New Implementation of Unsupervised ID3 Algorithm (NIU-ID3) Using Visual Basic.net

FARAJ A. EL-MOUADIB1, ZAKARIA S. ZUBI2, AHMED A. ALHOUNI3

1Computer Science Department, Faculty of Information Technology, Garyounis University, Benghazi, Libya, <elmouadib@yahoo.com>
2Computer Science Department, Faculty of Science, Altahadi University, Sirte, Libya, <zszubi@yahoo.com>
3Computer Science Department, Faculty of Science, Altahadi University, Sirte, Libya, <a_a_m_alhouni@yahoo.com>

Abstract: The data volumes have increased noticeably in the few passed years, for this reason some researchers think that the volume of data will be duplicated every year. So data mining seems to be the most promising solution for the dilemma of dealing with too much data and very little knowledge. Database technology has dramatically evolved since 1970s and data mining became the area of attraction as it promises to turn those raw data into meaningful knowledge, which businesses can use to increase their profitability. The data mining systems are classified based on specific set of criteria such as classification according to kinds of databases mined, classification according to kinds of knowledge mined, classification according to kinds of techniques utilized and classification according to applications adapted. This classification can also be helpful to potential users to distinguish data mining systems and identify those that are best match their specific needs.

The purpose of this paper is to implement one of the data mining techniques (classification) to deal with labeled data sets and merging it with another data mining technique (clustering) to deal with unlabeled data sets in a computer system using VB.net 2005. Our system (NIU-ID3), can deal with two types of data files namely; text data files and access database files. It can also preprocess unlabeled data (clustering of data objects) and process label data (classification). The NIU-ID3 can discover knowledge in two different forms, namely; decision trees and decision rules (classification rules), this approach is implemented in Visual Basic.net language with SQL. The system is tested with access database, text data (labeled datasets and unlabeled datasets) and presents the results in the form of decision trees, decision rules or simplified rules.

Key Words: -Data mining, Data classification, ID3 algorithm, Supervised learning, Unsupervised learning, Decision tree, Clustering analysis.

1 Introduction

Dealing with the huge amount of data produced by businesses has brought the concept of information architecture which started new project such as Data Warehousing (DW). The purpose of DW is to provide the users and analysts with an integrated view of all the data for a given enterprise. Data Mining (DM) and Knowledge Discovery in Databases (KDD) is one of the fast growing computer science fields. The popularity and importance is caused by an increased demand for analysis tools that help analysts and users to use, understand and benefit from these huge amounts of data. One theme of knowledge discovery is to gain general ideas from specific ones, which is the basic idea of learning. Machine learning is subfield of artificial intelligence field that deals with programs that learn from experience.

Concept learning can be defined as the general description of a category giving some positive and negative examples. So, it is an automatic inference of a general definition of some concept, given examples labeled as members or non-members of the concept. "The aim of concept learning is to induce a general description of a concept from a set of specific examples." [1]. One of human cognition capabilities is the skills to learn concepts from
examples. Human have remarkable ability for concept learning with the help of only a small number of positive examples of a concept. As the concept learning problems became more complex has necessitate the existence of more expressive representative language. According to [1], most of the concept learning systems uses attribute-value language where the data (examples) and output (concepts) are represented as conjunctions of attribute-value pairs. "Concept Learning is inferring a Boolean-valued function from training examples of its input and output."[5]. So, this simplicity of representation allowed efficient learning systems to be implemented. On the other hand this simplicity had represented the difficulty of inducing descriptions involving complex relations. "Inductive Learning, a kind of learning methods, has been applied extensively in machine learning."[2]. Current machine learning paradigms are divided to two groups, learning with teacher which is called supervised learning, and learning without teacher which is called unsupervised learning.

According to [6], Data mining is the process of discovering meaningful new correlations, patterns and trends by sifting through large amounts of data stored in repositories, using pattern recognition technologies as well as statistical and mathematical techniques. Data Mining is considered to be a revolution in information processing and there are many definitions in the literature to what constitute data mining. According to [5], the attraction of the wide use of data mining is due to:

- The availability of very large databases.
- The massive use of new techniques coming from other disciplines of computer science community like neural networks, decision trees, induction rules.
- Commercial interests in order to propose individual solutions to targeted clients.
- New software packages, more user-friendly, with attractive interfaces, directed as much towards decision makers as professionals analysts, but much more expensive.

The main objective of DM is to use the discovered knowledge for purposes of explaining current behavior, predicting future outcomes, or providing support for business decision. Data mining enables corporations and government agencies to analyze massive volumes of data quickly and relatively inexpensively. Today, mining can be performed on many types of data, including those in structured, textual, spatial, Web, or multimedia forms. "Data mining is the process of discovering advantageous patterns in data." [7].

