Classification algorithm in Data mining: An Overview

S.Neelamegam[#1], Dr.E.Ramaraj[*2]

[#1]*M.phil Scholar, Department of Computer Science and Engineering, Alagappa University, Karaikudi.*

[*2] *Professor, Department of Computer Science and Engineering, Alagappa University, Karaikudi.*

Abstract— Data Mining is a technique used in various domains to give meaning to the available data Classification is a data mining (machine learning) technique used to predict group membership for data instances. In this paper, we present the basic classification techniques. Several major kinds of classification method including decision tree, Bayesian networks, k-nearest neighbour classifier, Neural Network, Support vector machine. The goal of this paper is to provide a review of different classification techniques in data mining.

Keywords— Data mining, classification, Supper vector machine (SVM), K-nearest neighbour (KNN), Decision Tree.

I. INTRODUCTION

The development of Information Technology has generated large amount of databases and huge data in various areas. The research in databases and information technology has given rise to an approach to store and manipulate this precious data for further decision making. Data mining is a process of extraction of useful information and patterns from huge data. It is also called as knowledge discovery process, knowledge mining from data, knowledge extraction or data pattern analysis [1]. The Classification is the one of the major role in Data mining. Basically classification is a 2-step process; the first step is supervised learning for the sake of the predefined class label for training data set. Second step is classification accuracy evaluation. Likewise data prediction is also 2-step process. All the experiments are conducted by the Orange Data mining tool, and the input data sets are referring from UCI machine learning repository.

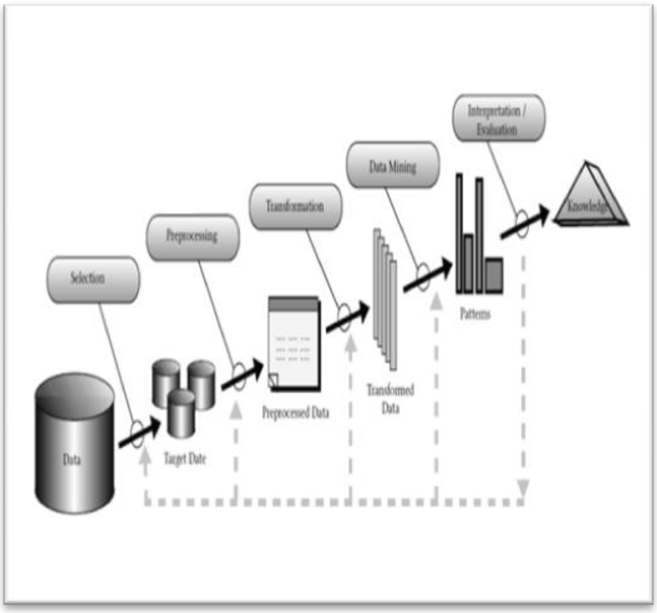

Figure 1. Knowledge discovery Process

Data mining is a logical process that is used to search through large amount of data in order to find useful data. The goal of this technique is to find patterns that were previously unknown. Once these

Patterns are found they can further be used to make certain decisions for development of their businesses.

Three steps involved are

- □ Exploration
- □ Pattern identification
- □ Deployment

Exploration: In the first step of data exploration data is cleaned and transformed into another form, and important variables and then nature of data based on the problem are determined.

Pattern Identification: Once data is explored, refined and defined for the specific variables the second step is to form pattern identification. Identify and choose the patterns which make the best prediction.

Deployment: Patterns are deployed for desired outcome.

II. CLASSIFICATION

Data mining algorithms can follow three different learning approaches: supervised, unsupervised, or semi-supervised. In supervised learning, the algorithm works with a set of examples whose labels are known. The labels can be nominal values in the case of the classification task, or numerical values in the case of the regression task. In unsupervised learning, in contrast, the labels of the examples in the dataset are unknown, and the algorithm typically aims at grouping examples according to the similarity of their attribute values, characterizing a clustering task. Finally, semi-supervised learning is usually used when a small subset of labelled examples is available, together with a large number of unlabeled examples.

