P-2846

AN EFFICIENT APPROACH FOR CANCER CLASSIFICATION

A PROJECT REPORT

Submitted by

DHEEPAK.C **71205104004**

HARIKARTHIK.P.C **71205104012**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2009

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"AN EFFICIENT APPROACH FOR CANCER CLASSIFICATION"** is the bonafide work of **"C.DHEEPAK and P.C.HARIKARTHIK"** who carried out the project work under my supervision.

SIGNATURE
Prof. Dr.S.Thangasamy
DEAN &
HEAD OF THE DEPARTMENT
Department of
Computer Science & Engineering
Kumaraguru College of Technology
Coimbatore-641006

SIGNATURE
Mrs.C.Ramathilagam, M.E
SUPERVISOR
Lecturer
Department of
Computer Science & Engineering
Kumaraguru College of Technology
Coimbatore-641006

We examined the candidates with University Register Nos. 71205104004 and 71205104012 in the project viva-voce examination held on 27.04.2009

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We

Dheepak.C	71205104004
Harikarthik.P.C	71205104012

hereby declare that the project entitled **"AN EFFICIENT APPROACH FOR CANCER CLASSIFICATION"** is a record of original work done by us and to the best of our knowledge, a similar work has not been submitted to Anna University or any Institutions, for fulfillment of the requirement of the course study.

The report is submitted in partial fulfillment of the requirement for the award of the Degree of Bachelor of Computer Science and Engineering of Anna University, Chennai.

Place: Coimbatore
Date : 27.04.2009

(C.Dheepak)

(P.C.Harikarthik)

ACKNOWLEDGEMENT

We extend our sincere thanks to our Vice Principal, **Prof. R.Annamalai,M.E.,**Kumaraguru College of Technology, Coimbatore, for his incredible support for all our toil regarding the project.

We are deeply obliged to **Dr.S.Thangasamy, Ph.D.,** Dean, Professor and Head of Department of Computer Science & Engineering for his concern and implication during the project course.

We are indent to express our heartiest thanks to **Mrs.P.Devaki,M.E,.** Asst.Professor, the project coordinator who has helped us to overcome the perplexity while choosing the project.

We articulate our thankfulness to our guide **Mrs.C.Ramathilagam, M.E**, Lecturer, who rendered her valuable guidance throughout the project path and support to perform our project work extremely well.

We thank all the **Teaching and Non-teaching staffs** of our department for providing us the technical support for our project.

We also thank our **friends and family** who helped us to complete this project fruitfully.

ABSTRACT

This project aims at finding the smallest set of genes that can ensure highly accurate classification of cancers from micro array data by using supervised machine learning algorithms. The method involves two steps. In the first step, some important genes are chosen using a feature importance ranking scheme. In the second step, the classification capability of all simple combinations of those important genes is tested by using a good classifier. We use Back propagation for cancer classifications. Three data sets are used for classification mainly Lymphoma data set, SRBCT data set and Liver Cancer data set.

In all data sets, a small part of the data is missing. A k-nearest neighbor algorithm should be applied to fill those missing values. The significance of finding the minimum gene subsets is three-fold: 1) It greatly reduces the computational burden and "noise" arising from irrelevant genes. 2) It simplifies gene expression tests to include only a very small number of genes rather than thousands of genes, which can bring down the cost for cancer testing significantly. 3) It calls for further investigation into the possible biological relationship between these small numbers of genes and cancer development and treatment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

SVM	Support Vector Machine
NN	Neural network
BPN	Backpropagation
FNN	Fuzzy Neural Network
CV	Cross Validation

1. INTRODUCTION

1.1. Problem Definition:

The project, "An efficient approach for cancer classification" aims at finding the minimum number of genes from a large dataset that are responsible for causing cancer in a human being. This minimum subset of genes is found out using a fuzzy neural network algorithm, Backpropagation algorithm.