Most data-mining methods (techniques) are based on very-well tested techniques from machine learning, pattern recognition, and statistics: classification, clustering, regression, etc... Data mining techniques are applicable to wide variety of problem areas. Some of these techniques are:

- **Classification** is a supervised technique that maps (classifies) data object into one of several predefined classes, i.e. given an object and its input attributes, classification output is one of possible mutually exclusive classes. The aim of classification task is to discover some kind of relationship between inputs attributes and output class, so that discovered knowledge can be used to predict the class of new unknown object.

- **Regression** is considered to be a supervised learning technique to build more or less transparent model, where the output is a continuous numerical value or vector of such values rather than discrete class.

- **Clustering** is unsupervised learning technique, which aims at finding clusters of similar objects sharing number of interesting properties.

- **Dependency modeling** consists of discovering model which describes significant dependencies among attributes.

- **Change and deviation detection** is the task of focusing on discovering most significant changes or deviations in data between actual content of data and its expected content (previously measured) or normative values.

- **Summarization** aims at producing compact and characteristic descriptions for a given set of data. It can take of multiple forms such as; numerical (simple descriptive statistical measures like means, standard deviations…), graphical forms (histograms, scatter plots…), or on the form of “if-then” rules.

## 2 Supervised learning

Supervised learning is learning process that supplied with a set of example. The set of examples consists of the input data along with the correct output (class) for each example. "The supervised learning paradigm employs a teacher in the machine learning..."
process.” [3]. Some examples of the well known supervised learning models include back propagation in neural networks, K-nearest neighbor, minimum entropy, and decision trees.

3 Unsupervised learning
In unsupervised learning there is no teacher nor is the data pre-classified. So, the algorithm is never giving training set and is basically left on its own to classify its inputs. "The unsupervised learning paradigm has no external teacher to oversee the training process, and the system forms (natural grouping) of the input patterns.” [3]. One of the most well-known unsupervised methods is clustering. In unsupervised learning, the final achieved result reflects the input data in a more objectively manner and the disadvantage of such learning process is that the achieved classes are not necessarily have subjective meaning.

4 Classification and Decision tree
Decision tree is a classification scheme that can be used to produce classification rules. In this section we will review some basic ideas of classification and the development of decision trees algorithm. The theoretical and practical aspects of the ID3 algorithm will also be presented and the features of ID3 will be explained in details. The way to deal with continuous valued attributes will be present. The design and implementation of our system will be demonstrated as well. Our system is unsupervised version of the ID3 and it is developed using Visual Basic.Net.

4.1 Classification background
With enormous amounts of data stored in databases and data warehouses, it is increasingly important to develop powerful tools for data analysis to turn such data into useful knowledge that can be used decision-making. One of the most well studied data mining functionalities is classification due to its wide used in many domains. "Classification is an important data mining problem. Given a training database of records, each tagged with a class label." [8]. The task of classification is first step to build a model (classifier) from the given data (pre-classified data objects) and second step is to use the model to predict or classify unknown data objects. The aim of a classification problem is to classify transactions into one of a discrete set of possible categories.

The input is a structured database comprised of attribute-value pairs. Each row of the database is a transaction and each column is an attribute taking on different values. One of the attributes in the database is designated as the class attribute; the set of possible values for this attribute being the classes.

Classification is a data mining technique that typically involves three phases, learning phase, testing phase and application phase. The learning model or classifier is built during learning phase. It may be in form of classification rules, decision tree, or mathematical formula. Since, class label of each training sample is provided, this approach is known as supervised learning. In testing phase test data are used to assess the accuracy of classifier. If classifier passes the test phase, it is used for classification of new, unclassified data tuples. The application phase, the classifier predicts class label for these new data objects. According to [2], classification has been applied in many fields, such as medical diagnosis, credit approval, customer segmentation and fraud detection.

There are several techniques (methods) of classification:
- Classification by decision tree induction such as: ID3 (Iterative Dichotomizer 3rd), C4.5, SLIQ, SPRINT and rainforest algorithms.
- Bayesian classification by the use of Bayes theorem.
- Classification by back propagation in the area of NN.
- Classification based on the concepts from association rule mining.
- Other classification methods: KNN classifiers, case based reasoning, genetic algorithms, rough set approach, fuzzy set approaches.

4.2 Decision tree algorithm
A decision tree is a flow chart like tree structure. The top most node in the tree is the root node. Each node in the tree specifies a test on some attribute and each branch descending from the node corresponds to one of the possible values of the attribute except for the terminal nodes that represent the class. An instance is classified by starting at the root node of the tree, testing the attribute specified by the given node, then moving down the tree branch corresponding to the value of the attribute in the given example. This process is repeated for the sub tree rooted at the current node. In order to classify an unknown sample, the attribute values of the sample are tested against the decision tree. A path is
traced from the root to a leaf node that holds the class for that instance.

According to [9], Top-Down Induction of Decision Tree (TDIDT) is general purpose systems which classify sets of examples based on their attribute values pairs. The TDIDT algorithm can be rerun to include new example of the data sets. While this is useful feature, it is also time consuming. One of earliest TDIDT algorithms is the Concept Learning System (CLS) by Hunt in 1966. The algorithm works by presenting system with training data from which top-down decision tree is developed based on frequency of information.

