Chapter 2
Data Mining Techniques and Tools
2.1 Introduction

The need to scale up human analysis capabilities in handling the large number of bytes that one can collect is both economic and scientific manner. Businesses use data to gain competitive advantage, increase efficiency, and provide more valuable services to customers. Data captured about the environment are the basic evidence used to build theories and models of the universe. Because computers have enabled humans to gather more data than can be digested, it is only natural to turn to computational techniques to help us unearth meaningful patterns and structures from the massive volumes of data. Hence, KDD is an attempt to address a problem that the digital information era made a fact of life for all. As large data sets encompass hidden trends, which convey valuable knowledge about the data set. The derived or acquired knowledge is very helpful in predicting the behavior of the user based on the data description. It can be expressed as rules or correlations highlight the associations that exist in the data [14].

Data mining research has drawn on a number of other fields such as statistics and machine learning. In chapter 1, data mining, applications of data mining, relationship with KDD and also, review the relations of data mining with some of the important areas has been discussed. This chapter explores the data mining techniques and the data mining tools

2.2 Statistics

Statistical techniques are driven by the data and are used to discover patterns and build predictive models. And from the users perspective one will be faced with a conscious choice when solving a "data mining" problem as to whether one wish to attack it with statistical methods or other data mining techniques. For this reason it is important to have some idea of how statistical techniques work and how they can be applied.

Knowing statistics in everyday life will help the average business person make better decisions by allowing them to figure out risk and uncertainty when all
the facts either aren’t known or can’t be collected. Even with all the data stored in the largest of data warehouses business decisions still just become more informed guesses. Statistics has been around for a long time easily a century and arguably many centuries when the ideas of probability began to gel. Today data mining has been defined independently of statistics though “mining data” for patterns and predictions is really what statistics is all about. Statistics can help greatly in this process by helping to answer several important questions about your data:

What patterns are there in database?

What is the chance that an event will occur?

Which patterns are significant?

What is a high level summary of the data that gives one some idea of what is contained in database?

Certainly statistics can do more than answer these questions but for most people today these are the questions that statistics can help answer.

Statistics is a science assisting one to make decisions under uncertainties (based on some numerical and measurable scales). Decision making process must be based on data neither on personal opinion nor on belief.

Statistics is a set of methods that are used to collect, analyze, present, and interpret data. Statistical methods are used in a wide variety of occupations and help people identify, study, and solve many complex problems. In the business and economic world, these methods enable decision makers and managers to make informed and better decisions about uncertain situations.

Vast amounts of statistical information are available in today's global and economic environment because of continual improvements in computer technology. To compete successfully globally, managers and decision makers must be able to understand the information and use it effectively. Statistical data analysis provides hands on experience to promote the use of statistical
thinking and techniques to apply in order to make educated decisions in the business world.

*Regression analysis* is a statistical tool for the investigation of relationships between variables. Usually, the investigator seeks to ascertain the causal effect of one variable upon another—the effect of a price increase upon demand, for example, or the effect of changes in the money supply upon the inflation rate. To explore such issues, the investigator assembles data on the underlying variables of interest and employs regression to estimate the quantitative effect of the causal variables upon the variable that they influence. The investigator also typically assesses the “statistical significance” of the estimated relationships, that is, the degree of confidence that the true relationship is close to the estimated relationship.

*Statistics for Prediction (Linear Regression)*

In statistics, prediction is usually synonymous with regression of some form. There are a variety of different types of regression in statistics but the basic idea is that a model is created that maps values from predictors in such a way that the lowest error occurs in making a prediction. The simplest form of regression is simple linear regression that just contains one predictor and a prediction. The relationship between the two can be mapped on a two dimensional space and the records plotted for the prediction values along the Y axis and the predictor values along the X axis. The simple linear regression model then could be viewed as the line that minimized the error rate between the actual prediction value and the point on the line (the prediction from the model). Graphically this would look as it does in Figure 2.1. The simplest form of regression seeks to build a predictive model that is a line that maps between each predictor value to a prediction value. Of the many possible lines that could be drawn through the data the one that minimizes the distance between the line and the data points is the one that is chosen for the predictive model.
On an average to guess the value on the line it should represent an acceptable compromise amongst all the data at that point giving conflicting answers. Likewise if there is no data available for a particular input value the line will provide the best guess at a reasonable answer based on similar data.

![Figure 2.1 Linear Regression](image)

The predictive model is the line shown in Figure 2.1. The line will take a given value for a predictor and map it into a given value for a prediction. The actual equation would look something like: Prediction = a + b * Predictor. It is just the equation 2.1 for a line. The trick, as always with predictive modeling, is to find the model that best minimizes the error. The most common way to calculate the error is the square of the difference between the predicted value and the actual value. Calculated this way points that are very far from the line will have a great effect on moving the choice of line towards them in order to reduce the error. The values of a and b, that minimize this error can be calculated directly from the data relatively quickly.

\[ Y = a + bX \]  

Adding more predictors to the linear equation 2.1, it can produce more complicated lines that take more information into account and hence make a better prediction. This is called *multiple linear regression* and might have an equation 2.2 (if 5 predictors were used X1, X2, X3, X4, X5):

\[ Y = a + b1(X1) + b2(X2) + b3(X3) + b4(X4) + b5(X5) \]
This equation still describes a line but it is now a line in a 6 dimensional space rather than the two dimensional space.

By transforming the predictors by squaring, cubing or taking their square root it is possible to use the same general regression methodology and now create much more complex models that are no longer simple shaped like lines. This is called non-linear regression. A model of just one predictor might look like this: \( Y = a + b1(X1) + b2(X2) \). In many real world cases analysts will perform a wide variety of transformations on their data just to try them out. If they do not contribute to a useful model their coefficients in the equation will tend toward zero and then they can be removed. The other transformation of predictor values that is often performed is multiplying them together. For instance a new predictor created by dividing hourly wage by the minimum wage might be a much more effective predictor than hourly wage by itself.

