with test data), at the expense of reducing the degree of purity of its leaves (Larose, 2005).
There are different learning algorithms designed to obtain DT models (see Table 2). The learning algorithm determines the following aspects:

- Specific compatibility with the type of variables: nature of the input variables and the output variable.
- Assessment procedure of the distance between groups in each division: division criteria.
- Restrictions can be placed on the number of branches each node can be divided into.
- Pruning parameters [pre-pruning / post-pruning]: minimum number of entries per node or branch, critical value of the division, performance difference between the extended and reduced tree. Pre-pruning implies using stopping criteria during the construction of the tree, whereas post-pruning applies the pruning parameters to the whole tree.

The most used algorithms (Table 2) are CART (*Classification And Regression Trees*), CHAID (*Chi-Squared Automatic Interaction Detection*), QUEST (*Quick, Unbiased, Efficient Statistical Tree*) and C4.5 / C5.0.
The CART algorithm was designed by Breiman et al. (1984) and it generates binary decision trees, where each node is divided exactly into two branches. In this way, if the input variable is nominal and has more than two categories, it groups different categories in one branch. If the input variable is nominal or continuous, it still generates two branches, associating a set of values limited by the operators to each one of them "less than or equal to" or "greater" than a certain value. The CART algorithm makes it possible to introduce nominal, ordinal and continuous input data into the model. The output variable of the model may likewise be nominal, ordinal or continuous.
The CHAID algorithm (Kass, 1980) was originally designed to handle only categorical variables. Nevertheless, nowadays it makes it possible to handle nominal and ordinal categorical output data and continuous variables. The tree construction process is based on the calculation of the significance of a statistical contrast as a criterion in order to decide the hierarchy of importance of the predictor variables, and to establish clusters of similar values (statistically homogeneous) with respect to the output variable, keeping all the values that turn out to be heterogeneous (distinct) unaltered. Similar values are melted in one category, forming part of one branch of the tree. The statistical test used depends on the level of measurement of the output variable. If the aforementioned variable is continuous, the F test is used. If the output variable is categorical, the Chi-square test is used.

A differential characteristic between the CART and CHAID algorithms is that the latter allows the division of each node into more than one branch; therefore it tends to create much wider trees than the binary development methods.
The QUEST algorithm (Loh & Shih, 1997) can be used if the output is nominal-categorical (allows the creation of classification trees). The tree construction process is also based on the calculation of the significance of a statistical contrast. For each input variable, if this is a nominal categorical variable, it calculates the critical level of a Pearson Chi-square independence contrast between the input variable and the output variable. If the input variable is ordinal or continuous, it uses the F test.
The C5.0 algorithm (Quinlan, 1997) only admits categorical output variables. Input variables may be categorical or continuous. This algorithm is the result of the evolution of algorithm C4.5 (Quinlan, 1993) designed by the same author and which has as a nucleus the ID3 version (Quinlan, 1986). The ID3 algorithm is based on the concept of *information gain* to select the best attribute.

ALGORITHMS	Input variables	Output variable	Type of prediction	Splitting branches	Splitting criterion
CHAID	categorical / numerical	categorical / numerical	classification / regression	≥2	Chi-square / F
QUEST	categorical / numerical	categorical	classification	=2	Chi-square / F
CART	categorical / numerical	categorical / numerical	classification / regression	=2	GINI / Least squared deviation
C4.5 / C5.0	categorical / numerical	categorical	classification	≥2	Gain Ratio

Table 2. Comparative between learning algorithms for decision trees (amplified version of Gervilla et al., 2009)

One of the most outstanding advantages of DT is their descriptive nature, which allows us to easily understand and interpret the decisions made by the model, as we have access to the rules that are used in the predictive task (an aspect that is not taken into consideration in other machine learning techniques, such as ANN). Thus, DT allow the graphic representation of a series of rules concerning the decision that must be made in the assignment of an output value for a certain entry, offering a friendly, intuitive explanation of the results.
On the other hand, the decision rules provided by a tree model have a predictive value (not only descriptive) from the moment in which their accuracy is assessed from independent data (test data) to the ones used in the construction of the model (training data).
Another attractive characteristic of DT is that they are intrinsically robust to missing values, as they handle them without having to impute values or eliminate observations.
Nevertheless, DT do have some weaknesses (Shmueli et al., 2007): they are sensitive to small changes in the data and, unlike the models that assume a particular relationship between the response and the prediction (e.g. a linear relationships like a linear regression), DT are

not linear and are not parametrical. This allows for a wide range of relationships between the predictors and the response, but it can be a weakness: given the fact that the partitions are carried out on unique predictors rather than on combinations of predictors, DT probably omit relationships between predictors, particularly linear structures such as linear or logistic regression models.

Another drawback in the construction of DT is the problem of overfitting the model, that is to say, the DT includes not only the real patterns or structures present in the data, but also part of the "noise". To reduce this problem as much as possible there are several strategies:

- Strategies that slow down the growth of the tree before it reaches the perfect classification of the examples in the training set (for instance, the CHAID algorithm).
- Strategies that make it possible for the tree to grow completely and afterwards carry out some pruning (for instance, the CART and C4.5 algorithms). These latter have been shown to be more efficient than the former.

One final disadvantage of DT is that they need a large set of data in order to build a good classifier. Nevertheless, Breiman & Cutler (2004) have introduced "Random Forests", which deal with these limitations. The basic idea is to create multiple DT from the data (and thus obtain the "forest") and to combine their result to obtain a better classifier (see Breiman, 2001).

2.3 k-Nearest Neighbor

When we run into new situations, we human beings are guided by memories of similar situations we have experienced in the past. This is the basis of the k-Nearest Neighbor (k-NN) technique. That is, the k-NN technique is based on the concept of similarity. Moreover, this technique constructs a classification method without making assumptions concerning the shape of the function that relates the dependent variable with the independent variables. The aim is to identify in a dynamic way *k* observations in the training data that are similar to a new observation that we want to classify. In this way, *k* similar (neighbouring) observations are used to classify the observation specifically in a class (see Figure 2). More specifically, k-NN looks for observations in the training data that are similar or near to the observation that has to be classified, based on the values of the independent variables (attributes). Then, depending on the classes of these nearby observations, it assigns a class to the observation that it wishes to classify, taking the majority vote of the neighbours to determine the class. In other words, it counts the number of cases in each class and assigns the new case to the one that most of its neighbours belongs to (Two Crows, 1999).