In 1986, Quinlan had modified CLS algorithm by enhancing it by the addition of the concept of windowing and information-based measure called entropy. The entropy is used to select the best attribute to split the data into two subsets, so every time the produced decision tree will be the same. The concept of windowing is used to ensure that all the cases in the data are correctly classified.

According to [10], there are several reasons that make decision tree very attracting learning tool. Such as:

- Decision tree learning is a mature technology. It has been in existence for 20+ years, has been applied to various real world problems, and the learning algorithm has been improved by several significant modifications.
- The basic algorithm and its underlying principles are easily understood.
- It is easy to apply decision tree learning to a wide range of problems.
- Several good, easy to use decision tree learning packages are available.
- It is easy to convert the induced decision tree to a set of rules, which are much easier for human to evaluate and manipulate, and to be incorporated into an existing rule based systems than other representations.

5 The ID3 algorithm

According to [9], the ID3 algorithm is a decision tree building algorithm which determines classification of objects by testing values of their properties. It builds tree in top down fashion, starting from set of objects and specification of properties. At each node of tree, the properties are tested and the result is used to partition data object set. This process is recursively carried out till each subset of the decision tree is homogeneous. In other words it contains objects belonging to same category. This then becomes leaf node. At each node of the tree, the tested property is chosen on bases of information theoretic criteria that seek to maximize the information gain and the minimize entropy. In simpler terms, the chosen property is the one that divides the set of objects in the most possible homogeneous subsets. The ID3 algorithm has been successfully applied to wide variety of machine learning problems. It is well known algorithm, however such approach has some limitations.

In ID3, windowing is to select a random subset of the training set to be used to build the initial tree. The remaining input cases are then classified using the tree. If the tree gives correct classification for these input cases then it is accepted for training set and the process ends. If this is not the case then the misclassified cases are appended to the window and the process continues until the tree gives the correct classification.

The information theoretic heuristic is used to produce shallower trees by deciding an order in which to select attributes. The first stage in applying the information theoretic heuristic is to calculate the proportions of positive and negative training cases that are currently available at a node. In the case of the root node this is all the cases in the training set. A value known as the information needed for the node is calculated using the following formula where \( p \) is the proportion of positive cases and \( q \) is the proportion of negative cases at the node:

\[
- p \log_2 p - q \log_2 q
\]

The basic algorithm of ID3

According to [11, 12, 13, and 14], given a set of examples \( S \), each of which is described by number of attributes along with the class attribute \( C \), the basic pseudo code for the ID3 algorithm is:

- If (all examples in \( S \) belong to class \( C \)) then make leaf labeled \( C \)
- Else select the “most informative” attribute \( A \)
- Partition \( S \) according to \( A \)’s values \( (v_1, \ldots, v_n) \)
- Recursively construct sub-trees \( T_1 \), \( T_2 \), \( T_n \) for each subset of \( S \).

ID3 uses a statistical property, called information gain measure, to select among the candidates attributes at each step while growing the tree. To
define the concept of information gain measure, it uses a measure commonly used in information theory, called entropy. The entropy is calculated by:

\[
\text{Entropy}(S) = \sum_{i=1}^{c} - p_i \log_2 p_i
\]

Where \( S \) is a set, consisting of \( s \) data samples, \( p_i \) is the portion of \( S \) belonging to the class \( i \). Notice that the entropy is 0 when all members of \( S \) belong to the same class and the entropy is 1 when the collection contains an equal number of positive and negative examples. If the collection contains unequal numbers of positive and negative examples, the entropy is between 0 and 1. In all calculations involving entropy, the outcome of \((0 \log 0)\) is defined to be 0. With the Information gain measure, given entropy as a measure of the impurity in a collection of training examples, a measure of effectiveness of an attribute in classifying the training data can be defined. This measure is called information gain and is the expected reduction in entropy caused by partitioning the examples according to this attribute. More precisely, the information gain is \( \text{Gain}(S, A) \) of an attribute \( A \), relative to a collection of examples \( S \), is defined as:

\[
\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Value}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)
\]

Where values of \( A \) is the set of all possible values for attribute \( A \), and \( S_v \) is the subset of \( S \) for which attribute \( A \) has value \( v \). The first term in the equation is the entropy of the original collection \( S \), and the second term is the expected value of the entropy after \( S \) is partitioned, using attribute \( A \). \( \text{Gain}(S, A) \) is the expected reduction in entropy caused by knowing the value of attribute \( A \).

Therefore the attribute having the highest information gain is to be preferred in favor of the others. Information gain is precisely the measure used by ID3 to select the best attribute at each step in growing the decision tree.

As an example indicated in [4, 9, 13, 15, 16, 17, 18 and 19], consider decision model to determine whether the weather is amenable to play baseball. Historic data of observations over period of two weeks is available to build a model as depicted in table 1.