When trying to predict a customer response that is just yes or no (e.g. they bought the product or they didn’t or they defaulted or they didn’t) the standard form of a line doesn’t work. Since there are only two possible values to be predicted it is relatively easy to fit a line through them. However, that model would be the same no matter what predictors were being used or what particular data was being used. Typically in these situations a transformation of the prediction values is made in order to provide a better predictive model. This type of regression is called logistic regression and because so many business problems are response problems, logistic regression is one of the most widely used statistical techniques for creating predictive models.

Statistical analysis is related with historical facts based on observed data. Secondly, statistical analysis focuses on finding and explaining the major sources of variations in the data, in contrast data mining endeavors to discover, not the obvious sources of variations, but rather the meaningful, although currently overlooked, information. Therefore, statistical analysis and data mining are complementary. Statistical analysis explains and removes the major part of data variations before data mining is used.
2.3 **Machine Learning**

Machine learning is the automation of a learning process and learning is tantamount to the construction of rules based on observations. This is a broad field which includes not only learning from examples, but also reinforcement learning, learning with a teacher etc. A learning algorithm takes the data set and its accompanying information as the input and returns a statement, e.g., a concept representing a result of learning as output. Inductive learning, where the system infers the knowledge itself from observing its environment, has two main strategies: Supervised Learning and Unsupervised Learning. The model produced by inductive learning methods can be used to predict the outcome of future situations; in other words, not only for state encountered but rather for unseen states that could occur.

2.3.1 **Supervised Learning**

Supervised Learning means learning from examples, where a training set is given which acts as examples for classes. The system finds a description of each class. Once the description has been formulated, it is used to predict the class of previously unseen classes. This is similar to discriminate analysis which occurs in statistics.

2.3.2 **Unsupervised Learning**

Unsupervised Learning is learning from observation and discovery. In this mode of learning, there is no training set or prior knowledge of classes. The system analyzes the given set of data to observe similarities emerging out of subsets of the data. The outcome is a set of class descriptions, one for each class, discovered in the environment. This is similar to cluster analysis in statistics.

Data mining and the part of machine learning deals with learning from examples. Data mining is concerned with finding understandable knowledge, while machine learning is concerned with improving the performance of
intelligent system or agent for problem-solving tasks. Integrating machine learning techniques into database systems, some of the databases require more efficient learning algorithms because realistic databases are normally very large and noisy [127].

2.3.3 Mathematical Programming

The relationship between Mathematical Programming and data mining was not so obvious until the pioneering work by O L Mangasarian. Most of the data mining task can be equivalently formulated as mathematical programming for which efficient algorithms are available. It provides a new insight into the problems of data mining. Support Vector Machines approach for classification.

2.4 Data Mining Techniques

The ultimate goal of data mining is prediction and description. Prediction makes use of existing variables in the database in order to predict unknown or future value of interest, and description focuses on finding patterns describing the data and the subsequent presentation description differ with respect to the underlying applications and the technique. There are several data mining techniques fulfilling these objectives. Some of these are associations, classifications, sequential patterns and clustering

Another approach of the study of DM techniques is to classify the techniques as:

- User-guided or verification-driven data mining and
- Discovery- driven or automatic discovery of rules.

2.4.1 Verification Model

In the process of data mining, the user makes a hypothesis and tests the hypothesis on the data to verify its validity. The emphasis is on the user who is
responsible for formulating the hypothesis and issuing the query on the data to affirm or negate the hypothesis.

2.4.2 Discovery Model

The discovery model differs in its emphasis, in that it is the system automatically discovering important information hidden in the data. The data is sifted in search of frequently occurring patterns, trends and generalizations about the data without intervention or guidance from the user [62].

The manner in which the rules are discovered depends on the class of the data mining application. The typical discovery driven tasks are

- Discovery of Association rules
- Discovery of Classification rules
- Clustering
- Discovery of Frequent episodes
- Deviation detection

These tasks are of exploratory in nature and cannot be handed over to currently available database technology. In the following sections, these tasks are discussed.

Discovery of Association rules

Association discovers relationship or dependency between multiple things, such as link analysis, market basket analysis, and variables dependency. Association exists at two levels: structured and quantitative. The structural level of method specifies (often in graphical form) which things are related; the quantitative level specifies the strengths of the relationship using some numerical scale. Market Basket analysis is a well known association application; it can be performed on the retail data of customer transactions to find out what items are frequently purchased together (a.k.a. itemsets). Apriori
is the basic algorithm for finding frequent itemsets. The extension of Apriori further handles multilevel, multidimensional, and more complex data structure.

Association rule is a popular and well researched method for discovering interesting relations between variables in large databases. Association rule mining finds interesting associations and/or correlation relationships among large set of data items. Association rules shows attributed value conditions that occur frequently together in a given dataset. Mining association rules on large data sets has received considerable attention in recent years. Association rules are useful for determining correlations between attributes of a relation and have applications in marketing, financial, and retail sectors. Furthermore, optimized association rules are an effective way to focus on the most interesting characteristics involving certain attributes. Optimized association rules are permitted to contain uninstantiated attributes and the problem is to determine instantiations such that either the support or confidence of the rule is maximized. For example, data are collected using bar-code scanners in supermarkets. Such ‘market basket’ data bases consist of a large number of transaction records. Each record lists all items bought by a customer on a single purchase transaction. Managers could use this data for adjusting store layouts, cross-selling, promotions, and catalog design and to identify customer segments based on buying patterns

**Clustering**

Clustering is a tool for data analysis, which solves classification problems. Its objective is to distribute cases (people, objects, events etc.) into groups, so that the degree of association can be strong between members of the same cluster and weak between members of different clusters.