The process involves four main steps. The available dataset is preprocessed to fill in any missing values then gene ranking and gene combination are done. The output is then forwarded onto a Feed-Forward Neural Network and thus we obtain the most influential cancer causing genes as the result.

1.2. Goals of the project:

- To reduce computational burden and noise arising from irrelevant genes
- To simplify gene expression tests

1.3. Existing System:

Many existing classifiers classify cancer datasets successfully. But those classifiers have certain limitations. Tibshirani et al. successfully classified the lymphoma data set with only 48 genes by using a statistical method called nearest shrunken centroids with an accuracy of 100 percent. For the SRBCT data, Khan et al classified all 20 testing samples with 96

genes. They used a two-layered linear neural network. Tibshirani et al. applied nearest shrunken centriods to the SRBCT data set. They obtained 100 percent accuracy with 43 genes.

For the method of nearest shrunken centroids, it categorizes each sample to the class whose centriod is nearest to the sample. The difference between standard nearest centroids and nearest shrunken centroids is that the latter uses only some important genes rather than all the genes to calculate the centroids. Deutsch reduced the number of genes required to correctly classify the four cancer subtypes in the SRBCT data set to 12 genes. In the same year, Lee and Lee also obtained 100 percent accuracy in this data set with an SVM classifier and the separability-based gene importance ranking. They used at least 20 genes to obtain this result. At the same time, they generated three principal components (PCs) from the 20 top genes. Their SVM also obtained 100 percent accuracy in the space defined by these three principal components. For the liver cancer data set, Chen et al. used 3,180 genes (represented by 3,964 cDNA) to classify HCC and the nontumor samples.

1.4. Problem in existing System

In existing system Ambroise and McLachlan indicated that testing results could be overoptimistic, caused by the "selection bias," if the testing samples were not excluded from the gene selection process. In fact, taking advantage of testing samples in any step of the classifier-building process, e.g., feature selection, parameter tuning, model selection, etc., will induce bias. Therefore, to honestly evaluate a classifier in a given data set, the testing samples must be totally excluded from the classifier building process.

According to this criterion, almost all of the above reported results are overoptimistic because all of the above classifiers more or less used the information of the testing samples in their training process.

1.5. Proposed System:

We propose a simple yet very effective method that leads to accurate cancer classification using expressions of only a very few genes. Furthermore, we evaluated our methods in an honest way, which excluded the influence of the bias. In proposed system we use Backpropagation first for cancer classification. We carried out 5-fold cross-validation (CV) in the training data set to tune their parameters.

2. OVERVIEW

2.1 Data Mining:

Data mining is the process of extracting hidden patterns from data. As more data is gathered, with the amount of data doubling every three years, data mining is becoming an increasingly important tool to transform this data into information. It is commonly used in a wide range of profiling practices, such as marketing, surveillance, fraud detection and scientific discovery.

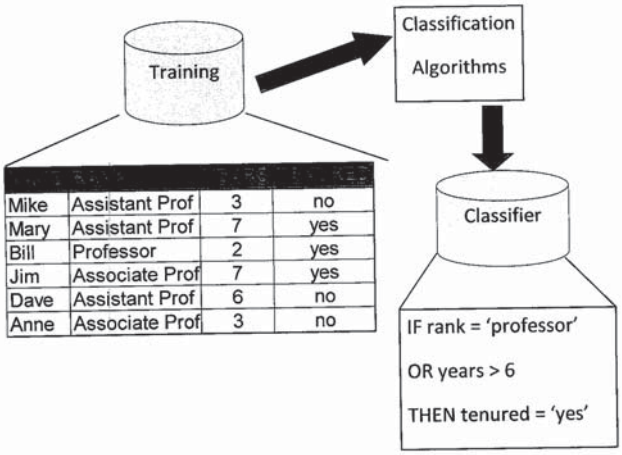
While data mining can be used to uncover hidden patterns in data samples that have been "mined", it is important to be aware that the use of a sample of the data may produce results that are not indicative of the domain. Data mining will not uncover patterns that are present in the domain, but not in the sample. There is a tendency for insufficiently knowledgeable "consumers" of the results to treat the technique as a sort of crystal ball and attribute "magical thinking" to it. Like any other tool, it only functions in conjunction with the appropriate raw material: in this case, indicative and representative data that the user must first collect. Further, the discovery of a particular pattern in a particular set of data does not necessarily mean that pattern is representative of the whole population from which that data was drawn. Hence, an important part of the process is the verification and validation of patterns on other samples of data.