<table>
<thead>
<tr>
<th>Day</th>
<th>outlook</th>
<th>temperature</th>
<th>humidity</th>
<th>wind</th>
<th>play ball</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>sunny</td>
<td>Hot</td>
<td>high</td>
<td>weak</td>
<td>no</td>
</tr>
<tr>
<td>D2</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>strong</td>
<td>no</td>
</tr>
<tr>
<td>D3</td>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>weak</td>
<td>yes</td>
</tr>
<tr>
<td>D4</td>
<td>rain</td>
<td>mild</td>
<td>high</td>
<td>weak</td>
<td>yes</td>
</tr>
<tr>
<td>D5</td>
<td>rain</td>
<td>cool</td>
<td>normal</td>
<td>weak</td>
<td>yes</td>
</tr>
<tr>
<td>D6</td>
<td>rain</td>
<td>cool</td>
<td>normal</td>
<td>strong</td>
<td>no</td>
</tr>
<tr>
<td>D7</td>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>strong</td>
<td>no</td>
</tr>
<tr>
<td>D8</td>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>weak</td>
<td>no</td>
</tr>
<tr>
<td>D9</td>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>weak</td>
<td>yes</td>
</tr>
<tr>
<td>D10</td>
<td>rain</td>
<td>mild</td>
<td>normal</td>
<td>weak</td>
<td>yes</td>
</tr>
<tr>
<td>D11</td>
<td>mild</td>
<td>normal</td>
<td>strong</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>D12</td>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>strong</td>
<td>no</td>
</tr>
<tr>
<td>D13</td>
<td>overcast</td>
<td>hot</td>
<td>normal</td>
<td>weak</td>
<td>yes</td>
</tr>
</tbody>
</table>

Table 1: Sample data to build a decision tree using ID3 algorithm.

The weather data attributes are: outlook, temperature, humidity, and wind speed. The target class is the classification of the given day as being suitable (yes) or not suitable (no). The domains of each of the attributes are:

- outlook = (sunny, overcast, rain).
- temperature = (hot, mild, cool).
- humidity = (high, normal).
- wind = (weak, strong).

To determine attribute that would be root node for the decision tree; the gain is calculated for all four attributes. First, we must calculate the entropy for all examples, with \( S \) by using the gain equation as follows:

\[
\text{Entropy}(S) = - \frac{9}{14} \log_2 \left( \frac{9}{14} \right) - \frac{5}{14} \log_2 \left( \frac{5}{14} \right) = 0.940
\]

Where:

\[
\text{Entropy}(\text{weak}) = - \frac{6}{8} \log_2 \left( \frac{6}{8} \right) - \frac{2}{8} \log_2 \left( \frac{2}{8} \right) = 0.811
\]

\[
\text{Entropy}(\text{strong}) = - \frac{3}{6} \log_2 \left( \frac{3}{6} \right) - \frac{3}{6} \log_2 \left( \frac{3}{6} \right) = 1.00
\]

\[
\text{Gain}(S, \text{wind}) = \text{Entropy}(S) - \frac{8}{14} \times \text{Entropy}(\text{weak}) - \frac{6}{14} \times \text{Entropy}(\text{strong}) = 0.940 - \left( \frac{8}{14} \times 0.811 \right) = 0.048
\]

After that we can calculate the information gain for all four attributes by the use of the previous equation as follows:

\[
\text{Entropy}(\text{weak}) = - \frac{6}{8} \log_2 \left( \frac{6}{8} \right) - \frac{2}{8} \log_2 \left( \frac{2}{8} \right) = 0.811
\]

\[
\text{Entropy}(\text{strong}) = - \frac{3}{6} \log_2 \left( \frac{3}{6} \right) - \frac{3}{6} \log_2 \left( \frac{3}{6} \right) = 1.00
\]

\[
\text{Gain}(S, \text{wind}) = \text{Entropy}(S) - \frac{8}{14} \times \text{Entropy}(\text{weak}) - \frac{6}{14} \times \text{Entropy}(\text{strong}) = 0.940 - \left( \frac{8}{14} \times 0.811 \right) = 0.048
\]

Similarly the gain is calculated for the other attributes,

\[
\text{Gain}(S, \text{outlook}) = 0.246
\]

\[
\text{Gain}(S, \text{temperature}) = 0.029
\]

\[
\text{Gain}(S, \text{humidity}) = 0.151
\]
Because the outlook attribute has the highest gain, therefore it is used as the decision tree root node. The outlook attribute has three possible values; the root node has three branches labeled with sunny, overcast and rain.

The next step is to develop the sub tree, one level at a time, starting from the left (under sunny) using the remaining attributes namely humidity, temperature and wind.

