International Journal on New Computer Architectures and Their Applications (IJNCAA) 1(1): 61-73
The Society of Digital Information and Wireless Communications, 2011 (ISSN 2220-9085)

# Database Preprocessing and Comparison between Data Mining Methods

Yas A. Alsultanny
College of Graduate Studies-Arabian Gulf University
Manama, P.O Box 26671, Kingdom of Bahrain
alsultanny@hotmail.com

## ABSTRACT

Database preprocessing is very important to utilize memory usage, compression is one of the preprocessing needed to reduce the memory required to store and load data for processing, the method of compression introduced in this paper was tested, by using proposed examples to show the effect of repetition in database, as well as the size of database, the results showed that as the repetition increased the compression ratio will be increased. The compression is one of the important activities for data preprocessing before implementing data mining. Data mining methods such as Na¨ıve Bayes, Nearest Neighbor and Decision Tree are tested. The implementation of the three methods showed that Na¨ıve Bayes method is effectively used when the data attributes are categorized, and it can be used successfully in machine learning. The Nearest Neighbor is most suitable when the data attributes are continuous or categorized. The third method tested is the Decision Tree, it is a simple predictive method implemented by using simple rule methods in data classification. The success of data mining implementation depends on the completeness of database, that represented by data warehouse, that must be organized by using the important characteristics of data warehouse.

## KEYWORDS

Data mining, Preprocessing, Nearest Neighbour, Na¨ıve Bayes, Decision tree.

## 1 INTRODUCTION

The extraction of useful and non-trivial information from the huge amount of data that is possible to collect in many and diverse fields of science, business and engineering, is called Data Mining (DM). DM is part of a bigger framework, referred to as Knowledge Discovery in Databases (KDD); this covers a complex process, from data preparation to knowledge modeling. Data compression is one of the preparations methods which are needed to compress the huge amount of database.

Data mining is a process that is used to identify hider, unexpected pattern or relationships in large quantities of data. Historically, the notion of finding useful patterns in data has been given a variety of names, including data mining, knowledge extraction, information discovery, information harvesting, data archaeology, and data pattern processing. The term data mining has mostly been used by statisticians, data analysts and the Management Information Systems (MIS) communities. The phrase knowledge discovery in databases was coined at the first KDD to emphasize that knowledge is the end product of a data-driven discovery. It has been popularized in the artificial intelligence and machine-learning fields. Figure 1 shows an overview of the data mining and KDD process [1].

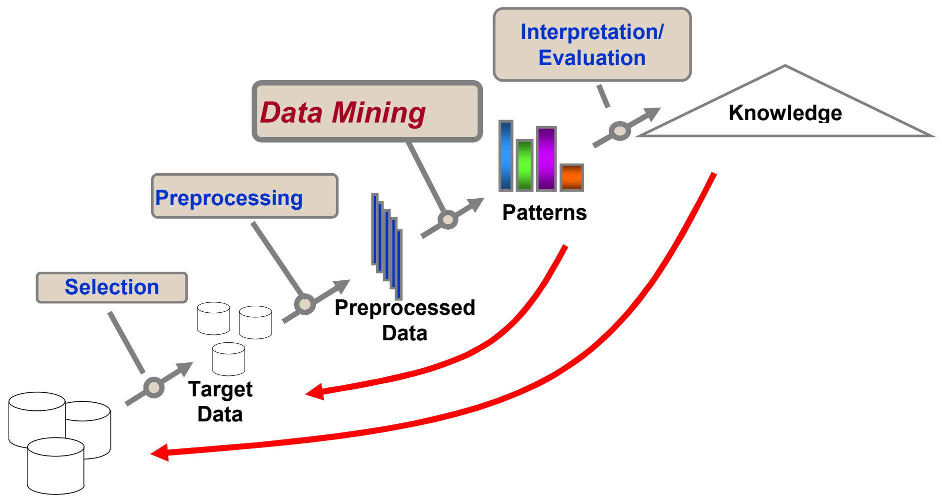


**Figure 1.** An overview of the data mining and KDD process

Data mining predicts future trends and behaviors, allowing business to make proactive, knowledge driven decisions. It moves beyond the analysis of past events provided by retrospective tools typical of decision support systems, answering questions that traditionally where too time consuming to resolve. Data mining scours databases for hidden patterns, finding predictive information that experts might overlook because it falls outside their expectations.

The two high-level primary goals of data mining in practice tend to be prediction and description. The prediction involves using some variables or fields in the database to predict unknown or future values of other variables of interest, and description focuses on finding human-interpretable patterns describing the data. Although the boundaries between prediction and description are not sharp (some of the predictive models can be descriptive, to the degree that they are understandable, and vice versa), the distinction is useful for understanding the overall discovery goal. The goals of prediction and description can be achieved using a variety of particular data-mining methods.

- Classification
- Regression
- Clustering
- Summarization
- Dependency modeling

## 2 DATA WAREHOUSE

Data warehouse is a repository of subject-oriented historical data that is organized to be accessible in a form readily acceptable for analytical processing activities (such as data mining, decision support querying, and other applications) [2].