In clustering, there is no preclassified data and no distinction between independent and dependent variables. Instead, clustering algorithms search for groups of records (the clusters composed of records similar to each other). The
algorithms discover these similarities. This way each cluster describes, in terms of data collected, the class to which its members belong. Clustering is a discovery tool. It may reveal associations and structure in data which, though not previously evident, nevertheless are sensible and useful once found. The results of cluster analysis may contribute to the definition of a formal classification scheme, such as taxonomy for related animals, insects or plants; suggest statistical models with which to describe populations; indicate rules for assigning new cases to classes for identification and diagnostic purposes; provide measures of definition, size and change in what previously were only broad concepts; or find exemplars to represent classes. Whatever business one is in, the chances are that sooner or later they will run into a classification problem. Cluster analysis might provide the methodology to help them to solve it. In clustering, there are no predefined classes and no examples. The records are grouped together on the basis of self-similarity. It is up to the user to determine what meaning, if any, to attach to the resulting clusters. Clustering is often done as a prelude to some other form of data mining [132].

*Discovery of Classification Rules*

Classification maps (or classifies) a data item into one of several predefined categorical classes. Decision tree neural network and some probability approaches are often used to perform this function. There are two steps to implement classification function. In the first step, classification model is build describing a predetermined set of classes or concepts. Second step, the model is used for classification. For example the classification rules learned in the first step from the analysis of data from existing customers can be used to predict the credit rating of new or future customers.

*Frequent Episodes*

Frequent Episodes are the sequence of events that occur frequently, close to each other and are extracted from the time sequence. How close it has to be to
consider it as frequent is domain dependent. This is given by the user as the input and the output are the prediction rules for the time sequences.

These episodes can be of three types: The serial episode which occurs in sequence. The parallel episode in which there are no constraint on the order of the event types A and B given and the non-serial or non-parallel episodes which occur in a sequence if the occurrence of A and B precede an occurrence of C, and there is no constraint on the relative order of A and B given.

*Deviation Detection*

Deviation detection is to identify outlying points in a particular data set, and explain whether they are due to noise or other impurities being present in the data or due to trivial reasons. It is usually applied with the database segmentation, and is the source of true discovery, since the outliers express deviations from some previously known expectation or norm. By calculating the values of measures and current data and comparing them with previous data as well as with the normative data, the deviations can be obtained.

Most of the techniques of data mining have elements of both the models. The following paragraphs discusses about relatively new computing paradigm

*Neural Network*

Neural networks have been successfully applied in a wide range of supervised and unsupervised learning applications. Neural network methods are commonly used for data mining tasks, because they often produce comprehensible models. A neural network is a computational technique that benefits from techniques similar to ones employed in the human brain. It is designed to mimic the ability of the human brain to process data and information and comprehend patterns. It imitates the structure and operations of the three dimensional lattice of network among brain cells (nodes or neurons, and hence the term “neural”). Technology is inspired by the architecture of the human brain, which uses many simple processing elements
operating in parallel to obtain high computation rates. Similarly, the neural network is composed of many simple processing elements or neurons operating in parallel whose functions are determined by network structure, connection strengths, and the processing performed at computing elements or nodes. The network’s strength is in its ability to comprehend and discern subtle patterns in a large number of variables at a time without being stifled by detail. It can also carry out multiple operations simultaneously. Not only can it identify patterns in a few variables, it also can detect correlations in hundreds of variables. It is this feature of the network that is particularly suitable in analyzing relationships among a large number of market variables. The networks can learn from experience. They can cope with “fuzzy” patterns — patterns that are difficult to reduce into precise rules. They can also be retrained and thus can adapt to changing market behavior. Even when a data set is noisy or has irrelevant inputs, the networks can learn important features of the data. Inputs that may appear irrelevant may in fact contain useful information. The promise of neural networks lies in their ability to learn patterns in a complex signal [126].

**Genetic Algorithms**

Genetic algorithms are algorithmic optimization strategies that are inspired by the principles observed in natural evolution. Of a collection of potential problem solutions that compete with each other, the best solutions are selected and combined with each other. In doing so, one expects that the overall goodness of the solution set will become better and better, similar to the process of evolution of a population of organisms. Genetic algorithms are used in data mining to formulate hypotheses about dependencies between variables, in the form of association rules or some other internal formalism.

**Rough Sets Approach**

A rough set is determined by a lower and upper bound of a set. Every member of the lower bound is a certain member of the set. Every non-member of the
upper bound is a certain non-member of the set. The upper bound of a rough set is the union between the lower bound and the so-called boundary region. A member of the boundary region is possibly (but not certainly) a member of the set. Therefore, rough sets may be viewed as with a three-valued membership function (yes, no, perhaps). Rough sets are a mathematical concept dealing with uncertainty in data. They are usually combined with other methods such as rule induction, classification, or clustering methods.

**Support Vector Machine (SVM)**

Support vector machine (SVM) is an algorithm for the classification of both linear and nonlinear data. It transforms the original data in a higher dimension, from where it can find a linear separator (hyper-plane) for separation of the data using essential training tuples called support vectors.

The following section gives a glance of Waikato Environment for Knowledge Analysis (WEKA 3-6-2), a data mining tool with modification in the built in functions. In the research, this tool has been used to test and implement the concepts.

### 2.5 WEKA

It is a data mining system developed by the University of Waikato in New Zealand that implements data mining algorithms. WEKA is a state-of-the-art facility for developing ML techniques and their application to real-world data mining problems. It is a collection of machine learning algorithms for data mining tasks. The algorithms are applied directly to a dataset. WEKA implements algorithms for data preprocessing, classification, regression, clustering, association rules; it also includes a visualization tools. The new machine learning schemes can also be developed with this package. WEKA is open source software issued under the GNU General Public License. The data file normally used by WEKA is in ARFF file format, which consists of special
tags to indicate different things in the data file. The following figure 2.2 shows the WEKA GUI interface

![Figure 2.2 WEKA GUI Interface](image)

Following are the list of panels available on WEKA GUI Interface

### 2.6 Explorer

The main interface in WEKA is the Explorer. It is an environment for exploring data with WEKA. At the very top of the window, just below the title bar, is a row of tabs. When the Explorer is first started only the first tab is active; the others are grayed out (Figure 2.3).

![Figure 2.3 WEKA Explorer Interface](image)
The tabs are as follows:

2.6.1 Preprocess

Choose and modify the data being acted on.