The calculation of gain is carried out for each of the attributes given the value of the previous value of the attribute. The final decision tree obtained as the result of ID3 algorithm is depicted in figure 1:

The following rules are generated from the above decision tree:

- IF outlook= overcast THEN play ball= yes
- IF outlook= rain ∧ wind= strong THEN play ball= yes
- IF outlook= rain ∧ wind= weak THEN play ball= yes
- IF outlook= sunny ∧ humidity= high THEN play ball= no
- IF outlook= rain ∧ humidity= high THEN play ball= no

5.1 Features of ID3
The most important feature of ID3 algorithm is its capability to break down a complex decision tree into a collection of simpler decision trees. Thus it provides a solution which is often easier to interpret. In addition, some of other important features are:
- Each attribute can provide at most one condition on a given path. This also contributes to comprehensibility of the resulted knowledge.
- Complete hypothesis space: any finite discrete valued function can be expressed.
- Incomplete search: searches incompletely through the hypothesis space until the tree is consistent with the data.
- Single hypothesis: only one current hypothesis (the best one) is maintained.
- No backtracking: once an attribute is selected, this can’t be changed.
- Full training set: attributes are selection by computing information gain on the full training set.

6 Design and implementation
6.1 Different types of attributes
Due to the fact that the ID3 algorithm deal with discrete valued attributes and we have decided to limit the number of values per attribute to four. The reason for that is to have simple decision trees and rules. If the number of different values per attribute is greater than four then the (NIU-ID3) system will reduce it to four by the use of discretization (normalization) techniques. This process is carried out on numerical attributes and for symbolic attribute values the same process will be carried out after coding the attributes values. For a continuous valued attribute A, the system partitions the attribute values into four intervals by:

\[
\text{Length of interval} = \frac{\text{Max}_A - \text{Min}_A}{C}
\]

Where: MaxA is the maximum value of attribute A, MinA is the minimum value of attribute A and C is the number of intervals (default value is 4).

For example: if we have a numerical attribute with the following values: 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 76, 77, 79.

We can calculate the interval length as: \((79 – 43) / 4 = 9\). The intervals and the corresponding values can be seen in the diagram below:
Table 2 depicts the original values with their corresponding new values.

<table>
<thead>
<tr>
<th>Original values</th>
<th>Corresponding values</th>
</tr>
</thead>
<tbody>
<tr>
<td>43, 44, 45, 46, 47, 48, 49, 50, 51</td>
<td>11</td>
</tr>
<tr>
<td>52, 53, 54, 55, 56, 57, 58, 59, 60</td>
<td>12</td>
</tr>
<tr>
<td>61, 62, 63, 64, 65, 66, 67, 68, 69</td>
<td>13</td>
</tr>
<tr>
<td>70, 71, 72, 73, 74, 76, 77, 79</td>
<td>14</td>
</tr>
</tbody>
</table>

Table 2: The original values with their corresponding new values.

Another example is for a symbolic attribute that has the following values: BB, CC, DD, EE, FF, GG and HH.

Such attribute values is dealt with in the following way:
- First, the values will be coded as: 01, 02, 03, 04, 05, 06 and 07.
- Second, we calculate the interval width as: (7 - 1) / 4 = 1.5.

The intervals and the corresponding values can be seen in the diagram below:

```
   25 4 55 7
 11 12 13 14
```

Table 3 depicts the original values with their corresponding codes and new values.

<table>
<thead>
<tr>
<th>Original values</th>
<th>Corresponding values</th>
<th>Corresponding discretized values</th>
</tr>
</thead>
<tbody>
<tr>
<td>BB, CC</td>
<td>01, 02</td>
<td>11</td>
</tr>
<tr>
<td>DD</td>
<td>03</td>
<td>12</td>
</tr>
<tr>
<td>EE, FF</td>
<td>04, 05</td>
<td>13</td>
</tr>
<tr>
<td>GG, HH</td>
<td>06, 07</td>
<td>14</td>
</tr>
</tbody>
</table>

Table 3: The original values with their corresponding codes and new discretized values.

Each code for the corresponding values consists of two digits the first represents the attribute number and the second represents the interval number (attribute's value). For example, the code 13 is for the third value of the first attribute.

### 6.2 System Design

We will demonstrate the design and implementation of our system, which is a new implementation of the ID3 algorithm to make it work in unsupervised manner by building a font-end to it. Our system is called New Implementation of Unsupervised ID3 (NIU-ID3). We will also give an overview of the over all architecture of the NIU-ID3, the tasks that the system can deal with, the data format that it uses and the results that is produced from it.

The NIU-ID3 system accomplishes its task via several stages that are executed in a serial fashion in the form of a wizard form (Data Mining Wizard). These stages are grouped into four main components:

- Data set is the data to be used to discover knowledge from.
- Preprocessing is the process of preparing the data for classification if the data is continuous or unlabeled.
- Classification is the classification process (ID3 algorithm) to produce the knowledge.
- Knowledge is the results discovered by classification process.