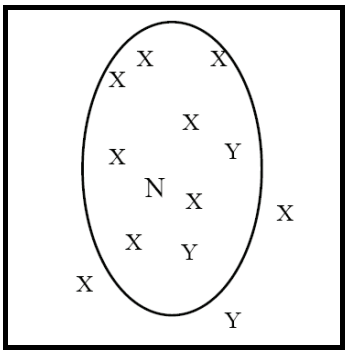

Fig. 2. Graphical representation of k-NN classification (Two Crows, 1999)

Even though the method may appear "naive", it is capable of competing with the other more sophisticated classification methods. Hence, where the linear model is rigid, k-NN is extremely flexible. The performance of this technique for data of the same size depends on *k*, and on the measurement used to determine which observations are nearby.

As a result, in the application of the technique we must take into account how many neighbours are to be considered (*k* value), how we measure the distance, how we combine the information for more than one observation and whether all the neighbours should bear the same weight (see Berry & Linoff, 2004).

As has been said, the k-NN technique classifies an unknown example in the most common class by using its *k* nearby neighbours. It assumes that all the examples correspond to points in an *n*-dimensional space. A neighbour is considered nearby if it has the least distance in the *n*-dimensional space of attributes (An, 2006). If we set *k*=1, the unknown example is classified in the class of the nearest neighbour in the training data.

Although there is no formula to choose the *k* number, it is worth noting that if we choose a small *k* value, the classification may possibly be too affected by outlier values or unusual observations. On the other hand, choosing a not very small *k* value will tend to damp any idiosyncratic behaviour learnt from the training data. Nevertheless, if we choose a *k* value that is too large, locally interesting behaviour will be overlooked.

You can let the data help to solve this problem by following a procedure of cross validation. That is, we can try several *k* values with different training sets chosen at random and choose the *k* value that minimizes the classification error.

With respect to the way of measuring the distance, the most common distance function is the Euclidean distance (1), where x and y represent the *m* values of the attributes of two cases.

$$d_{\text{Euclidean}}(x, y) = \sqrt{(x_i - y_i)^2} \tag{1}$$

Although, alternatively, the Manhattan distance may also be used (2)

$$d_{\text{Manhattan}}(x, y) = | x_1 - x_2 | + | y_1 - y_2 | \tag{2}$$

The Euclidean distance has three drawbacks:

- The distance depends on the units chosen to measure the variables.
- It does not take into account the variability of the different variables.
- It ignores the correlation between variables.

One solution is to use a measurement called statistical distance (or *Mahalanobis* distance).

By way of advantages of the k-NN technique we can highlight (An, 2006; Mitchell, 1997; Nisbet et al., 2009): first of all, it does not simplify the distribution of objects in space in a set of comprehensible characteristics; instead, the training set is stored completely as a description of this distribution. Furthermore, the k-NN method is intuitive, easy to implement and effective in practice. It can construct a different approximation to the target function for each new example to be classified, which is advantageous when the target function is very complex, but it can be described by a collection of less complex local approximations. Finally, this technique builds a classification method without carrying out assumptions concerning the shape of the function that relates the dependent variable (classification variable) with the independent variables (attributes).

On the other hand, the most important disadvantage is that k-NN is very sensitive to the presence of irrelevant parameters. Other disadvantages:

- The time to find nearby neighbours in a large training set may be very high.
- The number of observations that are needed in the training data increases exponentially with the number of dimensions (variables).

2.4 Naive Bayes

Bayesian methods use the Bayes rule or formula (3) (based on Bayes' theorem), which expresses a very powerful framework to combine information from the sample with expert opinion (prior probability) so as to produce an up-dated expert opinion (posterior probability) (Giudici, 2003).

Specifically, the Naive Bayes technique (NB) is a very powerful classification technique and is one of the most widely used ones, due to its computationally simple process (Hand & Yu, 2001). As it is based on Bayes' theorem, it can predict the probability of a given case belonging to a certain class. Its computational simplicity is due to the assumption known as class *conditional independence* (this assumes that the effect of an attribute value on a certain class is independent of the values of the other attributes), and in this sense it is considered a "naive" classifier (Han & Kamber, 2000, 2006).

This classifier predicts that a case A will belong to class C_i which has the highest X conditioned posterior probability (set of attributes of the case in the predictor variables). Bayes' theorem allows us to define the Bayes' formula (3) that this posterior probability provides; and since P(X) is constant for all the classes, we only need to maximize $P(X \mid C_i)P(C_i)$ in the classification process.

$$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)} \tag{3}$$

From a set of training data, $P(C_i)$ how many times each class C_i occurs in these data can be estimated. To reduce the computational cost of estimating $P(X \mid C_i)$ for all possible x_k (predictor variables), the classifier uses precisely the "naive" assumption that the attributes used to describe X are conditionally independent from each other given class C_i. This conditional independence can be found in the expression (4), where the *m* value indicates the number of predictor variables that participate in the classification.

$$P(X \mid C_i) = \prod_{k=1}^{m} P\left(x_k \mid C_i\right) \tag{4}$$

The studies that compare classification algorithms (e.g. Michie et al., 1994) have often shown that NB is comparable in its working with ANN and DT, and in fact exceeds these sophisticated classifiers if the attributes are conditionally independent given the class. Recent theoretical analyses have shown why NB is so robust (Domingos & Pazzani, 1997; Rish, 2001).

The appeal of NB classifiers lies in their simplicity, computational efficiency and good performance in classification. What is more, NB can easily handle unknown values or missing values. Nevertheless, it has three important drawbacks (Shmueli et al., 2007): first of all, it requires a large number of cases in order to obtain good results; secondly, if a prediction category is not present in the training data, the technique assumes that a new

case with this category in the predictor has zero probability; finally, even though we obtain a good performance if the aim is to classify or order cases according to their probability of belonging to a certain class, this method offers very biased results when the aim is to estimate the probability of belonging to a class.
Above and beyond these drawbacks, the NB technique is straightforward to use, it adjusts to the data and is easy to interpret. In addition, it requires only one exploration of the data. This simplicity, parsimony and interpretability has led it to enjoy widespread popularity, especially in the literature of machine learning (Hand et al., 2001).