2.2. Classification:

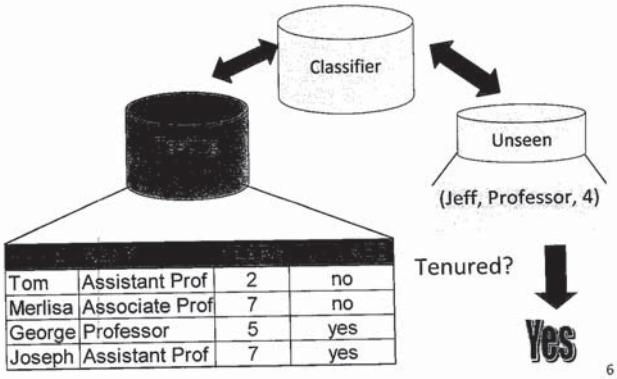
Data classification is a two step process. In the first step, a model is built describing a pre-determined set of data classes or concepts. The model is constructed by analyzing database tuples described by attributes. Each tuple is assumed to belong to a predefined class as determined by one of the attributes called the class label attribute. Data tuples are also referred to as samples, examples or objects. The data tuples are analyzed to build the model collectively from the training dataset. The individual tuples making up the training set are referred to as training samples and are randomly selected from the same population. Since the class label of each training sample is provided, the step is also known as supervised learning(ie the learning of the model is "supervised" in that it is told to which class each training sample belongs)

In the second step, the model is used for classification. First, the predictive accuracy of the model is estimated. The holdout method is a simple technique that uses a test set of class-labeled samples. These samples are randomly selected and are independent of the training sample. The accuracy of the model on a given test set is the percentage of test set samples that are correctly classified by the model. For each test sample, the known class label is compared with the learned model's class prediction for that sample. Note that if the accuracy of the model were estimated based on the training data set. This estimate could be optimistic since the learned model tends to overfit the data (ie it may have incorporated some particular anomalies of the training data that are not present in the overall sample population) Therefore, a test set is used.

Classification Process diagram- training and testing:



Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no



Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

2.3. Nearest Neighbor:

Clustering and the Nearest Neighbor prediction technique are among the oldest techniques used in data mining. Most people have an intuition that they understand what clustering is - namely that like records are grouped or clustered together. Nearest neighbor is a prediction technique that is quite similar to clustering - its essence is that in order to predict what a prediction value is in one record look for records with similar predictor values in the historical database and use the prediction value from the record that it "nearest" to the unclassified record.

2.3.1. A simple example of nearest neighbor:

A simple example of the nearest neighbor prediction algorithm is that if you look at the people in your neighborhood (in this case those people that are in fact geographically near to you). You may notice that, in general, you all have somewhat similar incomes. Thus if your neighbor has an income greater than $100,000 chances are good that you too have a high income. Certainly the chances that you have a high income are greater when all of your neighbors have incomes over $100,000 than if all of your neighbors have incomes of $20,000. Within your neighborhood there may still be a wide variety of incomes possible among even your "closest" neighbors but if you had to predict someone's income based on only knowing their neighbors you're best chance of being right would be to predict the incomes of the neighbors who live closest to the unknown person.

The nearest neighbor prediction algorithm works in very much the same way except that "nearness" in a database may consist of a variety of

factors not just where the person lives. It may, for instance, be far more important to know which school someone attended and what degree they attained when predicting income. The better definition of "near" might in fact be other people that you graduated from college with rather than the people that you live next to.