The NIU-ID3 system can manipulate with two types of data files; that is a text data or Access databases. It can also preprocess unlabeled data (clustering of data objects) and process label data (classification). Our system can discover knowledge into two different formats, namely; decision trees and classification rules.
Building a NIU-ID3 model: The model building phase can start by building a simulated model of the problem. This model will provide a clear understanding of the problem in question. From the literature, there are three perspectives used in the development of simulated models and these are:

- Use some Graphical User Interface (GUI) tools to develop the simulated model on the screen. Use arcs to connect the system components to create the logical model and the run simulated model. “In most cases, due to the limitation of the simulation program under use, some simplifications and approximations to the model may be required. Such simplifications or approximations can be very costly.” [20].

- Support the belief that any simulation program would not be able to model all tasks of systems without the need to make some modification. “This suggests that models should be developed from scratch by using a simulation modeling language.” [20]. This approach may increase the time needed to produce the system and may divert the developer to pay more attention to the programming challenges than understanding the system.

- Focuses on the use of GUI that will automatically generate code with the possibility of the developer intervention to make some changes to the code to match the system requirements. This is a very popular practice because it reduces the need time to produce the system, on the other hand code modification is a tedious task.

Figure-2: General architecture of NIU-ID3 system.

Figure-3: Build process of the NIU-ID3 system.

The model building phase consists of a number of steps, as follows:

1. Data loading which consists of two sub steps:
   - In loading data, our system deals with two types of data files; text files data and Access database files. Our system will ask the user to select the file type. The system initially designed to deal with only Access database files, so if the loaded data is not Access database file then a preprocessing step will be converted it to Access database file. So, the loaded database is called the training set. This data set consists of a set of tuples, each tuple consists of a number of values and an additional attribute called class attribute. At this stage by the use of ADO.NET technique to establish the connection between the database and the system. The ADO.NET is a part of the base class library that is included with the Microsoft .NET framework, and it is a set of computer software components that can be used by programmers to access the data. In reality, ADO.NET can not be connected with data sources directly, but it needs .NET data providers. Here we use OLE DB.NET data provider, and Microsoft® universal data access
component 2.5 or 2.6 (MDAC 2.5 or MDAC 2.6).

- The data selection sub step will display all table's names that are available of type Access to the user, so he/she can select one of them and the system starts the discovery process. There are some conditions on the data file that is loaded to our system. We will explain them in the next chapter.

2. Preprocessing: consists of four sub steps:
   - Check missing values: if there are some missing values in the loaded data our system will ask the user to change the data file due to the fact that the system is not accommodated to deal with missing values.
   - Converting a text file to Access database file: If the loaded data file is of type text, the system will convert it into an Access database file.
   - Data labeling: if the loaded data is un-labeled then the system will label it via the clustering component of the system.
   - Continuous and discrete valued attributes: The ID3 does not work with continuous valued attributes. If the number of values per attribute is more than four then the system will divide the range of attribute values into intervals using the length equation.

3. Classification is the process of building a model or a function that can be used to predict the class of unclassified data objects. In our system we use the ID3 algorithm for this task.

4. Knowledge is the end result that is produced from our system. The end result can be in one of different form such as; decision tree, decision rules or more general simplified rules.

In our system the end results can be saved in text files if the user desire along with the data.

7 Clustering Front-End module

The goal of NIU-ID3 system is to build a decision tree and to extract classification rules (decision rules) from the provided data set. Such rules can be used for prediction. The classification module of our system (ID3 algorithm) needs a labeled data set to train the classifier. Such data set consists of a number of records, each of which consists of several attributes. Attributes values will be dealt with accordingly. There is one distinguished attribute called the dependent (class) attribute.

In the case of un-labeled data, the clustering module will be used to carry out the labeling process. In this chapter we will focused or concerned on an one algorithm of clustering techniques, which called fuzzy k-means algorithm (extension of the normal k-means clustering method), and its software package, which called “FuzME program” to make it as a Front-End module to our system.

7.1 Clustering methods

In general, clustering is the process of grouping data objects into groups or clusters such that:
- Each group or cluster is homogeneous or compact with respect to certain characteristics. That is, objects in each group are similar to each other.
- Each group should be different from other groups with respect to the same characteristics; that is, objects of one group should be different from the objects of other groups.

Clustering is an unsupervised learning technique used to divide data sets into groups or clusters. These clusters can be viewed as a group of elements which are more similar to each other than elements belonging to other groups. An alternative definition of a cluster is a region with a relatively high density of points, separated from other clusters by a region with a relatively low density of points. "Clustering is a useful technique for the discovery of some knowledge from a dataset. It maps a data item into one of several clusters, where clusters are natural groupings of data items based on similarity metrics or probability density models. Clustering pertains to unsupervised learning, when data with class labels are not available." [21].

In general, data clustering algorithms can be categorized as hierarchical or partitioning.
Hierarchical algorithms find successive clusters using previously established clusters, whereas partitioning algorithms finds all clusters at once.