2.5 Logistic Regression

Linear regression is used to approach the relationship between a continuous response variable and a set of predictor variables. However, when the response variable is categorical, linear regression is not appropriate.
Logistic regression (LR) is a generalized linear model. It is used mainly to predict binary variables (with values like yes/no or 0/1). Thus, LR techniques may be used to classify a new observation, whose group is unknown, in one of the groups, based on the values of the predictor variables.
As with linear regression, the classification depends on the linear combination of the attributes. The logistic function (5) transforms the linear combination into an interval [0,1] (Ye, 2003). Thus, in order to use LR, the dependent variable is transformed into a continuous value which is a function of the probability of the event happening (Parr-Rud, 2001; Witten & Frank, 2005).

$$p = \frac{1}{1 + \exp^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k)}} \tag{5}$$

In LR we follow two steps: the first step consists of estimating the probability of belonging to each group and in the second step we use a cut off point with these probabilities to classify each case in one of the groups. The parameters of the model are estimated with the method of maximum likelihood through a process of successive iterations. For a more detailed explanation of the LR technique, consult Larose (2006).
Lastly, it is worth commenting that LR can produce stable results with relatively few data (Harris-Jones & Haines, 1998). On the other hand, the fact that traditional regression is so widely accepted, easily implemented and generally understood makes it even more attractive (Hill et al., 2004). What is more, LR shows behaviour analogous to a diagnostic test.

3. Applying data mining techniques

In this section we go further, from the applied point of view, into the integration within DM methodology of the techniques described in the previous section. We use the *Weka* platform to meet this aim. In the first section, we give a brief description of the functionalities integrated in the Weka interface, with the aim of providing the reader with a presentation of the tool and to specify its virtues. In the second section, we propose a case study in order to compare with Weka the performance of predictive models obtained with the techniques indicated.

3.1 Weka interface

Weka (Waikato Environment for Knowledge Analysis) is a data mining platform distributed under public license GNU-GPL: it is free software that can be freely used, copied, studied, modified and distributed and it is protected from appropriation attempts that would restrict these user liberties.

Bearing in mind its characteristics, we find that it is a tool which, first of all, has an interactive interface which contains four user-machine interaction modalities (Fig. 3):

- **Explorer:** is the most used mode and the most descriptive.
- **Experimenter:** useful mode to compare the performance of different predictive models (experiments).
- **KnowledgeFlow:** allows the visual programming of modelling design through connected object modules.
- **Simple CLI (Simple Client):** provides a console to execute the functionality of the system through commands; it makes it possible to carry out any operation supported by Weka directly, although it does demand a comprehensive command of the application.

Fig. 3. Applications of the Weka interface

Within the **Explorer** modality, it provides extensive support to the overall process of data mining (Fig. 4):

1. Access to databases, exploration and selection of data and data processing:
   - **Preprocess**: functionality aimed at importation, transformation (application of filters) and data extraction.
   - **Visualize**: functionality aimed at the visualization of data using graphic techniques.
2. Predictive and descriptive modelling. Compiles a wide range of data mining procedures for the obtention of knowledge models:
   - **Classify** (classification and regression): predictive modelling (supervised learning).
   - **Cluster** (grouping) and **Associate** (association rules): descriptive modelling (unsupervised learning).
   - **Select attributes**: selection of predictive attributes.

Lastly, it is worth highlighting the possibility of **system extensibility**: it allows the user to modify Weka by integrating new functionality developed in Java code, using its structure and object oriented functional design. This represents the main advantage as opposed to other closed code data mining platforms (commercial programmes).

In Witten & Frank (2005) you can find a detailed description of the different modalities of interaction with Weka.

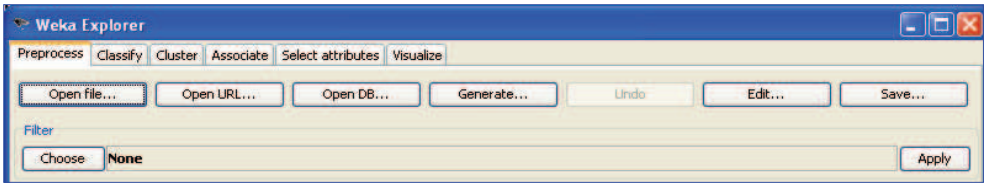

Fig. 4. Weka Explorer interface

3.2 Case study

Once we have presented the methodological basis of the techniques involved in this work, in this section we aim to contribute comparative elements of the information provided by these techniques in an applied context. These comparative elements refer to the predictive power (accuracy) of the knowledge models generated and the descriptive components that add informative value to the decision making processes (classifications). Thus we aim to provide a more integrating view, if possible, of DM methodology, since we provide common assessment parameters in order to compare the results obtained.

Nevertheless, from the analysis of these results we do not aim to reach substantial conclusions related to the context from which the data used comes, but rather our intention is to divulge to the readers a series of methodological tools that allow us to detect knowledge patterns in a way that is practically automatic and, on the other hand, to make it easier to interpret the descriptive elements associated with the assessment of the models obtained.

From an initial sample of 9300 young people aged between 14 and 18 years, in which information concerning variables intervening in the consumption of addictive substances was collected, we selected a sub-sample of 2526 young people. We are interested in studying the relationship between the consumption / non consumption of cannabis (output variable) among the people surveyed and the reasons the subjects surveyed have for consuming or not consuming drugs (input variables). Specifically, we collected fifteen possible reasons (variables) for consuming drugs and eleven reasons (variables) for not consuming; the possible response to each of these variables is dichotomous (yes/no). On the other hand, if in the initial sample (complete) the percentage of consumers of cannabis is nearly 18%, as opposed to 78% of non consumers, the sub-sample selected shows a greater balance between consumers/non consumers (44.4%/55.6%); this equilibrium (or balancing) is justified by methodological motives, as there must be a similar number of entries in each of the output categories (consumes/does not consume), so that they can be equally represented in the modelling stage (detection of classificatory patterns).