Nearest Neighbor techniques are among the easiest to use and understand because they work in a way similar to the way that people think - by detecting closely matching examples. They also perform quite well in terms of automation, as many of the algorithms are robust with respect to dirty data and missing data. As they enjoy similar levels of accuracy compared to other techniques the measures of accuracy such as lift are as good as from any other.

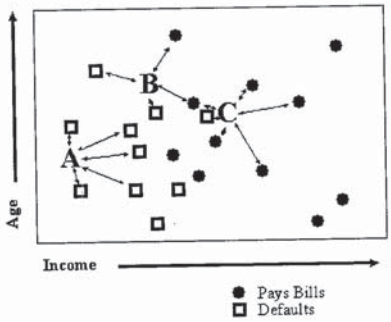
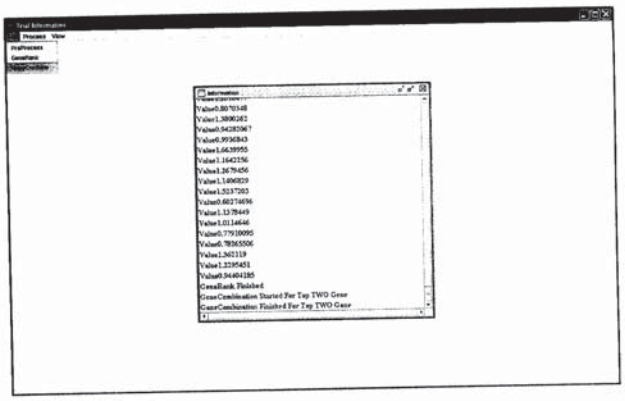
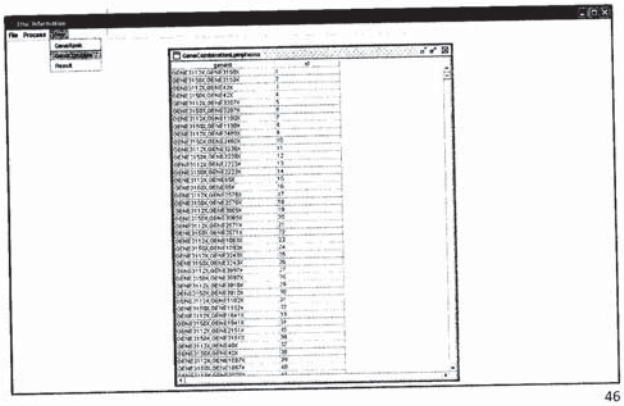


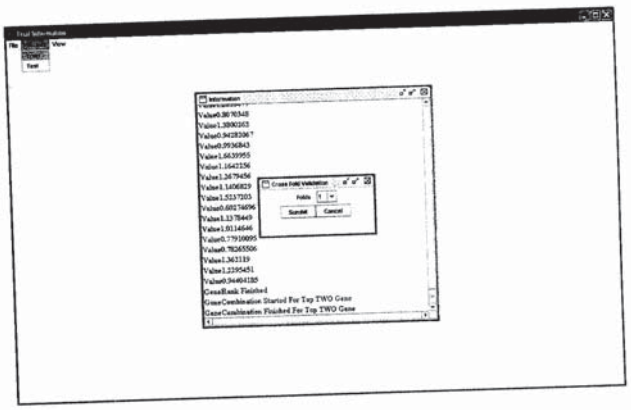
Figure *The nearest neighbors are shown graphically for three unclassified records: A, B, and C.*