The major benefits of a data warehouse are:

- The ability to reach data quickly, since they are located in one place.
- The ability to reach data easily and frequently by end users with Web browsers.

Characteristics of data warehousing are:

- Orga*nization*. Data are organized by subject.
- *Consistency*. In the warehouse data will be coded in a consistent manner.
- *Time variant*. The data are kept for many years so they can be used for trends, forecasting, and comparisons over time.
- *Non-volatile*. Once entered into the warehouse, data are not updated.

- *Relational.* Typically the data warehouse uses a relational structure.
- *Client/server.* The data warehouse uses the client/server architecture mainly to provide the end user an easy access to its data.
- *Web-based.* Data warehouses are designed to provide an efficient computing environment for Web-based applications.
- *Integration.* Data from various sources are integrated.
- *Real time.* Most applications of data warehousing are not in real time, it is possible to arrange for real-time capabilities.

It is important to note that if the data warehouse is organized with the above characteristics, the implementation of data mining by using different methods will have high degree of fidelity and can be used in decision making.

## 3 DATABASE MANAGEMENT

The efficient database management systems have been very important assets for management of a large amount of data and especially for effective and efficient retrieval of particular information from a large collection whenever needed. The proliferation of database management systems has also contributed to recent massive gathering of all sorts of information. Today, we have far more information than we can handle: from business transactions and scientific data, to satellite pictures, text reports and military intelligence. Information retrieval is simply not enough anymore for decision-making. Confronted with huge collections of data, we have now created new needs to help us make better managerial choices. These needs are automatic summarization of data, extraction of the "essence" of information stored, and the discovery of patterns in raw data [3].

Information technology cannot be done without using some kinds of data which are at the core of management and marketing operations. However, managing data is difficult for various reasons [1].

- The amount of data increases exponentially with time.
- Data are scattered throughout organizations.
- Data are collected by many individuals using several methods and devices.

Database is typically store large amount of data that must persist over long periods of time. The data is accessed and preprocessed repeatedly during this period. This contrasts with the notion of transient data structures that persist for only a limited time during program execution. Most databases are stored permanently (or persistently) on secondary storage, for the following reasons:

- Generally, database is too large to fit entirely in main memory.
- The circumstance that cause permanent loss of stored data arise less frequently for disk secondary storage devices, as nonvolatile storage whereas main memory is often called volatile storage
- The cost of storage per unit of data is an order of magnitude loss for disk secondary storage then for primary storage [4].

## 4 DATABASE PREPROCESSING

Database preprocessing is very necessary to prepare database to be used in knowledge extraction by data mining. In the following subsections the database compression was implemented to apply data mining methods.

## 4.1 Primary key

Each attribute in the class is represented by a field with a particular data type, and we can apply some constraints to the values we allow into a field. We have overlooked one constraint that is so important that it gets a section at to itself. This involves choosing a primary key for the data. It is imperative that we can always find a particular object (or row or record in a table). This means that all records must be unique; otherwise, you couldn't distinguish two that where identical.
A key is a field, or combination of fields, that is guaranteed to have a unique value for every record in the table. It is possible to learn quite a bit about a problem by considering which fields are likely to be possible keys [5].

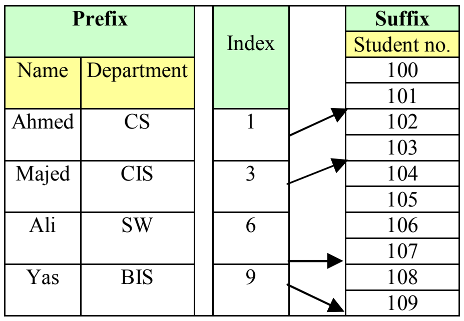
## 4.2 Data Compression

Item sets are often organized in collections in many Frequent Set Counting algorithms. Efficient representation of such collections can lead to important performance improvements. The compressed data structure is based on arrays, as in Figure 2, at each iteration k; the first array (prefix) stores the different prefixes of length k-1, where i is the element of the second array (index), we store the position in the suffix array of the section of suffixes that share the same prefix, in the third array (suffix) all the length-1 suffixes are stored. Therefore, when the item sets in the collection have to be enumerated, we first access the prefix array. Then, from the corresponding entry in the index array we get the section of suffixes, stored in suffix, needed to complete the item sets.