In a session with **Explorer**, first of all we loaded (Open file…) in the **Preprocess** section the data to be analyzed, whose structure (database) has been adapted to the Weka format: Arff file. Once the data file is open, it is possible to explore the variables they contain (Fig. 5). It is also possible to read data in the CSV (comma delimited) format from Weka, although it is not possible to import databases in the more widely used formats such as Excel, Access, SPSS, etc. However, there is the possibility of converting these other more common formats into the native Arff format from the data mining platform *RapidMiner* (free, open code software, which also allows the use of the algorithms included in Weka). For instance, if the data source is in SPSS format, we can indicate whether we are interested in extracting the names of the variables and/or their labels in another format (Arff, in this case), and whether we are interested in using the labels of the values (option be default) instead of the numerical values.

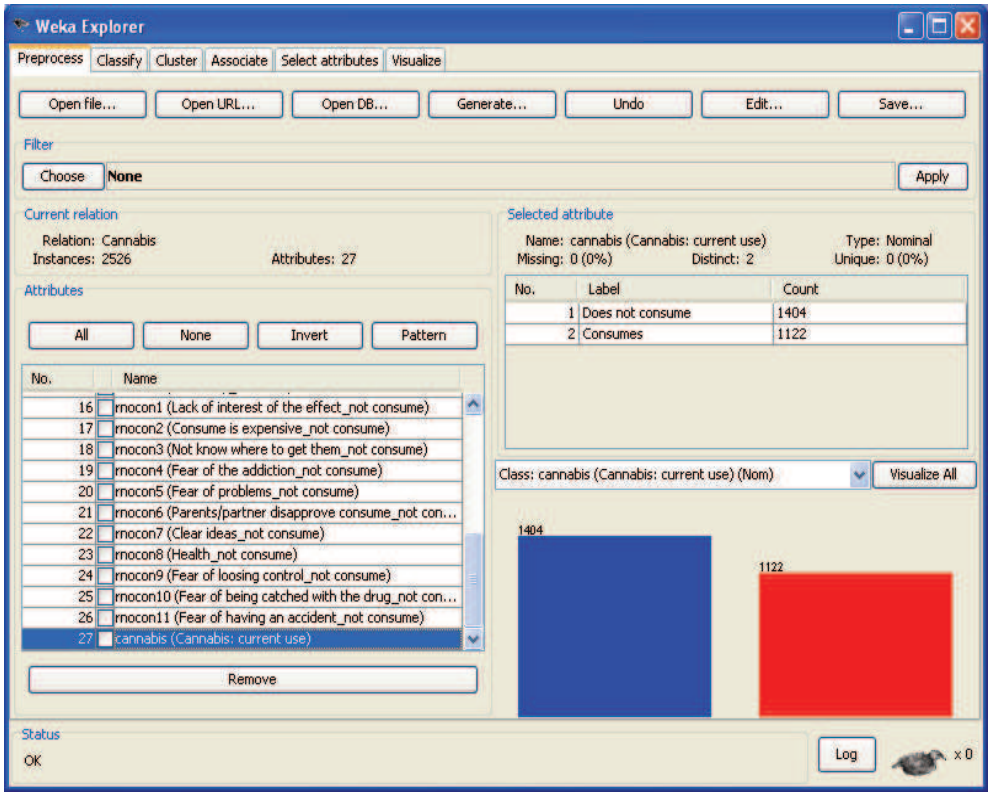

Fig. 5. Weka Explorer interface: exploring variables

From the section **Classify**, we access the different modelling techniques integrated in Weka (Fig. 6). For instance, we can select the *J48* classifier (weka.classifiers.trees.J48); this classifier uses the C4.5 algorithm (Quinlan, 1993) to generate a classification tree which is in agreement with a series of parameters determined by the algorithm (to edit them, click on the classifier) and other parameters determined by data mining methodology (in *Test options*). In the example (Fig. 6) we have indicated that the J48 classifier uses 70%of the sample (training data) to create the model, and the rest as test data. The output variable is also indicated (predicted variable), which by default is the last variable in the database. The *Start* button allows us to generate the model and access (in *Classifier output*) the model's assessment results. It can be observed that the model has correctly classified 599 of the 758 test patterns (79%), with a larger percentage of hits in the category *Does not consume* (83.4%) than in the category *Consumes* (73.9%).

It is possible to access the graphic representation of the classification tree (Fig. 7) through the options of the contextual menu (*Visualize tree*) of the model generated (in the *Result list*).