The classification task can be seen as a supervised technique where each instance belongs to a class, which is indicated by the value of a special goal attribute or simply the class attribute. The goal attribute can take on categorical values, each of them corresponding to a class. Each example consists of two parts, namely a set of predictor attribute values and a goal attribute value. The former are used to predict the value of the latter. The predictor attributes should be relevant for predicting the class of an instance. In the classification task the set of examples being mined is divided into two mutually exclusive and exhaustive sets, called the training set and the test set. The classification process is correspondingly divided into two phases: training, when a classification model is built from the training set, and testing, when the model is evaluated on the test set. In the training phase the algorithm has access to the values of both predictor attributes and the oal attribute for all examples of the training set, and it uses that information to build a classification model. This model represents classification knowledge – essentially, a relationship between predictor attribute values and classes – that allows the prediction of the class of an example given its predictor attribute values. For testing, the test set the class values of the examples is not shown. In the testing phase, only after a prediction is made is the algorithm allowed to see the actual class of the just-classified example. One of the major goals of a classification algorithm is to maximize the predictive accuracy obtained by the classification model when classifying examples in the test set unseen during training.

Classification is the task of generalizing known structure to apply to new data. For example, an email program might attempt to classify an email as legitimate or spam.

Common algorithms include

- Decision Tree,
- K-Nearest Neighbor,
- Support Vector Machines,
- Naive Bayesian Classification,
- Neural Networks.

Decision Tree

A Decision Tree Classifier consists of a decision tree generated on the basis of instances. A decision tree is a classifier expressed as a recursive partition of the instance space. The decision tree [4] 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 an internal or test node. All other nodes are called leaves (also known as terminal or decision nodes). In a decision tree, each internal node splits the instance space into two or more sub-spaces a certain discrete function of the input attributes values.

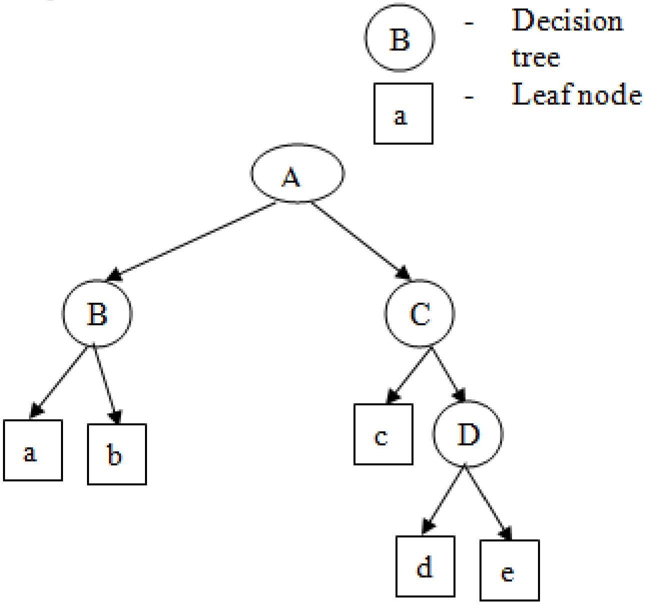

Figure 2: Decision tree model.

The root and the internal nodes are associated with attributes, leaf nodes are associated with classes. Basically, each non-leaf node has an outgoing branch for each possible value of the attribute associated with the node. To determine the class for a new instance using a decision tree, beginning with the root, successive internal nodes are visited until a leaf node is reached. At the root node and at each internal node, a test is applied. The outcome of the test determines the branch traversed, and the next node visited. The class for the instance is the class of the final leaf node.

The estimation criterion [5] in the decision tree algorithm is the selection of an attribute to test at each decision node in the tree. The goal is to select the attribute that is most useful for classifying examples. A good quantitative measure of the worth of an attribute is a statistical property called information gain that measures how well a given attribute separates the training examples according to their target classification. This measure is used to select among the candidate attributes at each step while growing the tree.

K-Nearest Neighbor Classifiers (KNN)

K-Nearest neighbor classifiers are based on learning by analogy. The training samples are described by n dimensional numeric attributes. Each sample represents a point in an n-dimensional space. In this way, all of the training samples are stored in an n-dimensional pattern space. When given an unknown sample, a k-nearest neighbour classifier searches the pattern space for the k training samples that are closest to the unknown sample. "Closeness" is defined in terms of Euclidean distance, where the Euclidean distance, where the Euclidean distance between two points,