*Example 1*: The database in Figure 2 consists from three attributes (three columns). In this data there are three columns within which some instances have more than one frequency. The only column that doesn't have repeated data is column one that contain the primary key, therefore, this is the column that will be excluded from compression.
Thus, our principle will be to compress data that has frequently repeated instance. The initial step of compression will be to compress data that has frequently repeated entries and once these columns are identified the compression starts. The rate of compression in Figure 2 is (22/30=73.3%).

| Student no. | Name | Department |
|---|---|---|
| 100 | Ahmed | CS |
| 101 | Ahmed | CS |
| 102 | Majed | CIS |
| 103 | Majed | CIS |
| 104 | Ali | SW |
| 105 | Ali | SW |
| 106 | Ali | SW |
| 107 | Yas | BIS |
| 108 | Yas | BIS |
| 109 | Yas | BIS |

a) Uncompressed original data: 3*10=30



b) Compressed data= Prefix + Index + Suffix = (2*4) + 4 + 10= 22

a) Data before compression
b) Data after compression

**Figure 1.** Data Compression

*Example 2*: The data compression ratio is variable and depends on the data itself. Figure 3 shows another examples of data compression, where the data consist from four attributes (four columns), the first column represent the primary key (student no.) will not be compressed, the rate of compression in this example is (26/40=65%).

| Student no. | Name | Department | Course |
|---|---|---|---|
| 100 | Ahmed | CS | C++ |
| 101 | Ahmed | CS | C++ |
| 102 | Majed | CIS | Java |
| 103 | Majed | CIS | Java |
| 104 | Ali | SW | C# |
| 105 | Ali | SW | C# |
| 106 | Ali | SW | C# |
| 107 | Yas | BIS | VB |
| 108 | Yas | BIS | VB |
| 109 | Yas | BIS | VB |

a) Uncompressed original data: 4*10=40

| Prefix | | | Index | Suffix |
|---|---|---|---|---|
| Name | Department | Course | | Student no. |
| | | | | 100 |
| | | | | 101 |
| Ahmed | CS | C++ | 1 | 102 |
| | | | | 103 |
| Majed | CIS | Java | 3 | 104 |
| | | | | 105 |
| Ali | SW | C# | 6 | 106 |
| | | | | 107 |
| Yas | BIS | VB | 9 | 108 |
| | | | | 109 |

b) Compressed data= Prefix + Index + Suffix = (4*3) + 4 + 10= 26

a) Data before compression.
b) Data after compression

**Figure 3.** Data Compression

*Example 3*: As a third example to show the compression with five attributes (five columns), the compression depends on the data itself, and its size. Figure 4 shows the rate of compression in this example is (30/50=60%).

| Student no. | Name | Department | Course | Nationality |
|---|---|---|---|---|
| 100 | Ahmed | CS | C++ | BAH |
| 101 | Ahmed | CS | C++ | BAH |
| 102 | Majed | CIS | Java | KU |
| 103 | Majed | CIS | Java | KU |
| 104 | Ali | SW | C# | KSA |
| 105 | Ali | SW | C# | KSA |
| 106 | Ali | SW | C# | KSA |
| 107 | Yas | BIS | VB | IRQ |
| 108 | Yas | BIS | VB | IRQ |
| 109 | Yas | BIS | VB | IRQ |

a) Uncompressed original data: 5*10=50

| Prefix | | | | Index | Suffix |
|---|---|---|---|---|---|
| | | | | | Student no. |
| Name | Department | Course | Nationality | | 100 |
| | | | | | 101 |
| Ahmed | CS | C++ | BAH | 1 | 102 |
| | | | | | 103 |
| Majed | CIS | Java | KU | 3 | 104 |
| | | | | | 105 |
| Ali | SW | C# | KSA | 6 | 106 |
| | | | | | 107 |
| Yas | BIS | VB | IRQ | 9 | 108 |
| | | | | | 109 |

b) Compressed data= Prefix + Index + Suffix = (4*4) + 4 + 10= 30

a) Data before compression.
b) After data compression

**Figure 4.** Data Compression

Figure 5 shows the compression between the original data and compressed data, where the ratio of compression will be increased with increasing data size, because the probabilities of data repetition will be increased, and this will be suitable with data mining as a preprocessing step, when the database will be huge and cause problems in storing data.

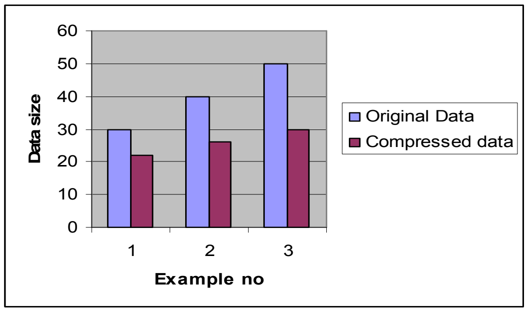


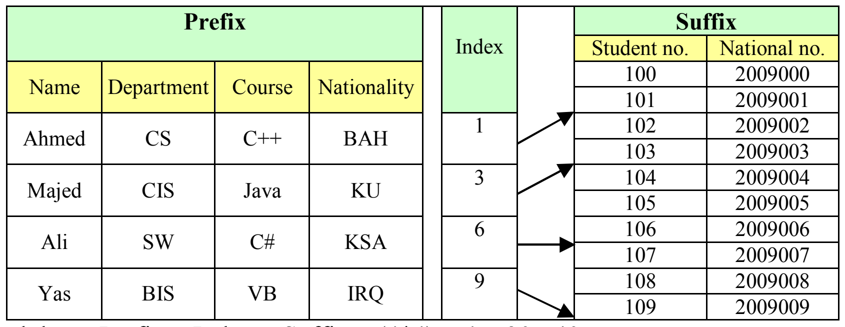
**Figure 5.** Comparison between original data and compressed data.