2.6.1.1 Loading Data

The first four buttons at the top of the preprocess section enables to load data into WEKA:

*Open file:* Brings up a dialog box to browse for the data file on the local file system. Using the Open file, button can be read files in a variety of formats: WEKA’s ARFF format, CSV format, C4.5 format, or serialized Instances format. ARFF files typically have an .arff extension, CSV files a .csv extension, C4.5 files a .data and .names extension, and serialized Instances objects a .bsi extension.

![Figure 2.4 WEKA Explorer after loading data](image)
2.6.1.2 The Current Relation

Once some data has been loaded (Figure 2.4), the Preprocess panel shows a variety of information. The Current relation box (the “current relation” is the currently loaded data, which can be interpreted as a single relational table in database terminology) has three entries:

i. Relation: The name of the relation, as given in the file it was loaded from.

ii. Instances: The number of instances (data points/records) in the data.

iii. Attributes: The number of attributes (features) in the data.

2.6.1.3 Working with Attributes

Below the Current relation box is a box titled Attributes. There are four buttons, and beneath them is a list of the attributes in the current relation. The list has three columns:

i. No: A number that identifies the attribute in the order they are specified in the data file.

ii. Selection tick boxes: These allow selecting which attributes are present in the relation.

iii. Name: The name of the attribute, as it was declared in the data file. When different rows in the list of attributes are clicked, the fields change in the box to the right titled.

iv. Selected attribute: This box displays the characteristics of the currently highlighted attribute in the list:

   a. Name: The name of the attribute, the same as that given in the attribute list.

   b. Type: The type of attribute, most commonly Nominal or Numeric.

   c. Missing: The number (and percentage) of instances in the data for which this attribute is missing (unspecified).
d. Distinct: The number of different values that the data contains for this attribute.

e. Unique: The number (and percentage) of instances in the data having a value for this attribute that no other instances have.

Below these statistics is a list showing more information about the values stored in this attribute, which differ depending on its type. If the attribute is nominal, the list consists of each possible value for the attribute along with the number of instances that have that value. If the attribute is numeric, the list gives four statistics describing the distribution of values in the data—the minimum, maximum, mean and standard deviation. And below these statistics there is a colored histogram, color-coded according to the attribute chosen as the Class using the box above the histogram. Finally, after pressing the Visualize All button, histograms for all the attributes in the data are shown in a separate window. Returning to the attribute list, to begin with all the tick boxes are unticked. They can be toggled on/off by clicking on them individually. The four buttons above can also be used to change the selection:

2.6.1.4 Preprocessing

i. All: All boxes are ticked.

ii. None: All boxes are cleared.

iii. Invert: Boxes that are ticked become unticked and vice versa.

iv. Pattern: Enables the user to select attributes based on a Perl 5 Regular Expression. E.g., *id selects all attributes which name ends with id. Once the desired attributes have been selected, they can be removed by clicking the Remove button below the list of attributes. They can be undone by clicking the Undo button, which is located in the top-right corner of the Preprocess panel.
2.6.1.5 Working with Filters

The preprocess section allows filters to be defined that transform the data in various ways. The Filter box is used to set up the filters that are required. At the left of the Filter box is a Choose button. By clicking this button it is possible to select one of the filters in WEKA. Once a filter has been selected, its name and options are shown in the field next to the Choose button. Clicking on this box with the left mouse button brings up a GenericObjectEditor dialog box. A click with the right mouse button (or Alt + Shift +left click) brings up a menu where one can choose, either to display the properties in a GenericObjectEditor dialog box, or to copy the current setup string to the clipboard.

The GenericObjectEditor Dialog Box

The GenericObjectEditor dialog box lets to configure a filter. The same kind of dialog box is used to configure other objects, such as classifiers and clusters. The fields in the window reflect the available options.

i. Show properties: It has the same effect as left-clicking on the field, i.e., a dialog appears allowing to alter the settings.

ii. Copy configuration to clipboard: It copies the currently displayed configuration string to the system’s clipboard and therefore can be used anywhere else in WEKA or in the console. This is rather handy if one has to setup complicated, nested schemes.

iii. Enter configuration: It is the “receiving” end for configurations that got copied to the clipboard earlier on. This also allows transferring a filter setting from the Preprocess panel to a FilteredClassifier used in the Classify panel.

Left-Clicking on any of these gives an opportunity to alter the filters settings. For example, the setting may take a text string, in which case the string can be typed into the text field provided. Or it may give a drop-down box listing
several states to choose from. Or it may do something else, depending on the information required. Information on the options is provided in a tool tip if the mouse pointer is hovered on the corresponding field. More information on the filter and its options can be obtained by clicking on the More button in the About panel at the top of the GenericObjectEditor window. Some objects display a brief description of what they do in an About box, along with a More button. Clicking on the More button brings up a window describing what the different options do. Others have an additional button, Capabilities, which lists the types of attributes and classes the object can handle. At the bottom of the GenericObjectEditor dialog are four buttons. The first two, Open... and save... allow object configurations to be stored for future use. The Cancel button backs out without remembering any changes that have been made. Once one is satisfied with the object and settings chosen, clicking OK returns back to the main Explorer window.

**Applying Filters**

Once a filter is selected and configured, it can be applied to the data by pressing the Apply button at the right end of the Filter panel in the Preprocess panel. The Preprocess panel will then show the transformed data. The change can be undone by pressing the Undo button. The edit button can also be used to modify the data manually in a dataset editor. Finally, the Save button at the top right of the Preprocess panel saves the current version of the relation in file formats that can represent the relation, allowing it to be kept for future use. The “supervised filters” require a class attribute to be set, and some of the “unsupervised attribute filters” will skip the class attribute if one is set. Note that it is also possible to set Class to none, in which case no class is set.

2.6.2 **Classification**

Classification is used to train and test learning schemes that classify or perform regression.
2.6.2.1 Selecting a Classifier

At the top of the classify section is the Classifier box. This box has a text field that gives the name of the currently selected classifier, and its options. Clicking on the text box with the left mouse button brings up a GenericObjectEditor dialog box, just the same as for filters, which can be used to configure the options of the current classifier. With a right click (or Alt + Shift+ left click) string can be copied to the clipboard or display the properties in a GenericObjectEditor dialog box. The Choose button allows choosing one of the classifiers that are available in WEKA.