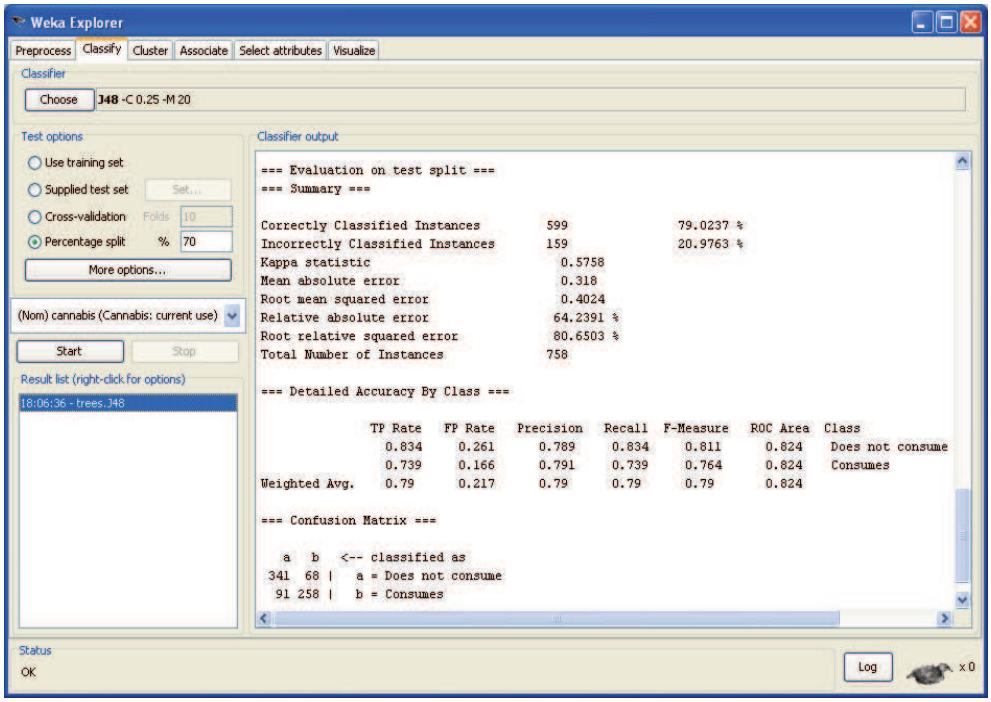

Fig. 6. Evaluation of the selected decision tree model on test data

What is more, the descriptive nature of decision trees allows us to easily derive the *classification rules* used by the model, with information from their **support** (percentage of cases in which the antecedent and consequent of the rule – correct prediction – are found with respect to the total subjects) and their **confidence** (percentage of cases in which the antecedent and consequent of the rule are found with respect to the number of subjects in which the antecedent of the rule is found). In Table 3 we show some examples. If we focus on the rule associated to leaf 1 (*Leaf1* in Fig. 7), it can be seen that out of the 1768 subjects who took part in the generation of the model (training stage), 550 (31.3%) of them consider – antecedents of the rule – their pleasurable nature a reason for consuming drugs (rcons3) and do not consider the fact that their friends consume a reason for consuming (rcons9) and at the same time –consequent of the rule – they are consumers of cannabis. Hence, if we want to classify new subjects in the output variable (consumes / does not consume), in the case in which their values in the output variables coincide with the model's rule, this would indicate that they are consumers of cannabis, with a confidence of 82.3% in the decision adopted. The confidence of the rule indicates to us, therefore, that there are 17.7% of subjects (118 cases) who fulfil the antecedent of the rule, but not the consequent (they are not consumers of cannabis).

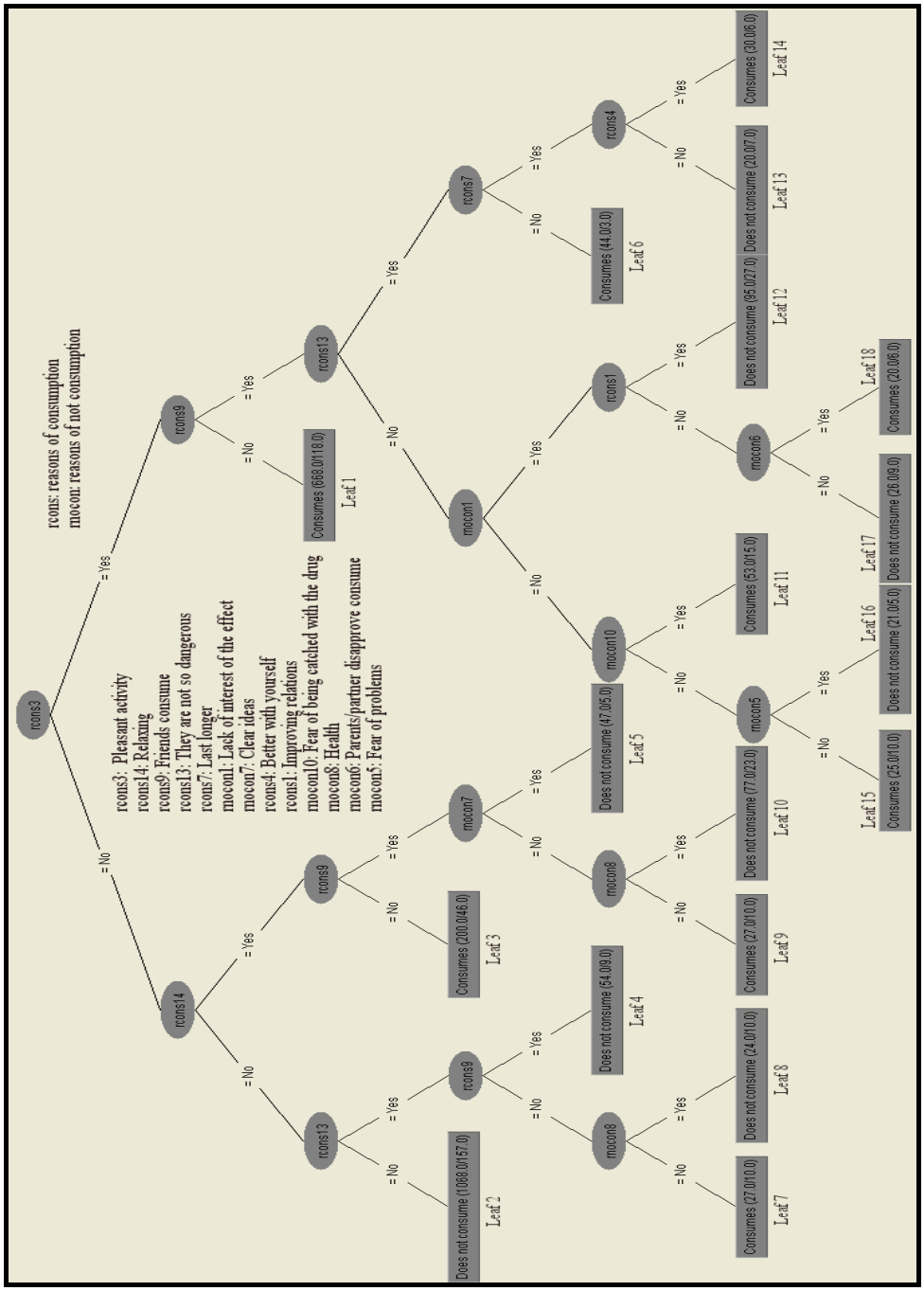

Fig. 7. Decision tree to classify subjects in "consumes" or "does not consume" cannabis

Leaf 1 **Rule:** IF (rcons3 = Yes) & (rcons9 = No) THEN (Consumes) **Cases:** 668; **Correctly classified:** 550 → **Rule support:** 31.1% (550/1768) **Rule confidence:** 82.3% (550/668)
Leaf 2 **Rule:** IF (rcons3 = No) & (rcons14 = No) & (rcons13 = No) THEN (Does not consume) **Cases:** 1068; **Correctly classified:** 911 → **Rule support:** 51.5% (911/1768) **Rule confidence:** 85.3% (911/1068)
Leaf 5 **Rule:** IF (rcons3 = No) & (rcons14 = Yes) & (rcons9 = Yes) & (rnocon7 = Yes) THEN (Does not consume) **Cases:** 47; **Correctly classified:** 42 → **Rule support:** 2.4% (42/1768) **Rule confidence:** 89.4% (42/47)
Leaf 14 **Rule:** IF (rcons3 = Yes) & (rcons9 = Yes) & (rcons13 = Yes) & (rcons7 = Yes) & (rcons4 = Yes) THEN (Consumes) **Cases:** 30; **Correctly classified:** 24 → **Rule support:** 1.4% (24/1768) **Rule confidence:** 80% (24/30)

Table 3. Some classification rules from the selected decision tree

As mentioned above, with Weka we have generated other classifying models from the techniques described in the previous section, using the same training (70%) and test (30%) sub-samples in all of them.