### 4.3 Two primary key

Sometimes there are two attributes (two columns) or more did not have repeated items in the database, in this case any one of these attributes can be used as a primary key. These two attributes or more cannot be compressed. Figure 6 shows one example of this kind of database the rate of compression in this example is (40/60=66.6%). This is less than the ratio obtained in Figure 5.

| Student no. | National no. | Name | Department | Course | Nationality |
|---|---|---|---|---|---|
| 100 | 2009000 | Ahmed | CS | C++ | BAH |
| 101 | 2009001 | Ahmed | CS | C++ | BAH |
| 102 | 2009002 | Majed | CIS | Java | KU |
| 103 | 2009003 | Majed | CIS | Java | KU |
| 104 | 2009004 | Ali | SW | C# | KSA |
| 105 | 2009005 | Ali | SW | C# | KSA |
| 106 | 2009006 | Ali | SW | C# | KSA |
| 107 | 2009007 | Yas | BIS | VB | IRQ |
| 108 | 2009008 | Yas | BIS | VB | IRQ |
| 109 | 2009009 | Yas | BIS | VB | IRQ |

a) Uncompressed original data: 6*10=60



| Prefix | | | |
|---|---|---|---|
| Name | Department | Course | Nationality |
| Ahmed | CS | C++ | BAH |
| Majed | CIS | Java | KU |
| Ali | SW | C# | KSA |
| Yas | BIS | VB | IRQ |

| Index |
|---|
| 1 |
| 3 |
| 6 |
| 9 |

| Suffix | |
|---|---|
| Student no. | National no. |
| 100 | 2009000 |
| 101 | 2009001 |
| 102 | 2009002 |
| 103 | 2009003 |
| 104 | 2009004 |
| 105 | 2009005 |
| 106 | 2009006 |
| 107 | 2009007 |
| 108 | 2009008 |
| 109 | 2009009 |

b) Compressed data= Prefix + Index + Suffix = (4*4) + 4 + 20= 40
a) Data before compression.
b) After data compression

**Figure 6.** Data Compression with two uncompressed columns

## 5 DATA MINING TECHNOLOGY

Data mining is an important information technology used to identify significant data from vast amounts of records. In other words, it is the process of exposing important hidden patterns in a set of data.
Large volume of data and complexity in problem solving inspire research in data mining and modern heuristics. Data mining (i.e. knowledge discovery) is the process of automating information discovery. It is the process of analyzing data from different perspectives, summarizing it into useful information, and finding different patterns (e.g. classification, regression, and clustering). Many problems are difficult to be solved analytically in a feasible time. Therefore, researchers are trying to find search techniques or heuristics to get a good enough or satisfactory solution in a reasonable time [6], [7].

## 6 DATA MINING

Data mining involves the use of sophisticated data analysis tools to discover previously unknown, valid patterns and relationships in large data sets. These tools can include statistical models, mathematical algorithms, and machine learning methods (algorithms that improve their performance automatically through experience, such as neural networks or decision trees). Consequently, data mining consists of more than collecting and managing data, it also includes analysis and prediction.
Data mining is the principle of sorting through large amounts of data and picking out relevant information. It is usually used by business intelligence organizations, and financial analysts, but it is increasingly used in the sciences to extract information from the enormous data sets generated by modern experimental and observational methods. It has been described as "the nontrivial extraction of implicit, previously unknown, and potentially useful information from data" and "the science of extracting useful information from large data sets or databases" [8, 9, 10 and 11].

## 7 DATABASE CLASSIFICATIONS

Three methods of data mining classification will be tested and compared, these are;

1. Na¨ıve Bayes that can be used when all the attributes are categorical [12].
2. Nearest Neighbour that can be used when all the attributes are continuous [13].
3. A decision tree algorithm is a predictive model that can be viewed as a tree [14].

### 7.1 Na¨ıve Bayes Classifiers

The data that will be used in comparison is the buses arrivals, if the probability of an event, e.g. that the 6.30 p.m. Buses from the city centre to your local station arrives on time, is a number from 0 to 1 inclusive, with 0 indicating ‘impossible’ and 1 indicating ‘certain’. A probability of 0.7 implies that if we conducted a long series of trials, e.g. the arrival recorded at 6.30 p.m. buses day by day for N days, the buses expected to be on time on 0.7 × N days. The longer the series of trials the more reliable this estimate is likely to be.
*Example:* To the Buses example, if we expected four mutually exclusive and exhaustive events.

E1 – Buses Switched.
E2 – Buses ten minutes or more lately.
E3 – Buses less than ten minutes late.
E4 – Buses on time or early.