2.6.2.2 Test Options

To obtain a reliable estimate of classifier accuracy (or predictor accuracy in terms of error, there exist various test modes. Cross validation, splitting, using test data etc. are the various test modes available in WEKA.

The result of applying the chosen classifier will be tested according to the options that are set by clicking in the Test options box. There are four test modes:

i. Percentage split: In this method, the given data is divided into two independent sets, a training set and a test set. Typically, two-thirds of data are allocated to the training set, and the remaining one-third is allocated to the test set. The training set is used to derive the model, whose accuracy is estimated with the test set (Figure 2.5). The classifier is evaluated on how well it predicts a certain percentage of the data which is held out for testing. The amount of data held out depends on the value entered in the % field. Note: No matter which evaluation method is used, the model that is output is always the one build from all the training data.
ii. Supplied test set: In this test mode, a file is supplied as a test set. The classifier is evaluated on how well it predicts the class of a set of instances loaded from a file.

iii. Cross-validation: In k-fold cross validation, the initial data are randomly partitioned into k mutually exclusive subsets or “folds”, each of approximately of equal size (figure 2.6). Training and testing is performed k times. In first iteration first fold is reserved as the test set, and the remaining partitions are collectively used to train the model. In further iterations, the next fold will be the test set and the rest of them collectively used to train the model. Here each sample is used the same number of times for training and once for testing. For classification, the accuracy estimate is the overall number of correct classifications from the k iterations, divided by total number of tuples in the initial data. The classifier is evaluated by cross-validation, using the number of folds that are entered in the Folds text field.
The following figure 2.7 is an example of calculating Error Rate for 3 fold cross validation training & Testing.

iv. Use training set: In this method, the complete initial data set is used for training the model (figure 2.8). The classifier is evaluated on how well it predicts the class of the instances it was trained on.
Further testing options can be set by clicking on the More options... button:

a) **Output model:** The classification model on the full training set is output so that it can be viewed, visualized, etc. This option is selected by default.

b) **Output per-class statistics:** The precision/recall and true/false statistics for each class are output. This option is also selected by default.

c) **Output entropy evaluation measures:** Entropy evaluation measures are included in the output. This option is not selected by default.

d) **Output confusion matrix:** The confusion matrix of the classifier’s predictions is included in the output. This option is selected by default.

e) **Store predictions for visualization:** The classifier’s predictions are remembered so that they can be visualized. This option is selected by default.

f) **Output predictions:** The predictions on the evaluation data are output.

g) **Output additional attributes:** If additional attributes need to be output alongside the predictions, e.g., an ID attribute for tracking misclassifications, then the index of this attribute can be specified here. The usual WEKA ranges are supported, “first” and “last” are therefore valid indices as well (example: “first-3, 6, 8, 12-last”).
h) **Cost-sensitive evaluation:** The errors are evaluated with respect to a cost matrix.

i) **Random seed for xval / % Split:** This specifies the random seed used when randomizing the data before it is divided up for evaluation purposes.

j) **Preserve order for % Split:** This suppresses the randomization of the data before splitting into train and test set.

k) **Output source code:** If the classifier can output the built model as Java source code, one can specify the class name also. The code will be printed in the “Classifier output” area.

2.6.2.3 The Class Attribute

The classifiers in WEKA are designed to be trained to predict a single ‘class’ attribute, which is the target for prediction. Some classifiers can only learn nominal classes; others can only learn numeric classes; still others can learn both.

By default, the class is taken to be the last attribute in the data. If a classifier is to be trained to predict a different attribute, click on the box below the Test options box to bring up a drop-down list of attributes to choose from.

2.6.2.4 Training a Classifier

Once the classifier, test options and class have all been set, the learning process is started by clicking on the Start button. While the classifier is busy being trained, the little bird moves around. The training process can be stopped at any time by clicking on the Stop button. When training is complete, several things happen. The Classifier output area to the right of the display is filled with text describing the results of training and testing. A new entry appears in the Result list box. Figure 2.9 depicts the result of Linear regression Model and figure 2.10 depicts classifier accuracy along with confusion matrix.
2.6.2.5 The Classifier Output Text

The text in the Classifier output area has scroll bars allowing browsing the results. Clicking with the left mouse button into the text area, while holding
Alt and Shift, brings up a dialog that enables to save the displayed output in a variety of formats (currently, BMP, EPS, JPEG and PNG). Of course, the Explorer window can also be resized to get a larger display area. The output is split into following several sections:

i. Run information. A list of information giving the learning scheme options, relation name, instances, attributes and test mode that were involved in the process.

ii. Classifier model (full training set). A textual representation of the classification model that was produced on the full training data.

iii. The results of the chosen test mode are broken down thus:

   a) Summary: A list of statistics summarizing how accurately the classifier was able to predict the true class of the instances under the chosen test mode.

   b) Detailed Accuracy by Class: A more detailed per-class breaks down of the classifier’s prediction accuracy.

   c) Confusion Matrix: It shows how many instances have been assigned to each class. Elements show the number of test examples whose actual class is the row and whose predicted class is the column.

   d) Source code (optional): This section lists the Java source code if one chose “Output source code” in the “More options” dialog.