Specifically, to generate a neural network we used the *MultilayerPerceptron* classifier (weka.classifiers.functions.MultilayerPerceptron) (it uses the *backpropagation* algorithm), keeping 20% of the training data as validation data (*validationSetSize* parameter of the classifier).

In order to obtain the k-Nearest Neighbor model, we used the *IBk* classifier (weka.classifiers.lazy.IBk), with the Euclidean distance function.

Lastly, we generated a Naive Bayes model (weka.classifiers.bayes.NaiveBayes) and a logistic regression model (weka.classifiers.functions.Logistic).

Table 4 shows the comparison of the correct classifications (accuracy) between models, through the respective confusion matrices (bivariate contingency tables in which the real classification categories are crossed with the entries in the categories estimated by the model). It is shown that in all the models selected there is a relatively high accuracy (ranging between 76.6% for the *IBk* model and 80.1% for the *Logistic* model). If we analyze the accuracy according to the classificatory category, in all the models the percentage of correct classifications in the category *Does not consume* is greater (ranging between 81.4% for the *NaiveBayes* model and 90.5% for the *IBk* model), whereas in the *Consumes* category the range of correct classifications moves between 60.5% for the *IBk* model and 75.9% for the *Logistic* model.

Since the same test sub-sample was used in all the models selected, it is possible to carry out a comparison of the coincidences in the classification of the cases (Table 5). The condition of coincidence was established in those cases in which all the models converge on the same

		Actual category		
Decision Tree		Does not consume	Consumes	Total
Predicted category	Does not consume	**341**	91	432
	Consumes	68	**258**	326
J48	Total	409	349	**758**
	Accuracy	83.4%	73.9%	**79.0%**
Artificial Neural Network		Does not consume	Consumes	Total
Predicted category	Does not consume	**358**	111	469
	Consumes	51	**238**	289
MultilayerPerceptron	Total	409	349	**758**
	Accuracy	87.5%	68.2%	**78.6%**
k-Nearest Neighbor		Does not consume	Consumes	Total
Predicted category	Does not consume	**370**	138	508
	Consumes	39	**211**	250
IBk	Total	409	349	**758**
	Accuracy	90.5%	60.5%	**76.6%**
Naive Bayes		Does not consume	Consumes	Total
Predicted category	Does not consume	**333**	96	429
	Consumes	76	**253**	329
NaiveBayes	Total	409	349	**758**
	Accuracy	81.4%	72.5%	**77.3%**
Logistic Regression		Does not consume	Consumes	Total
Predicted category	Does not consume	**342**	84	426
	Consumes	67	**265**	332
Logistic	Total	409	349	**758**
	Accuracy	83.6%	75.9%	**80.1%**

Table 4. Confusion matrix and model performance with test data

classification, whereas the condition of non coincidence is given when at least one model classifies in a different way to the rest. It can be seen that there are 76% of cases in which all the models converge on the same classification, with the greater agreement found in the category *does not consume*.

	Prediction agreement			
	Does not consume	Consumes	Not agreement	Total
Cases	367	209	182	758
Percent	48.4%	27.6%	24%	
	Agreement: 76% (576 cases)			

Table 5. Prediction agreement between models

We can also analyze the confusion matrix between the cases that converge on the classification predicted by the five models (576 subjects) and the real classification category (Table 6). We found that the percentage of correct classifications is 84.9%, with greater accuracy (91.9%) in the category *does not consume*.

		Actual category		
Agreement		Does not consume	Consumes	Total
Predicted category	Does not consume	**307**	60	367
	Consumes	27	**182**	209
	Total	334	242	**576**
	Accuracy	91.9%	75.2%	**84.9%**

Table 6. Confusion matrix between prediction agreement and actual classification

Weka also allows us to access the individual prediction data if the option *Output predictions* is activated using the button *More options…* (see Fig. 6); these predictions are included in the window of results (*Classifier output*) together with the model's assessment information. In this way, we can compare the predicted classification and confidence level (probability) in this classification, case by case and for each model.

We extracted the individual predictions obtained in each of the five models to a data base outside Weka (Fig. 8), so as to be able to compare the degree of agreement and the confidence level in the joint classificatory decision (0: does not consume; 1: consumes) for a certain case. For instance, we can compare the predicted classification in cases 1 and 6; in both cases, the five models converge on the classification (*consumes*), even though the confidence level for this decision is lower (on average) in subject 1 (p=0.805) than in subject 6 (p=0.925).

Weka also allows us to reassess a given model with a new set of data independent from the one used in its construction. Through the contextual menu of the *Result list* section of Explorer (see Fig. 6), the model can be loaded in memory (*Load model* option) if it had been saved (*Save model* option) before closing the Weka session in which it was built. Once the model has been loaded in the memory, a new set of data must be chosen through the

Supplied test set option, and finally select *Re-evaluate model on current test set* in the contextual menu of the model. This option can also serve in new cases to find out the classification proposal of already validated predictive models (by activating the *Output predictions* option).