X=(x1,x2,……,xn) and Y=(y1,y2,….,yn) is d(X, Y)= 2

$$d(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i \ - y_i\ )^2}$$

slower at classification since all computation is delayed to that time. Unlike decision tree induction and backpropagation, nearest neighbor classifiers assign equal weight to each attribute. This may cause confusion when there are many irrelevant attributes in the data. Nearest neighbor classifiers can also be used for prediction, that is, to return a real-valued prediction for a given unknown sample. In this case, the classifier returns the average value of the real-valued associated with the k nearest neighbors of the unknown sample. The *k-nearest neighbors' algorithm* is amongest the simplest of all machine learning algorithms. An object is classified by a majority vote of its neighbors, with the object

being assigned to the class most common amongst its *k* nearest neighbors. *k* is a positive integer, typically small. If *k* = 1, then the object is simply assigned to the class of its nearest neighbor. In binary (two class) classification problems, it is helpful to choose *k* to be an odd number as this avoids tied votes. The same method can be used for regression, by simply assigning the property value for the object to be the average of the values of its *k* nearest neighbors. It can be useful to weight the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones.

The neighbors are taken from a set of objects for which the correct classification (or, in the case of regression, the value of the property) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. In order to identify neighbors, the objects are represented by position vectors in a multidimensional feature space. It is usual to use the Euclidian distance, though other distance measures, such as the Manhanttan distance could in principle be used instead.

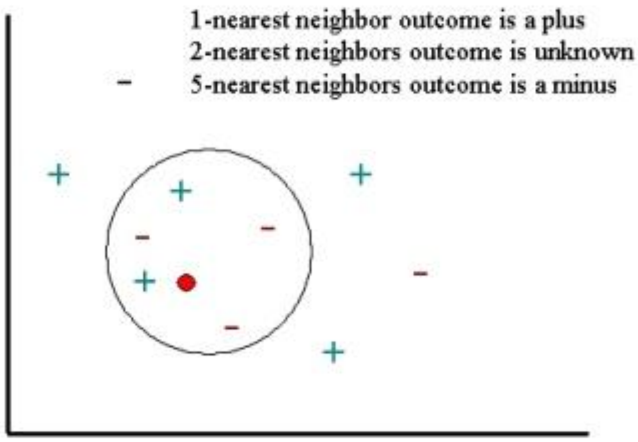
1-nearest neighbor outcome is a plus
2-nearest neighbors outcome is unknown
5-nearest neighbors outcome is a minus

Figure 3: k-nearest neighbour model

The *k*-nearest neighbour algorithm is sensitive to the local structure of the data. The unknown sample is assigned the most common class among its k nearest neighbors. When k=1, the unknown sample is assigned the class of the training sample that is closest to it in pattern space. Nearest neighbour classifiers are instance-based or lazy learners in that they store all of the training samples and do not build a classifier until a new(unlabeled) sample needs to be classified. This contrasts with eager learning methods, such a decision tree induction and back propagation, which construct a generalization model before receiving new samples to classify. Lazy learners can incur expensive computational costs when the number of potential neighbours (i.e.,stored training samples)with which to compare a given unlabeled sample is great. Therefore, they require efficient indexing techniques. An expected lazy learning methods are faster at a training than eager methods. When using a k-nearest neighbor algorithm, choosing an appropriate k value is significant. If the k value is too small it is susceptible to overfitting and would misclassify some traditionally easy to classify situations. For example imagine a cluster of records that all have a class label called "plus" except for one point in the cluster labelled as "minus". If a k of one were chosen for an input that is in the cluster, but it just so happens to be closest to the minus, then there is a good chance that that point was misclassified. This is evident by the fact that if k was 2 or more the resulting classification would be different. As well as having a k value that is too small it is important to choose a value that isn't too large as it can also lead to misclassification. This can be seen in a situation with a clust of one class surround by a cluster of another class. Even if the input is right in the middle of the first cluster if one looks at too many points is possible it starts to count the records from the surrounding cluster as well.

Support Vector Machine (SVM)

SVM was first introduced by Vapnik [6] and has been very effective method for regression, classification and general pattern recognition. It is considered a good classifier because of its high generalization performance without the need to add a priori knowledge, even when the dimension of the input space is very high. It is considered a good classifier because of its high generalization performance without the need to add a priori knowledge, even when the dimension of the input space is very high. The aim of SVM is to find the best classification function to distinguish between members of the two classes in the training data. The metric for the concept of the "best" classification function can be realized geometrically. For a linearly separable dataset, a linear classification function corresponds to a separating hyperplane *f(x)* that passes through the middle of the two classes, separating the two.

Once this function is determined, new data instance f(*xn)* can be classified by simply testing the sign of the function *f* (*xn*); *xn* belongs to the positive class if *f(xn)*>0.