The probability of an event is usually indicated by a capital letter P, so we might have;

P (E1) = 0.04
P (E2) = 0.17
P (E3) = 0.21
P (E4) = 0.58

Each of these probabilities is between 0 and 1 inclusive, as it has to be qualify as a probability. They also satisfy a second important condition: the sum of the four probabilities has to be 1, because precisely one of the events must always occur. In this case;

$$P(E1) + P(E2) + P(E3) + P(E4) = 1 \qquad (1)$$

Generally we are not in a position to know the true probability of an event occurring. To do so for the buses example we would have to record the buses’ arrival time for all possible days on which it is scheduled to run, then count the number of times events E1,

E2, E3 and E4 occur and divide by the total number of days, to give the probabilities of the four events. In practice this is often prohibitively difficult or impossible to do, especially if the trials may potentially go on forever. Instead we keep records for a sample of say 365 days, count the number of times E1, E2, E3 and E4 occur, divide by 365 (the number of days) to give the frequency of the four events and use these as estimates of the four probabilities.
The outcome of each trial is recorded in one row of a table. Each row must have one and only one classification. For classification tasks, the usual terminology is to call a table (dataset) such as Table 1, a training set. Each row of the training set is called an instance. An instance comprises the values of a number of attributes and the corresponding classification. The training set constitutes the results of a sample of trials that we can use to predict the classification of other (unclassified) instances. Suppose that our training set consists of 24 instances, each recording the value of four attributes as well as the classification. We will use classifications: switched, very late, late and on time to correspond to the events E1, E2, E3 and E4 described previously.

**Table 2.** The buses dataset

| No. | Day | Season | Wind | Rain | Class |
|---|---|---|---|---|---|
| 1 | Weekday | Spring | None | Low | On Time |
| 2 | Weekday | Autumn | None | Medium | On Time |
| 3 | Weekday | Winter | None | High | Late |
| 4 | Thursday | Summer | None | Low | On Time |
| 5 | Weekday | Winter | High | High | Very Late |
| 6 | Friday | Summer | High | Low | Late |
| 7 | Holiday | Winter | Normal | High | On Time |
| 8 | Weekday | Autumn | High | Low | On Time |
| 9 | Weekday | Winter | None | High | Late |
| 10 | Thursday | Spring | Normal | Low | On Time |
| 11 | Holiday | Autumn | Normal | Low | Switched |
| 12 | Weekday | Winter | High | High | Very Late |
| 13 | Friday | Spring | Normal | Low | On Time |
| 14 | Weekday | Winter | None | High | Late |
| 15 | Weekday | Autumn | High | High | Very Late |
| 16 | Weekday | Spring | None | Low | On Time |
| 17 | Friday | Summer | None | Low | On Time |
| 18 | Weekday | Spring | None | Medium | On Time |
| 19 | Thursday | Summer | None | Low | On Time |
| 20 | Weekday | Summer | Normal | Low | On Time |
| 21 | Weekday | Autumn | None | Low | On Time |
| 22 | Thursday | Spring | Normal | High | Very Late |
| 23 | Weekday | Summer | None | Low | On Time |
| 24 | Weekday | Autumn | High | Low | Late |

For the buses data we can tabulate all the conditional and prior probabilities as shown in Table 2.

**Table 2.** Conditional and prior probabilities

| | class= On Time | class= Late | class= Very Late | class= Switched |
|---|---|---|---|---|
| day=Weekday | 8/14=0.57 | 4/5=0.8 | 3/4=0.75 | 0/1=0 |
| day=Friday | 2/14=0.14 | 1/5=0.2 | 0/4=0 | 0/1=0 |
| day=Holiday | 1/14=0.07 | 0/5=0 | 0/4=0 | 1/1=1 |
| day=Thursday | 3/14=0.21 | 0/5=0 | 1/4=0.25 | 0/1=0 |
| season=Spring | 5/14=0.36 | 1/5=0.2 | 1/4=0.25 | 0/1=0 |
| season=Winter | 1/14=0.07 | 2/5=0.4 | 2/4=0.5 | 0/1=0 |
| season=Summer | 5/14=0.36 | 0/5=0 | 0/4=0 | 0/1=0 |
| season=Autumn | 3/14=0.21 | 1/5=0.2 | 1/4=0.25 | 1/1=0 |
| wind=None | 9/14=0.64 | 0/5=0 | 0/4=0 | 0/1=0 |
| wind=High | 1/14=0.07 | 3/5=0.6 | 3/4=0.75 | 0/1=0 |
| wind=Normal | 4/14=0.28 | 1/5=0.2 | 1/4=0.25 | 1/1=0 |
| Rain=None | 11/14=0.78 | 1/5=0.2 | 0/4=0 | 1/1=0 |
| Rain =Medium | 2/14=0.14 | 0/5=0 | 0/4=0 | 0/1=0 |
| Rain =High | 1/14=0.07 | 3/5=0.6 | 4/4=1 | 0/1=0 |
| Prior Probability | 14/24=0.58 | 5/24=0.21 | 4/24=0.17 | 1/24=0.04 |