2.6.2.6 The Result List

After training several classifiers, the result list will contain several entries. If the entries are left clicked, it flicks back and forth between the various results that have been generated. Pressing Delete removes a selected entry from the results. Right-clicking an entry invokes a menu containing these items:

i. View in main window: It shows the output in the main window (just like left-clicking the entry).
ii. View in separate window: It opens a new independent window for viewing the results.

iii. Save result buffer: Brings up a dialog allowing saving a text file containing the textual output.

iv. Load model: Loads a pre-trained model object from a binary file.

v. Save model: Saves a model object to a binary file. Objects are saved in Java 'serialized object' form.

vi. Re-evaluate model on current test set: Takes the model that has been built and tests its performance on the data set that has been specified with the Set button under the Supplied test set option.

vii. Visualize classifier errors: Brings up a visualization window that plots the results of classification. Correctly classified instances are represented by crosses, whereas incorrectly classified ones show up as squares.

viii. Visualize tree or Visualize graph: Brings up a graphical representation of the structure of the classifier model, if possible (i.e. for decision trees or Bayesian networks). The graph visualization option only appears if a Bayesian network classifier has been built. In the tree visualizer, a menu can be brought up by right-clicking a blank area, pan around by dragging the mouse, and see the training instances at each node by clicking on it.

ix. Visualize margin curve: Generates a plot illustrating the prediction margin. The margin is defined as the difference between the probability predicted for the actual class and the highest probability predicted for the other classes. For example, boosting algorithms may achieve better performance on test data by increasing the margins on the training data.

x. Visualize threshold curve: Generates a plot illustrating the tradeoffs in prediction that are obtained by varying the threshold value between classes. For example, with the default threshold value of 0.5, the predicted probability of ‘positive’ must be greater than 0.5 for the instance to be predicted as ‘positive’. The plot can be used to visualize the
precision/recall trade-off, for ROC curve analysis (true positive rate vs false positive rate), and for other types of curves.

xi. Visualize cost curve: Generates a plot that gives an explicit representation of the expected cost.

xii. Plugins: This menu item only appears if there are visualization plugins available (by default: none).

2.6.2.7 Evaluating a classifier

Building a classifier is only one part of the equation, evaluating how well it performs is another important part. WEKA supports two types of evaluation:

i. Cross-validation: If one only has a single dataset and wants to get a reasonable realistic evaluation. Setting the number of folds equal to the number of rows in the dataset will give one leave-one-out cross-validation (LOOCV). The crossValidateModel method of the Evaluation class is used to perform cross-validation with an untrained classifier and a single dataset. Supplying an untrained classifier ensures that no information leaks into the actual evaluation. Even though it is an implementation requirement, that the buildClassifier method resets the classifier, it cannot be guaranteed that this is indeed the case (“leaky” implementation). Using an untrained classifier avoids unwanted side effects, as for each train/test set pair, a copy of the originally supplied classifier is used. Before cross-validation is performed, the data gets randomized using the supplied random number generator (java.util.Random). It is recommended that this number generator is “seeded” with a specified seed value. Otherwise, subsequent runs of cross-validation on the same dataset will not yield the same results, due to different randomization of the data.

ii. Dedicated test set: The test set is solely used to evaluate the built classifier. It is important to have a test set that incorporates the same (or similar) concepts as the training set, otherwise one will always end up with poor
performance. The evaluation step, including collection of statistics, is performed by the Evaluation class (package weka.classifiers).

2.6.2.8 Classifying instances

After a classifier setup has been evaluated and proven to be useful, a built classifier can be used to make predictions and label previously unlabeled data. After all the instances have been labeled, the newly labeled dataset gets written back to disk to a new file.

2.6.3 Clustering

2.6.3.1 Selecting a Cluster

Clicking on the clustering scheme listed in the Clusterer box at the top of the window brings up a GenericObjectEditor dialog with which to choose a new clustering scheme.

2.6.3.2 Cluster Modes

The Cluster mode box is used to choose what to cluster and how to evaluate the results. The first three options are the same as for classification: Use training set, Supplied test set and Percentage split—except that now the data are assigned to clusters instead of trying to predict a specific class. The fourth mode, Classes to clusters evaluation, compares how well the chosen clusters match up with a pre-assigned class in the data. The drop-down box below this option selects the class, just as in the Classify panel. An additional option in the Cluster mode box, the Store clusters for visualization tick box, determines whether or not it will be possible to visualize the clusters once training is complete. When dealing with datasets that are so large that memory becomes a problem it may be helpful to disable this option.
2.6.3.3 Ignoring Attributes

Often, some attributes in the data should be ignored when clustering. The Ignore attributes button brings up a small window that allows selecting which attributes are ignored. Clicking on an attribute in the window highlights it, holding down the SHIFT key selects a range of consecutive attributes, and holding down CTRL toggles individual attributes on and off. To cancel the selection, back out with the Cancel button. To activate it, click the Select button. The next time clustering is invoked, the selected attributes are ignored.

2.6.3.4 Working with Filters

The FilteredClusterer meta-clusterer offers the user the possibility to apply filters directly before the cluster is learned. This approach eliminates the manual application of a filter in the Preprocess panel, since the data gets processed on the fly. Useful if one needs to try out different filter setups.

2.6.3.5 Learning Clusters

The Cluster section, like the Classify section, has Start/Stop buttons, a result text area and a result list. These all behave just like their classification counterparts. Right-clicking an entry in the result list brings up a similar menu, except that it shows only two visualization options: Visualize cluster assignments and Visualize tree. The latter is grayed out when it is not applicable.

2.6.4 Associate

It is used to learn association rules for the data. This panel contains schemes for learning association rules, and the learners are chosen and configured in the same way as the clusters, filters, and classifiers in the other panels. Once appropriate parameters for the association rule learner have been set, click the Start button. When complete, right-clicking on an entry in the result list allows the results to be viewed or saved.
2.6.5 Selecting Attributes

Select the most relevant attributes in the data. Attribute selection involves searching through all possible combinations of attributes in the data to find which subset of attributes works best for prediction. To do this, two objects must be set up: an attribute evaluator and a search method. The evaluator determines what method is used to assign a worth to each subset of attributes. The search method determines what style of search is performed.

The Attribute Selection Mode box has two options:

i. Use full training set: The worth of the attribute subset is determined using the full set of training data.

ii. Cross-validation: The worth of the attribute subset is determined by a process of cross-validation. The Fold and Seed fields set the number of folds to use and the random seed used when shuffling the data. As with Classify. There is a drop-down box that can be used to specify which attribute to treat as the class.

After clicking, it starts running the attribute selection process. When it is finished, the results are output into the result area, and an entry is added to the result list. Right-clicking on the result list gives several options. The first three, (View in main window, View in separate window and Save result buffer), are the same as for the classify panel. It is also possible to visualize reduced data, or an attribute transformer such as Principal Components can be used to visualize transformed data. The reduced/transformed data can be saved to a file with the Save reduced data. / Save transformed data... option.