Finally, it is possible to access the Weka *Experimenter* model in order to configure experiments (simulations) with different predictive model techniques applied to the same set of data (Fig. 9), and thus be able to assess and compare the capacity of the techniques to generate good predictive models.

	actual	predictedJ48	predictedMLP	predictedKNN	predictedNB	predictedLog	predictionJ48	predictionMLP	predictionKNN	predictionNB	predictionLog
1	1	1	1	1	1	1	,816	,903	,636	,868	,803
2	0	0	0	0	0	0	,731	,733	,687	,583	,500
3	1	0	0	0	0	0	,835	,867	,565	,646	,820
4	0	0	0	0	0	0	,835	,889	1,000	,835	,819
5	1	0	0	1	0	0	,835	,902	,600	,793	,841
6	1	1	1	1	1	1	,816	,963	,857	,996	,993
7	0	0	0	0	0	0	,835	,890	,700	,925	,850
8	1	0	0	0	0	0	,835	,942	,722	,911	,914
9	1	1	0	0	0	0	,774	,687	,600	,537	,533
10	0	0	0	0	0	0	,835	,936	,818	,962	,962
11	0	1	0	0	1	1	,816	,689	,680	,558	,542
12	1	0	1	1	1	1	,835	,766	,545	,889	,792

Fig. 8. Comparative according to cases (test data) of the degree of agreement between models

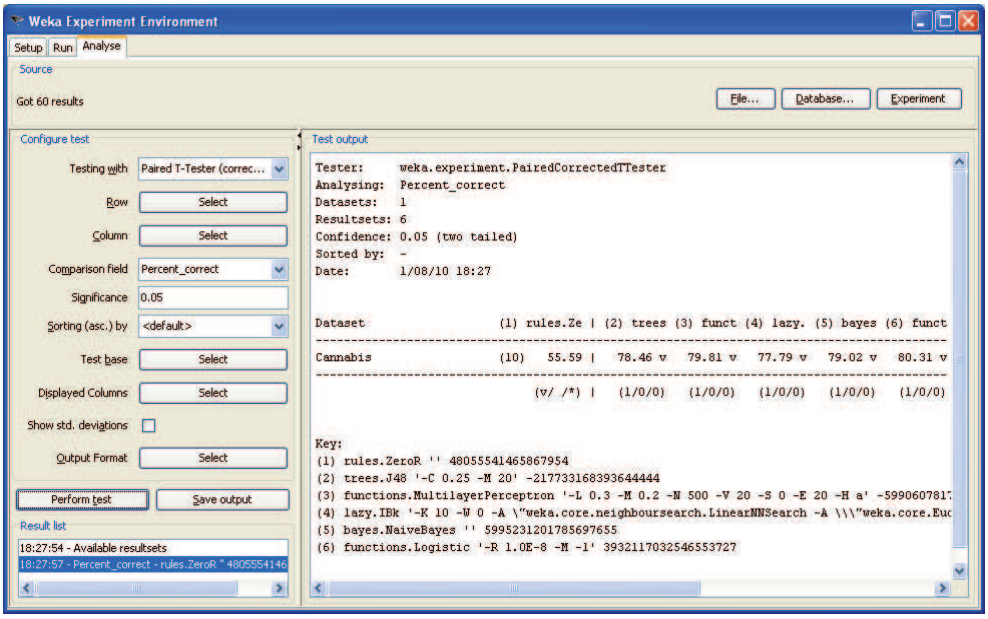

Fig. 9. Comparison of the accuracy (percent correct) between techniques

We proposed an experiment to compare the five classificatory techniques under study in relation to the ZeroR classifier (weka.classifiers.rules.ZeroR). This is the simplest classifier, which predicts the mode (the most repeated value) for a categorical output variable (and the mean for a numerical output variable). In the experiment, we indicate the division of the sample into training (70%) and test (30%) data and 10 repetitions (models) were specified for each technique.

The comparison was carried out from the mean percentage values of the cases correctly classified in the ten models generated for each technique and their standard deviations (hidden by default, but which can be shown if the corresponding option is chosen). As well as the percentage of correct cases (*Percent_correct*) it is possible to choose other indices of interest in the *Comparison field* (see Fig. 9), such as the area under ROC curve (*Area_under_ROC*). The values of these indices for each of the specified repetitions are stored in an Arff data file, whose variables can be analyzed in Weka itself or in other data analysis programmes (with prior extraction to a compatible data format).

By pressing the button *Experiment* (Fig. 9), the mean values and standard deviations of the chosen comparison index are used to study (from a statistical *t* test) whether there are significant differences between the classificatory accuracy of the models obtained from the first of the techniques defined in the experiment (in this case, ZeroR) and the classificatory accuracy of each of the remaining techniques used.

As can be observed, the ZeroR classifier indicates us that the mean percentage of correct classifications (with test data) is 55.6%, which obviously corresponds to the percentage of subjects in the sample who are non consumers (most frequent category). The rest of the classifiers show a much higher mean percentage of classifications, whose difference with respect to this reference value is, in all cases, statistically significant.

4. Discussion

When faced with the question, *what is the best algorithm for classification?* There is evidently no general answer that can help us to know prior to an analysis of the data which technique or algorithm I should apply in order to obtain the best classificatory model. In this sense, Nisbet et al. (2009, p. 256) indicate us that if different classificatory algorithms are used, we will discover that the best algorithm for classifying a set of data may not work well in another set of data; in other words, different techniques or algorithms have a better functioning in different data sets, and in this sense, they claim that "*using a diversity of algorithms is best*". They even establish an analogy between the process of creation of knowledge models and the process of sculpting a statue, which they call "*the art of data mining*" (Nisbet et al., 2009, p. 46). In the literature you can find DM definitions which point in this same direction, for instance: "data mining is the art of discovering meaningful patterns in data" (Pyle, 2003). Weiss & Indurkhya (1998, p. 21), reflect on this question and ask the question, "*Data Mining: Art or Science?*", which they answer, "no universal best approach is describable for data mining; making good decisions is part art, part science"; in this sense, these authors combine the art and science of DM in their work: they use science when it is known and effective, and offer guidance in practical issues that are not easily quantifiable.

Our scientific contribution to DM has been precisely to present arguments that point in the direction of convincing applied researchers of the usefulness of using different techniques or algorithms in the search for knowledge models that will help in decision making concerning a given problem. Researchers should take on the role of designer in this task: they should design the data selection, cleaning and preparation processes, as well as the model obtention and validation processes through the wide repertory of associated techniques, algorithms and parameters that are at their disposal. Precisely, the Weka platform offers an ideal space to combine the art and science of DM in an effective way, as demonstrated in the previous section of the chapter.

5. References

An, A. (2006). Classification Methods. In J. Wang (Ed.), *Encyclopedia of Data Warehousing and Mining* (pp. 144-149). Hershey, PA: Idea Group Inc.

Berry, M. & Linoff, G. (2004). *Data Mining Techniques. For marketing, sales, and customer relationship management* (2nd ed.). Indianapolis: Wiley.