For example, the conditional probability P (day = weekday | class = on time) is the number of instances in the buses dataset for which day = weekday and class = on time, divided by the total number of instances for which class = on time. These numbers can be counted from Table 1 as 8 and 14, respectively. So the conditional probability is 8/14 = 0.57.
The prior probability of class = very late is the number of instances in Table 1 for which class = very late divided by the total number of instances, i.e. 4/24 = 0.17.
We can now use these values to calculate the probabilities of real interest to us. These are the posterior probabilities of each possible class

occurring for a specified instance, i.e. for known values of all the attributes. We can calculate these posterior probabilities by using Na¨ıve Bayes Classification.
When using the Na¨ıve Bayes method to classify a series of unseen instances the most efficient way to start is by calculating all the prior probabilities and also all the conditional probabilities involving one attribute, though not all of them may be required for classifying any particular instance.
Using the values in each of the columns of Table 2 in turn, we obtain the following probabilities for each possible classification for the unseen instance:

| | | | | |
|---|---|---|---|---|
| Weekday | Winter | High | High | ***Very Late*** |

Class = on time
0.58 * 0.57 * 0.07 * 0.07 * 0.07=0.0001
Class = late
0.21 * 0.8 * 0.4 * 0.6 * 0.6=0.0240
Class = very late
0.17 * 0.75 * 0.5 * 0.75 * 0.75=0.0360
Class = Switched
0.04 * 0 * 0 * 0 * 0 =0.0000

The largest value is for class = *Very Late.*

Since we are looking for the probability, we selected the highest value which is less than one to be the value for the unseen. Note that the four values calculated are not themselves probabilities, as they do not sum to 1. This is the significance of the phrasing 'the posterior probability. Each value can be 'normalized' to a valid posterior probability simply by dividing it by the sum of all four values. In practice, we are interested only in finding the largest value so the normalization step is not necessary.

The Na¨ıve Bayes approach is a very popular one, which often works well. However it has a number of potential problems, the most obvious one being that it relies on all attributes being categorical. In practice, many datasets have a combination of categorical and continuous attributes, or even only continuous attributes. This problem can be overcome by converting the continuous attributes to categorical ones.
A second problem is that estimating probabilities by relative frequencies can gave a poor estimate if the number of instances with a given attribute/value combination is small. In the extreme case where it is zero, the posterior probability will inevitably be calculated as zero. This happened for class = *switched*.

## 7.2 Nearest Neighbor Classification

Nearest Neighbor classification is mainly used when all attribute values are continuous, although it can be modified to deal with categorical attributes. The idea is to estimate the classification of an unseen instance using the classification of the instance or instances that are closest to it, in some sense that we need to define.
Supposing we have training set with just two instances such as that shown in Table 3. There are six attribute values, followed by a classification (positive or negative). We are then given a third instance

**Table 3**. Training set for two instances

| a | b | c | d | e | f | Class |
|---|---|---|---|---|---|---|
| yes | no | no | 11 | 100 | low | negative |
| yes | yes | yes | 25 | 200 | high | positive |

| | | | | | | |
|---|---|---|---|---|---|---|
| **yes** | **no** | **no** | **22** | **180** | **High** | **???** |

What should its classification be?

Even without knowing what the six attributes represent, it seems intuitively obvious that the unseen instance is nearer to the second instance than to the first.

In practice there are likely to be many more instances in the training set but the same principle applies. It is usual to base the classification on those of the k nearest neighbors (where k is a small integer such as (3 or 5), not just the nearest one. The method is then known as k-Nearest Neighbor or just k-NN classification as follows;

*Basic k-Nearest Neighbour Classification Algorithm*

- Find the k training instances that are closest to the unseen instance.
- Take the most commonly occurring classification for these k instances.

We can illustrate k-NN classification diagrammatically when the dimension (i.e. the number of attributes) is small. The following example illustrates the case where the dimension is just 2. In real-world data mining applications it can of course be considerably larger.

Table 4 shows a training set with 25 instances, each giving the values of two attributes for Material Order and Material Value and an associated classification.

How can we estimate the classification for an 'unseen' instance where the first and second attributes are 28 and 300, respectively?

For this small number of attributes we can represent the training set as 25 points on a two-dimensional graph with values of the first and second attributes measured along the horizontal and vertical axes, respectively. Each point is labeled with a + or − symbol to indicate that the classification is positive or negative, respectively.

The third attribute is used to find the unknown class (negative or positive) based on the nearest neighbor's classification and the higher number of occurrences. Based on our earlier assumption the class turned to be positive as illustrated in Figure 4.