Our system clustering algorithm: As it has been mentioned previously, our system works in unsupervised fashion which needs to label unlabeled data before it can generate the knowledge in the form of decision tree. To label unlabeled data, our system uses a program called FuzME which based on the clustering algorithm called fuzzy k-means algorithm.

**Fuzzy k-means algorithm**: In fuzzy clustering, each data object has a degree of belonging in each cluster. So each data object belongs to all clusters with varying degree of membership. Thus, data objects on the edge of a cluster belong to the cluster with lesser degree than data objects that are in the center of the cluster.

In fuzzy K-means clustering, the centroid of a cluster is the average weight of the degree of belonging to the cluster. "Fuzzy-k-means clustering is an extension of the normal, crisp-k-means clustering method to account for uncertainties associated with class boundaries and class membership. As in k-means clustering, the iterative procedure minimizes the within-class sum of squares, but each object (or cell on a map) is assigned a continuous class membership value ranging from 0 to 1 in all classes, rather than a single class membership value of 0 or 1 used in the normal k-means clustering method (DeGrujijter and McBratney, 1988). Fuzzy-k-means clustering was conducted using the FuzME program (Minasny and McBratney, 2002) with Mahalanobis distance and a fuzzy exponent of 1.2. Each cell was assigned to a single yield category based on the highest fuzzy membership value at this particular location."[22].

**FuzME program**: In our system NIU-ID3 uses FuzME program (based on Fuzzy k-means algorithm) as front-end module to label unlabeled data objects. FuzME program was published and presented by Minasny B. and McBratney A. in the year of 2002 from the Australian Centre for Precision Agriculture (ACPA) at the University of Sydney, Australia. More information about the program can found on the following web site http://www.usyd.edu.au/su/agric/acpa.

**Input to FuzME program**: According to [23], the data file that can be accepted as input to FuzME program must be in text format, where the first row must be start with the word "id" followed by attributes names. The second and consecutive rows start with the id as a number for each data object followed by the values of the attributes separated by a single space. As an example, figure-4, depicts an input to the FuzME program. The data file consists of 14 instances, each of which consists of four attributes.

<table>
<thead>
<tr>
<th>Id</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
</tr>
<tr>
<td>2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
</tr>
<tr>
<td>3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
</tr>
<tr>
<td>4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
</tr>
<tr>
<td>5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
</tr>
<tr>
<td>6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
</tr>
<tr>
<td>7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
</tr>
<tr>
<td>8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
</tr>
<tr>
<td>9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
</tr>
<tr>
<td>10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
</tr>
<tr>
<td>11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
</tr>
<tr>
<td>12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
</tr>
<tr>
<td>13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
</tr>
<tr>
<td>14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
</tr>
</tbody>
</table>

Figure-4: Text file format present to FuzME program.

**Output to FuzME program**: The execution of the FuzME program will generate in many text files as a result (output). The produced text files are: number of files each of them is named as n_class, where n is the number of produced (i.e. 2_class.txt, 3_class.txt, 4_class.txt, 5_class.txt, etc ...), number of files each of them is named as n_dscr that contains description of the produced clustering files (i.e. 2_dscr.txt, 3_dscr.txt, 4_dscr.txt, 5_dscr.txt, etc ...), Control, FuzMeout, pca, and summary. Our system needs only to use only one n_class depending on the number of classes the number of cluster that the user had specified.

The output file of the FuzME program is a text file that consists of each object and which cluster that the data object falls in. For example the output text file depicted in figure-5, is the result from executing the FuzME program on the data of figure-4 and the desired number of cluster is 2, which are coded as 2a and 2b respectively. So the output file name is 2_class.txt.

So, our system needs only to use the second (class) and add it to the original data text file as depicted in figure-6 to be used as input to the system.
Front-End module Implementation: In our system, we link the FuzME program with the implementation of the ID3 algorithm by a shell function, which is considered as a technique of vb.net 2005 used to access to link external objects. "A Shell link is a data object that contains information used to access another object in the Shell's namespace that is, any object visible through Microsoft Windows Explorer. The types of objects that can be accessed through Shell links include files, folders, disk drives, and printers. A Shell link allows a user or an application to access an object from anywhere in the namespace."[24]. The external object that can be accessed or linked to must reside on the current computer disk drives.

8 Experiments and results
The purpose of this study is to produce different forms of knowledge and to implement the ID3 algorithm in supervised and unsupervised fashion using Visual Basic.net 2005. In this paper, we demonstrate the obtained results of applying our system to different types and sizes of data from a variety of domains. To test the effectiveness of our system, we have conducted some experiments using many real data sets (databases). We used real data in the experiments that available on public domain (the Internet). All of our experiments are performed on a PC with Microsoft Windows XP professional operating system (service pack 2) with a processor speed of 2.7 GHz, RAM size of 512MB and hard disk of size 80 GB. The PC computer is also equipped with Microsoft® Universal Data Access Component 2.5 or 2.6 (MDAC 2.5 or MDAC 2.6) also a reference to Microsoft® Service Component OLEDB Service Component 1.0 stored as; “C:\Program Files\Common Files\System\OLE DBoledb32.dll. Since our system works with Access database and with text files after some preprocessing in converting the text files into Access database files. Then the actual loading of the data to the system can take place.