2.6.6 Visualize

View an interactive 2D plot of the data. Visualization section allows visualizing 2D plots of the current relation.
2.6.6.1 The Scatter Plot Matrix

When Visualize panel is selected, it shows a scatter plot matrix for all the attributes, colour coded according to the currently selected class. It is possible to change the size of each individual 2D plot and the point size, and to randomly jitter the data (to uncover obscured points). It also possible to change the attribute used to colour the plots, to select only a subset of attributes for inclusion in the scatter plot matrix, and to sub sample the data.

2.6.6.2 Selecting an individual 2D Scatter Plot

A click on a cell in the scatter plot matrix will bring up a separate window with a visualization of the selected scatter plot. Data points are plotted in the main area of the window. At the top is two drop-down list buttons for selecting the axes to plot. The one on the left shows which attribute is used for the x-axis; and the right shows which is used for the y-axis. Beneath the x-axis selector is a drop-down list for choosing the colour scheme. Below the plot area, a legend describes what values the colours correspond to. If the values are discrete, the colour used for each one can be modified by clicking on them and making an appropriate selection in the window that pops up. To the right of the plot area is a series of horizontal strips. Each strip represents an attribute, and the dots within it show the distribution of values of the attribute. These values are randomly scattered vertically to help see concentrations of points. The axes used in the main graph can be chosen by clicking on these strips. The ‘X’ and ‘Y’ written beside the strips shows what the current axes are (‘B’ is used for ‘both X and Y’).

Above the attribute strips is a slider labeled Jitter, which a random displacement is given to all points in the plot. Dragging it to the right increases the amount of jitter, this is useful for spotting concentrations of points. Without jitter, a million instances at the same point would look no different to just a single lonely instance.
2.6.6.3 Selecting Instances

There may be situations where it is helpful to select a subset of the data using the visualization tool. (A special case of this is the UserClassifier in the Classify panel, which lets one build one’s own classifier by interactively selecting instances.) Below the y-axis selector button is a drop-down list button for choosing a selection method. A group of data points can be selected in four ways:

i. Select Instance: Clicking on an individual data point brings up a window listing its attributes. If more than one point appears at the same location, more than one set of attributes is shown.

ii. Rectangle: A rectangle can be created, by dragging, that selects the points inside it.

iii. Polygon: A free-form polygon can be built that selects the points inside it. Left-click to add vertices to the polygon, right-click to complete it. The polygon will always be closed off by connecting the first point to the last.

iv. Polyline: A polyline can be built that distinguishes the points on one side from those on the other. Left-click to add vertices to the polyline, right-click to finish. Once an area of the plot has been selected, it turns grey. At this point, clicking the Submit button removes all instances from the plot except those within the grey selection area. Clicking on the Clear button erases the selected area without affecting the graph. Once any points have been removed from the graph, the Submit button changes to a Reset button. This button undoes all previous removals and returns back to the original graph with all points included. Finally, clicking the Save button allows to save the currently visible instances to a new ARFF file.

2.7 Experimenter

It is an environment for performing experiments and conducting statistical tests between learning schemes.
2.8 Knowledge Flow

This environment supports essentially the same functions as the Explorer but with a drag-and-drop interface. One advantage is that it supports incremental learning.

2.9 SimpleCLI

It provides a simple command-line interface that allows direct execution of WEKA commands for operating systems that do not provide their own command line interface.

Once the tabs are active, clicking on them flicks between different screens, on which the respective actions can be performed. The bird stays visible regardless of which section one is in. The following paragraphs discuss the status box, log button, WEKA status icon and graphical output controls available in the bottom area of the window.

i. Status Box

The status box appears at the very bottom of the window. It displays messages that keep informed about what’s going on. For example, if the Explorer is busy loading a file, the status box will say that.

ii. Log Button

Clicking on this button brings up a separate window containing a scrollable text field. Each line of text is stamped with the time it was entered into the log. WEKA keeps a log record of all the action performed. For people using the command line or the SimpleCLI, the log now also contains the full setup strings for classification, clustering, attribute selection, etc., so that it is possible to copy/paste them elsewhere.
iii. WEKA Status Icon

To the right of the status box is the WEKA status icon. When no processes are running, the bird sits down and takes a nap. The number beside the × symbol gives the number of concurrent processes running. When the system is idle it is zero, but it increases as the number of processes increases. When any process is started, the bird gets up and starts moving around. If it’s standing but stops moving for a long time, something has gone wrong! In that case one should restart the WEKA Explorer.

iii. Graphical output

Graphical displays in WEKA support saving the output to a file. A dialog for saving the output can be brought up with Alt + Shift + left-click. Supported formats are currently Windows Bitmap, JPEG, PNG and EPS (encapsulated Postscript). The dialog also allows specifying the dimensions of the generated image.

2.10 WEKA GUI Interface sections

The WEKA GUI Interface menu consists of following four sections:

2.10.1 Program

i. LogWindow: Opens a log window that captures all that is printed to stdout or stderr. It is useful for environments like MS Windows, where WEKA is normally not started from a terminal.

ii. Memory Usage: Display in the log box the amount of memory available to WEKA.

iii. Run garbage collector: Force the Java garbage collector to search for memory that is no longer needed and free it up, allowing more memory for new tasks. Note that the garbage collector is constantly running as a background task anyway.
iv. Exit: Closes WEKA.

2.10.2 Tools other useful applications.

i. ArffViewer: An MDI application for viewing ARFF files in spread-sheet format.

ii. SqlViewer: Represents an SQL worksheet, for querying databases via JDBC.

iii. Bayes net editor: An application for editing, visualizing and learning Bayes nets.

2.10.3 Visualization

It is the ways of visualizing data with WEKA.

i. Plot: For plotting a 2D plot of a dataset.

ii. ROC: Displays a previously saved ROC curve.

iii. TreeVisualizer: For displaying directed graphs, e.g., a decision tree.

iv. GraphVisualizer: Visualizes XML BIF or DOT format graphs, e.g., for Bayesian networks.

v. BoundaryVisualizer: Allows the visualization of classifier decision boundaries in two dimensions.