**Table 4.** Training set for material data

| No. | Attribute (1) for Mat. Order | Attribute (2) for Mat. Value | Class |
|---|---|---|---|
| **1** | 1 | 773 | − |
| **2** | 2 | 200 | + |
| **3** | 3 | 100 | − |
| **4** | 4 | 200 | − |
| **5** | 6 | 100 | − |
| **6** | 7 | 200 | − |
| **7** | 8 | 100 | − |
| **8** | 9 | 100 | − |
| **9** | 10 | 110 | − |
| **10** | 11 | 773 | + |
| **11** | 12 | 120 | − |
| **12** | 13 | 125 | − |
| **13** | 14 | 115 | − |
| **14** | 15 | 130 | − |
| **15** | 16 | 100 | − |
| **16** | 17 | 95 | − |
| **17** | 21 | 80 | + |
| **18** | 22 | 80 | + |
| **19** | 23 | 80 | + |
| **20** | 24 | 80 | + |
| **21** | 25 | 80 | + |
| **22** | 26 | 240 | + |
| **23** | 27 | 240 | + |
| **24** | 28 | 240 | + |
| **25** | 29 | 800 | + |

A circle has been added to enclose the three nearest neighbors of the unseen instance, which is shown as a small circle close to the centre of the larger one.

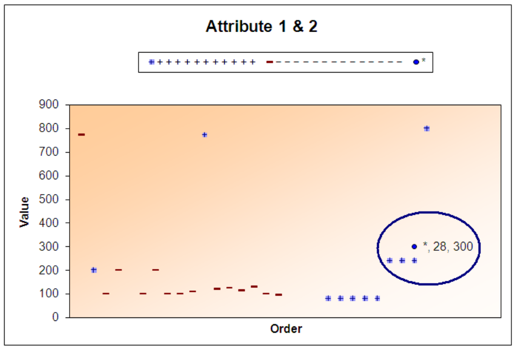


**Figure 4.** Two-dimensional representation of training data in Table 4

## 7.3 Decision Trees for Classification

A decision tree is a predictive model that, as its name implies, can be viewed as a tree. Specifically each branch of the tree is a classification question and the leaves of the tree are partitions of the dataset with their classification.

A decision tree is a classifier expressed as a recursive partition of the instance space. A decision tree consists of nodes that form a Rooted Tree, meaning it is a Directed Tree with a node called root that has no incoming edges. All other nodes have exactly one incoming edge. A node with outgoing edges is called internal node or test nodes. All other nodes are called leaves (also known as terminal nodes or decision nodes).

In a decision tree, each internal node splits the instance space into two or more subspaces according to a certain discrete function of the input attributes values. In the simplest and most frequent case each test considers a single attribute, such that the instance space is partitioned according to the attribute's value. In the case of numeric attributes the condition refers to a range.

Each leaf is assigned to one class representing the most appropriate target value. Usually the most appropriate target value is the class with the greatest representation, because selecting this value minimizes the zero-one loss. However if a different loss function is used then a different class may be selected in order to minimize the loss function. Alternatively the leaf may hold a probability vector indicating the probability of the target value having a certain value.

A decision tree is created by a process known as splitting on the value of attributes (or just splitting on attributes), i.e. testing the value of an attribute such as CTRYNAME and then creating a branch for each of its possible values. In the case of continuous attributes the test is normally whether the value is ‘less than or equal to’ or ‘greater than’ a given value known as the split value. The splitting process continues until each branch can be labeled with just one classification. Decision trees have two different functions: data compression and prediction.

Table 5 Gives a training set of data collected about 20 employees, tabulating four items of data about each one (title, marital status, sex and position) against the department joined.

**Table 4.** Training set for the IT Dept/TM Dept example

| No | Title | Married | National | Position | Class |
|---|---|---|---|---|---|
| **1** | Employee | Yes | No | Engineer | IT Dept. |
| **2** | Contractor | Yes | Yes | Engineer | IT Dept. |
| **3** | Employee | No | Yes | Engineer | TM Dept. |
| **4** | Employee | No | Yes | Engineer | IT Dept. |
| **5** | Employee | No | Yes | Technician | IT Dept. |
| **6** | Contractor | Yes | Yes | Technician | IT Dept. |
| **7** | Employee | Yes | No | Engineer | IT Dept. |
| **8** | Employee | Yes | Yes | Technician | TM Dept. |
| **9** | Contractor | No | Yes | Technician | IT Dept. |
| **10** | Contractor | No | Yes | Technician | IT Dept. |
| **11** | Contractor | No | Yes | Engineer | IT Dept. |
| **12** | Employee | No | Yes | Technician | TM Dept. |
| **13** | Employee | Yes | Yes | Technician | TM Dept. |
| **14** | Employee | Yes | No | Engineer | IT Dept. |
| **15** | Employee | Yes | No | Engineer | IT Dept. |
| **16** | Employee | No | Yes | Technician | TM Dept. |
| **17** | Contractor | No | Yes | Technician | IT Dept. |
| **18** | Contractor | No | Yes | Technician | IT Dept. |
| **19** | Employee | No | Yes | Engineer | TM Dept. |
| **20** | Employee | Yes | Yes | Technician | TM Dept. |

What determines who joins which department? It is possible to generate many different trees from this data using the Top Down Induction of Decision Trees (TDIDT) algorithm [11]. One possible decision tree is Figure 5.

This is a remarkable result. All the Contractor employees work in IT Department. For the direct employees, the critical factor is position status. If they are engineer ones all work in IT Department. If they are Technician, they are working in TM Department. All non-national are direct employees and working in IT Department.