Access database file: the Access database file name can be numerical or symbolic, and it must be consists of at least one relational table and the name of the relational table can be numerical or symbolic. The relational table, as depicted in figure-7, consists of number of tuples, each of which has number of attributes. The type of attributes can be numeric or symbolic discrete or continuous. Numerical attribute values can be real or integer. Each attribute must have a value, so no missing values are allowed. A symbolic attribute value can be symbols, numbers or both of them. Each tuple corresponds to one instance (example).

Text file specifications: The text file name accepted by our system can be numerical or symbolic, and it can consist of any type of data (numerical or symbolic). Each text file consists of number of rows or lines as depicted in figure-8. The first line consists of the attribute names. The second line and subsequent ones consist of the data. The data values
can be of any type (i.e. numerical or symbolic). A numerical attribute values can be real or integer. A symbolic attribute value can be symbols, numbers or both. The values in each row must be separated by one space. Each row corresponds to one example or instance and it consists of the values of the attributes. Also, here no missing values are allowed.

Figure-8: Text file format.

Our experiments: We have conducted a total of 8 experiments with different data sets. The differences in the data sets are in data types and sizes. The results of these experiments are summarized in table-3 as follow:

<table>
<thead>
<tr>
<th>Experiment No.</th>
<th>Data set name</th>
<th>Data type</th>
<th>Data labeled or unlabeled</th>
<th>No. of tuples</th>
<th>No. of attributes including class attribute</th>
<th>No. of tree levels</th>
<th>No. of decision rules</th>
<th>No. of simplified rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Play tennis</td>
<td>Symbolic/ discrete</td>
<td>Labeled</td>
<td>14</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Club membership</td>
<td>Symbolic/ discrete</td>
<td>Labeled</td>
<td>12</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Stock market</td>
<td>Symbolic/ discrete</td>
<td>Labeled</td>
<td>10</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>London stock market</td>
<td>Symbolic/ discrete</td>
<td>Labeled</td>
<td>6</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Titanic</td>
<td>Symbolic/ discrete</td>
<td>Labeled</td>
<td>2201</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>Iris</td>
<td>Symbolic/ continuous/ numerical</td>
<td>Labeled</td>
<td>150</td>
<td>5</td>
<td>3</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>Unlabeled play tennis</td>
<td>Symbolic/ discrete</td>
<td>Unlabeled</td>
<td>14</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>Unlabeled Titanic</td>
<td>Symbolic/ discrete</td>
<td>Unlabeled</td>
<td>2201</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

Table-3: Summary of the experiments results.

Depending on the obtained results from our system, the author would like to make the following remarks:

1. The results obtained from all experiments giving us decision trees with three levels, because we used discretization techniques to reduce the number of values per attribute to 4.

2. The results obtained from experiments no. 1, 2, 3 and 4 are the same results as in [8, 9, 12, 13, 15, 16, 17, 18, and 19].

3. We had no previous results for experiment no. 5, so we could not compare it with previous ones and we think that this result is satisfactory depending on the accurate results that we have obtained in experiments 1 to 4.

4. For experiment no. 6, there were some differences between our results and the results published in [25, 26 and 27]. The differences in results could be due to:
   - The discretization (normalization) technique used in [25].
   - C4.5 (classification) algorithm and the discretization (normalization) technique used in [26 and 27].

5. The results obtained from experiments no. 7 and 8 are different from the ones published in experiments no. 1 and 5, this could be due to the labeling process via the use of FuzMe Program.

9 Conclusion

In this paper, we have added a front-end to the ID3 algorithm, so it works in unsupervised mode. Generally, our system consisted of two parts: the first part is the implementation of the ID3 algorithm to be used in classifying labeled data sets; the second part is used to label the unlabeled data sets using FuzMe Program. Our system, NIU-ID3 has been tested with a number of different data sets (labeled, unlabeled and different data types and sizes). We believe that our system will enable decision makers such as; managers, analysts, engineers, physicians, etc... to take the correct decisions.

From our system's results, we can conclude that:

1. Our system has produced very accurate results such as the ones in experiments 1, 2, 3, and 4.

2. The decision trees produced by our system were very clear to visualize.

3. The rules produced by our system were simple to understand and clear to visualize.

4. We think that the results we obtained for experiment no. 5 is satisfactory.
5. The differences in results of experiment 6, with the original results from [25, 26 and 27] could be due to the discretization techniques used and the difference in the classification algorithm or the labeling process such as experiment 7 and 8.

References:


[27] Xue Li. A Tutorial on Induction of Decision Trees, School of information technology and electrical engineering, tutorial, University Of Queensland, 2002.