Because there are many such linear hyperplanes, SVM guarantee that the best such function is found by maximizing the margin between the two classes. Intuitively, the margin is defined as the amount of space, or separation between the two classes as defined by the hyperplane. Geometrically, the margin corresponds to the shortest distance between the closest data points to a point on the hyperplane. To ensure that the maximum margin hyperplanes are actually found, an SVM classifier attempts to maximize the following function with respect to a and b

$$Lp = -\frac{1}{2} \| \underset{w}{\rightarrow} \| - \sum_{i=1}^{t} a_1 Y_1 \left(\underset{w}{\rightarrow} . \underset{x}{\rightarrow} + b \right) + \sum_{i=1}^{t} a_i$$

where *t* is the number of training examples, and *i* , *i* = 1, . . . , *t*, are non-negative numbers such that the derivatives of *LP* with respect to *i* are zero. *i* are the Lagrange multipliers and *LP* is called the Lagrangian. In this equation, the vectors and constant *b* define the hyperplane. A learning machine, such as the SVM, can be modeled as a function class based on some parameters.Different function classes can have different capacity in learning, which is represented by a parameter *h* known as the VC dimension. The VC dimension measures the maximum number of training examples where the function class can still be used to learn perfectly, by obtaining zero error rates on the training data, for any assignment of class labels on these points. It can be proven that the actual error on the future data is bounded by a sum of two terms. The first term is the training error, and the second term if proportional to the square root of the VC dimension *h*. Thus, if we can minimize *h*, we can minimize the future error, as long as we also minimize the training error, SVM can be easily extended to perform numerical calculations.

One of the initial drawbacks of SVM is its computational inefficiency. However, this problem is being solved with great success. One approach is to break a large optimization problem into a series of smaller problems, where each problem only involves a couple of carefully chosen variables so that the optimization can be done efficiently. The process iterates until all the decomposed optimization problems are solved successfully. A more recent approach is to consider the problem of learning an SVM as that of finding an approximate minimum enclosing ball of a set of instances.

Bayesian Networks

A Bayesian network (BN) consists of a directed, acyclic graph and a probability distribution for each node in that graph given its immediate predecessors [7]. A Bayes Network Classifier is based on a bayesian network which represents a joint probability distribution over a set of categorical attributes. It consists of two parts, the directed acyclic graph G consisting of nodes and arcs and the conditional probability tables. The nodes represent attributes whereas the arcs indicate direct dependencies. The density of the arcs in a BN is one measure of its complexity. Sparse BNs can represent simple probabilistic models (e.g., naïve Bayes models and hidden Markov models), whereas dense BNs can capture highly complex models. Thus, BNs provide a flexible method for probabilistic modelling.

Neural Network

An artificial neural network (ANN), often just called a "neural network" (NN), is a mathematical model or computational model based on biological neural networks, in other words, is an emulation of biological neural system. It consists of an interconnected group of artificial neurons and processes information using a connectionist approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase.

III. Conclusions

Data mining offers promising ways to uncover hidden patterns within large amounts of data. These hidden patterns can potentially be used to predict future behaviour. The availability of new data mining algorithms, however, should be met with caution. First of all, these techniques are only as good as the data that has been collected. Good data is the first requirement for good data exploration. Assuming good data is available, the next step is to choose the most appropriate technique to mine the data. In this paper, we present the basic classification techniques. Several major kinds of classification method including decision tree induction, Bayesian networks, k-nearest neighbour classifier and Neural Network.

REFERENCES

[1] CLUSTERING AND CLASSIFICATION: DATA MINING APPROACHES by Ed Colet

[2] Support Vector Machine Solvers L´eon Bottou NEC Labs America, Princeton, NJ 08540, USA Chih-Jen Lin cjlin@csie.ntu.edu.tw Department of Computer Science National Taiwan University, Taipei, Taiwan

[3] Orange biolab Documentation.

[4] Decision tree Lior Rokach Department of Industrial Engineering Tel-Aviv University, Oded Maimon Department of Industrial Engineering Tel-Aviv University maimon@eng.tau.ac.il

[5] Overview of Decision Trees by H.Hamilton. E. Gurak, L. Findlater W. Olive

[6] A Tutorial on Support Vector Machines for Pattern Recognition CHRISTOPHER J.C. BURGES

[7] Overview of Bayesian network approaches to model gene-environment interactions and cancer susceptibility Chengwei Su, Angeline Andrew, Margaret Karagas,Mark E. Borsuk.