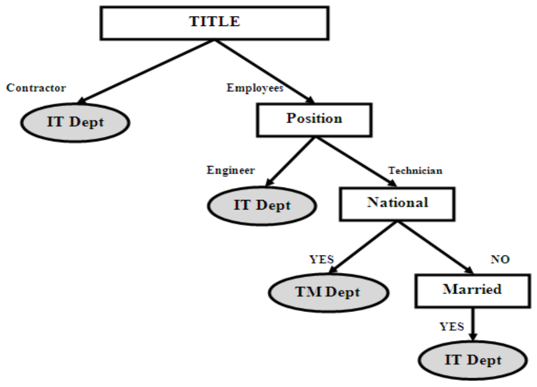


**Figure 5.** IT Dept and TM Dept example: decision tree.

## 8 CONCLUSIONS

The method of compression that introduced in this paper can be considered an important method in database processing such as data mining when the time of data access and memory storage optimization is very important, a proposed examples to show the compression ratio was depends on the types of database repetition that occur in database and the size of data, as the size of data increased the compression ratio of data will be increased. The compression ratio is very important in reducing the storage needed, and it is considered a very important step in data mining preparation before processing. By data compression, data can occupy a little memory as possible, and permit processor optimization to be effective at run time. In this paper the compression ratio with one attribute (primary key) can have more compression ratio compared with the similar data, which have two unrepeated attributes, so we can conclude that this method of compression will be suitable with database have repetition in their attributes.

Classification is one of the most common data mining tasks. The three techniques; Na¨ıve, nearest neighbor and decision tree are implemented in this paper with different resources of data. The Bayes Na¨ıve Bayes algorithm was calculated the most likely classification for weather forecasting of buses dataset and the probability for unknown classification. This algorithm can be effective in deductive learning algorithm for machine learning similar to the example introduced in this paper to predicate the bus arrival depends on classification historical data.

Nearest Neighbor technique which is a prediction technique that is quite similar to clustering and it calculates the instance to determine an unknown classification of attributes, and get highest number of occurrences. This algorithm is most suitable when all the attribute values are continuous or categorized attributes; the predicated value is the value which takes its significant value from its neighbor.

Finally, the decision tree technique which has two different functions: data compression and prediction. It is popular for classification and predication, it classify data by rules that can be expressed in English, so that can be understand them easily. The decision tree can gave high accuracy in classify the data at hand and establishing a pattern that is helpful in future predictions.

The three different methods have been used with different sets of data; each one has its own data set. The results show that the three methods are appropriate for determining the target value, but selecting the suitable method is very important to obtain correct results and this selection depends mainly on the nature of data types and their completeness by knowing exactly what the data warehouse have.

## 9 REFERENCES

1. Bramer, M. A.: Principles of Data Mining. London: Springer (2007).
2. Tuban, Leidner, Mclean, Welherbe: Information Technology for Management, John Wiley and Sons, Inc, 7th ed (2010)
3. Osmar R. Zaïane: Principles of Knowledge Discovery in Databases (1999).
4. Ramez Elmasri and Shamkant B. Navathe: Fundamentals of Database Systems, 5th edition, Addison-Wesley Pub Co. (2007).
5. Clare Churcher: Beginning Database Design from Novice to Professional, Apress (2007).
6. Soransen K., Janssens G.: Data Mining with Genetic Algorithms on Binary Trees, European Journal of Operational Research, vol. 151, pp. 253--264 (2003).
7. Ghosh A., Nath B.: Multi-Objective Rule Mining using Genetic Algorithms, Information Science: An International Journal, vol. 163 no. 1-3, pp. 123--33 (2004).
8. Frawley W., and Piatetsky-Shapiro and Matheus C.: Knowledge Discovery in Databases: An Overview. AI Magazine: pp. 213--228. ISSN 0738-4602 (1992).
9. Hand D., Mannila H., Smyth P.: Principles of Data Mining. MIT Press, Cambridge, MA. (2001).
10. 10. Olson, D., Delen, D.: Advanced Data Mining Techniques. Springer-Verlag Berlin Heidelberg (2008).
11. Sharma, S., Osei-Bryson, K.-M.: Framework for Formal Implementation of the Business Understanding Phase of Data Mining Projects. Expert Systems with Applications, 36 (2), pp. 4114--4124 (2009).
12. Bhargavi, P., Jyothi, S.: Applying Naive Bayes Data Mining Technique for Classification of Agricultural Land Soils. International Journal of Computer Science and Network Security, 9 (8), pp. 117--122 (2009).
13. Seyed Mousavi R., Krysia Broda: Impact of Binary Coding on Multiway-split TDIDT Algorithms, International Journal of Electrical, Computer, and Systems Engineering 2;3, http://www.waset.org. (2008).
14. Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, Dan Steinberg: Top 10 Algorithms in Data Mining, Knowledge Information Systems, Springer, 14:1–37, (2008).