Bigus, J.P. (1996). *Data Mining with neural networks: solving business problems from application development to decision support*. New York: McGraw-Hill.

Breiman, L. & Cutler, A. (2004). *Random Forests*. Retrieved from http://stat-www.berkeley.edu/users/breiman/RandomForests/cc_home.htm

Breiman, L. (2001). Random Forests. *Machine Learning,* 45, 1, 5-32.

Breiman, L., Friedman, J.H., Olshen, R.A. & Stone, C.J. (1984). *Classification And Regression Trees.* Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software.

Domingos, P. & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning,* 29, 103-130.

Gervilla, E., Jiménez, R., Montaño, J.J., Sesé, A., Cajal, B. & Palmer, A. (2009). The methodology of *Data Mining*. An application to alcohol consumption in teenagers. *Adicciones,* 21, 1, 65-80.

Giudici, P. (2003). *Applied Data Mining. Statistical Methods for Business and Industry.* England: John Wiley & Sons.

Han, J. & Kamber, M. (2000). *Data Mining: Concepts and Techniques*. San Francisco: Morgan Kaufmann.

Han, J. & Kamber, M. (2006). *Data Mining: Concepts and Techniques* (2nd ed.). San Francisco: Morgan Kaufmann.

Hand, D.J. (2007). Principles of Data Mining. *Drug Safety,* 30, 7, 621-622.

Hand, D.J., Mannila, H. & Smith, P. (2001). *Principles of Data Mining*. London: The MIT Press.

Hand, D.J. & Yu, K. (2001). Idiot's Bayes – not so stupid after all? *International Statistical Review,* 69, 3, 385-398.

Harris-Jones, C. & Haines, T.L. (1998). Sample Size and Misclassification: Is More Always Better? In *Proceedings of the Second International Conference On the Practical Application of Knowledge Discovery and Data Mining*, London, U.K., 301-312.

Hastie, T., Tibshirani, R. & Friedman, J. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer.

Hill, C.M., Malone, L.C. & Trocine, L. (2004). Data Mining and Traditional Regression. In H. Bozdogan (Ed.), *Statistical Data Mining and Knowledge Discovery* (pp. 259-275). Boca Raton, FL: Chapman & Hall.

Kantardzic, M. (2003). *Data Mining: Concepts, Models, Methods, and Algorithms*. New York: Wiley.

Kass, G.V. (1980). An exploratory technique for investigating large quantities of categorical data. *Applied Statistics,* 29, 2, 119-127.

Larose, D.T. (2005). *Discovering Knowledge in Data : An Introduction to Data Mining*. Hoboken, NJ: Wiley.

Larose, D.T. (2006). *Data Mining Methods and Models*. Hoboken, NJ: Wiley.

Loh, W. & Shih, Y. (1997). Split selection methods for classification trees. *Statistica Sinica,* 7, 815-840.

Michie, D., Spiegelhalter, D.J. & Taylor, C.C. (Eds.) (1994). *Machine Learning, Neural and Statistical Classification.* New York: Ellis Horwood Ltd.

Mitchell, T.M. (1997). *Machine Learning*. New York: McGraw-Hill.

Montaño, J.J. & Palmer, A. (2003). Numeric sensitivity analysis applied to feedforward neural networks. *Neural Computing and Applications,* 12, 2, 119-125.

Nisbet, R., Elder, J. & Miner, G. (2009). *Handbook of Statistical Analysis & Data Mining Applications*. San Diego, CA: Academic Press.

Palmer, A., Fernández, C. & Montaño, J.J. (2001). Sensitivity Neural Network 1.0 [Computer program]. Available at mailto: alfonso.palmer@uib.es

Parr-Rud, O. (2001). *Data Mining Cookbook. Modeling Data for Marketing, Risk, and Customer Relationship Management*. New York: John Wiley & Sons.

Paul, S., Guatam, N. & Balint, R. (2002). *Preparing and Mining Data with Microsoft SQL Server 2000*. Online Books: Microsoft.

Pyle, D. (2003). Data Collection, Preparation, Quality, and Visualization. In N. Ye (Ed.), *The handbook of Data Mining* (pp. 365-391). New Jersey: Lawrence Erlbaum Associates.

Quinlan, J.R. (1986). Induction of Decision Trees. *Machine Learning*, 1, 1, 81-106.

Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Quinlan, J.R. (1997). *C5.0 Data Mining Tool*. Rule Quest Research. Avalaible from http://www.rulequest.com

Rish, I. (2001). An empirical study of the naive Bayes classifier. *Proceedings of IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence.*

Rumelhart, D.E., Hinton, G.E. y Williams, R.J. (1986). Learning internal representations by error propagation. In D.E. Rumelhart & J.L. McClelland (Eds.), *Parallel distributed processing* (pp. 318-162). Cambridge, MA: MIT Press.

Shmueli, G., Patel, N.R. & Bruce, P.C. (2007). *Data Mining for Business Intelligence. Concepts, Techniques, and Applications in Microsoft Office Excel with XLMiner.* New Jersey: John Wiley & Sons.

Two Crows Corporation (1999). *Introduction to Data Mining and Knowledge Discovery* (3rd ed.). Maryland: Two Crows Corporation.

Weiss, S.M. & Indurkhya, N. (1998). *Predictive Data Mining. A Practical Guide*. San Francisco, CA: Morgan Kauffman.

Witten, I.H. & Frank, E. (2005). *Data Mining. Practical Machine Learning Tools and Techniques* (2nd ed.). San Francisco, CA: Morgan Kauffman.

Ye, N. (Ed.) (2003). *The handbook of Data Mining*. New Jersey: Lawrence Erlbaum Associates.

Knowledge-Oriented Applications in Data Mining
Edited by Prof. Kimito Funatsu

ISBN 978-953-307-154-1
Hard cover, 442 pages
Publisher InTech
Published online 21, January, 2011
Published in print edition January, 2011

The progress of data mining technology and large public popularity establish a need for a comprehensive text on the subject. The series of books entitled by 'Data Mining' address the need by presenting in-depth description of novel mining algorithms and many useful applications. In addition to understanding each section deeply, the two books present useful hints and strategies to solving problems in the following chapters. The contributing authors have highlighted many future research directions that will foster multi-disciplinary collaborations and hence will lead to significant development in the field of data mining.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821