2.10.4 Help

Online resources for WEKA can be as follows:

i. WEKA homepage: Opens a browser window with WEKA’s home page.

ii. HOWTOs, code snippets, etc: The general WekaWiki [2], containing lots of examples and HOWTOs around the development and use of WEKA.

iii. Weka on SourceforgeWEKA’s: project homepage on Sourceforge.net.
2.11 Accuracy and Error Measures

There exist various measures to estimate the accuracy of a classifier or predictor, to compare the accuracy of different methods for building more than one classifier. The following section 2.11.1 describes predictor error measures and section 2.11.2 describes measures for computing classifier accuracy.

2.11.1 Predictor Error Measures

Let D is dataset with values \((x_1, y_1), (x_2, y_2), \ldots, (x_d, y_d)\). Let \(y_i\) is the actual value and \(\hat{y}_i\) is the predicted value for the independent variable \(x_i\), \(y\) is the mean value of \(y_i\) and \(x\) is the mean value of \(x_i\). Since predictor returns a continuous value rather than a categorical label, it is difficult to say exactly whether the predicted value \(\hat{y}_i\) for \(x_i\) is correct. Instead of focusing on whether \(\hat{y}_i\) is an exact match with \(y_i\), it instead look at how far off the predicted value is from the actual known value. Loss functions measure the error between \(y_i\) and predicted value \(\hat{y}_i\). The most common loss functions are:

\[
\text{Absolute error} = |y_i - \hat{y}_i| \quad (2.3)
\]

\[
\text{Squared error} = (y_i - \hat{y}_i)^2 \quad (2.4)
\]

Based on the above equations 2.3 and 2.4, the test error (rate), or generalization error, is the average loss over the test set. Thus, the following error rates are obtained:

\[
\text{Mean Absolute Error} = \frac{\sum_{i=1}^{d}(y_i - \hat{y}_i)}{d} \quad (2.5)
\]

\[
\text{Mean Squared Error} = \frac{\sum_{i=1}^{d}(y_i - \hat{y}_i)^2}{d} \quad (2.6)
\]

The mean squared error exaggerates the presence of outliers, while the Mean Absolute Error (MAE) does not. If the square root of the mean squared error were to be taken, the resulting error measure is called the Root Mean
**Squared Error (RMSE).** This is useful in that it allows the error measured to be of the same magnitude as the quantity being predicted.

Some time one may desire the error to be relative to what it would have been if it had just predicted \( y \) the mean value of \( y_i \) from the training data \( D \). That is, one can normalize the total loss by dividing the total loss incurred from always predicting the mean. Relative measures of error include:

\[
\text{Root Relative Squared Error} = \sqrt{\frac{\sum_{d=1}^{D} (y_i - y_i')^2}{\sum_{d=1}^{D} (y_i - y)^2}}
\]

\[
\text{Root Relative Squared Error} = \sqrt{\frac{\sum_{d=1}^{D} (y_i - y_i')^2}{\sum_{d=1}^{D} (y_i - y)^2}}
\] (2.8)

### 2.11.2 Classifier Accuracy Measures

The accuracy of a classifier on a given test set is the percentage of test set tuples that are correctly classified by the classifier.

The *confusion matrix* is a visualization tool typically used in supervised learning. It is used to represent the test result of a prediction model. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class. One benefit of confusion matrix is that it is easy to see if the system is confusing two classes (i.e. commonly mislabeling one as another).

When a data set is unbalanced (when the number of samples in different classes vary greatly) the error rate of a classifier is not representative of the true performance of the classifier. This can easily be understood by an example: If there are for example 990 samples from class A and only 10 samples from class B, the classifier can easily be biased towards class A. If the classifier classifies all the samples as class A, the accuracy will be 99%. This
is not a good indication of the classifier’s true performance. The classifier has a 100% recognition rate for class A but a 0% recognition rate for class B.

The entries in the confusion matrix (table 2.1) have the following meaning in the context of the study:

- a is the number of correct predictions that an instance is negative,
- b is the number of incorrect predictions that an instance is positive,
- c is the number of incorrect of predictions that an instance negative, and
- d is the number of correct predictions that an instance is positive.

### Table 2.1 Confusion Matrix

<table>
<thead>
<tr>
<th>Confusion Matrix</th>
<th>Predicted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Negative</td>
</tr>
<tr>
<td>Actual</td>
<td></td>
</tr>
<tr>
<td>Negative</td>
<td>a</td>
</tr>
<tr>
<td>Positive</td>
<td>c</td>
</tr>
</tbody>
</table>

Several standard terms have been defined for the 2 class matrix:

The accuracy (AC) is the proportion of the total number of predictions that were correct. It is determined using the equation 2.9:

\[
AC = \frac{a + d}{a + b + c + d} \tag{2.9}
\]

The recall, sensitivity or true positive rate (TP) is the proportion of positive cases that were correctly identified, is calculated using the equation 2.10:

\[
TP = \frac{d}{c + d} \tag{2.10}
\]
The false positive rate (FP) is the proportion of negatives cases that were incorrectly classified as positive, is calculated using the equation 2.11:

\[ FP = \frac{b}{a+b} \]  

(2.11)

The Precision is defined as the percentage of true positive that are actually true positive is calculated using equation 2.12:

\[ Precision = \frac{d}{b+d} \]  

(2.12)

The accuracy determined using equation 2.9 may not be an adequate performance measure when the number of negative cases is much greater than the number of positive cases [133]. Suppose there are 1000 cases, 995 of which are negative cases and 5 of which are positive cases. If the system classifies them all as negative, the accuracy would be 99.5%, even though the classifier missed all positive cases. So the other performance measures as indicated from 2.10 to 2.12.

2.12 Conclusions

This chapter illustrates the various techniques of data mining and the data mining tool, WEKA 3-6-2 used for the testing and implementation of the models.