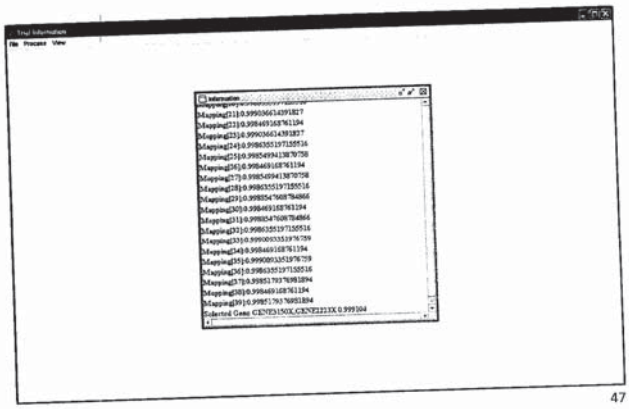
Gene Combination:

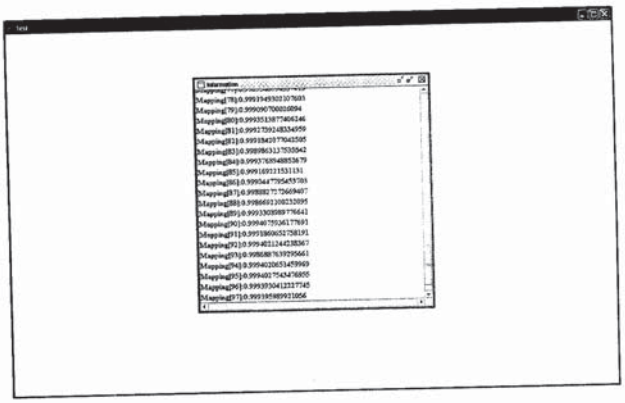




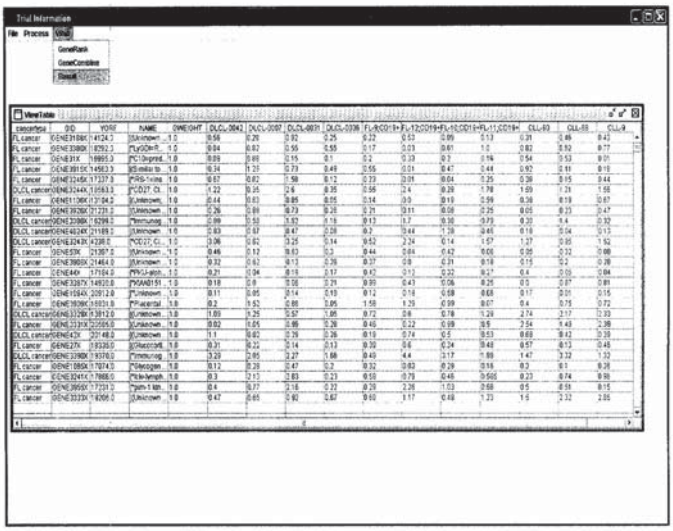
Backpropagation:







Result:



8. References:

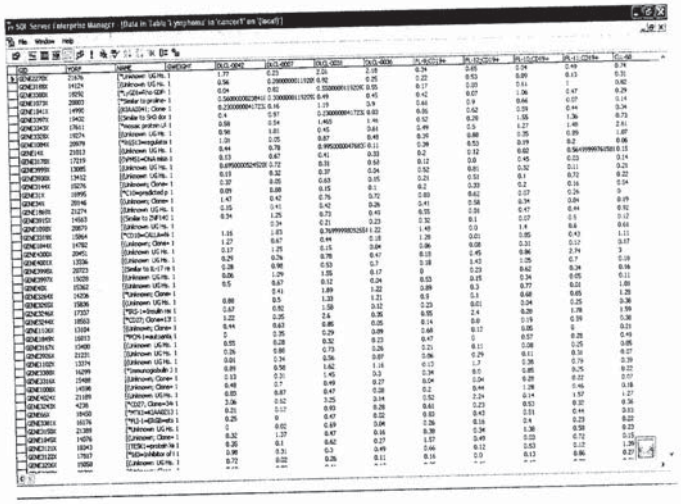
1. "Data Mining Concepts and Techniques", Jiawei Han, Micheline Kamber, Harcourt India / Morgan Kauffman, 2001.

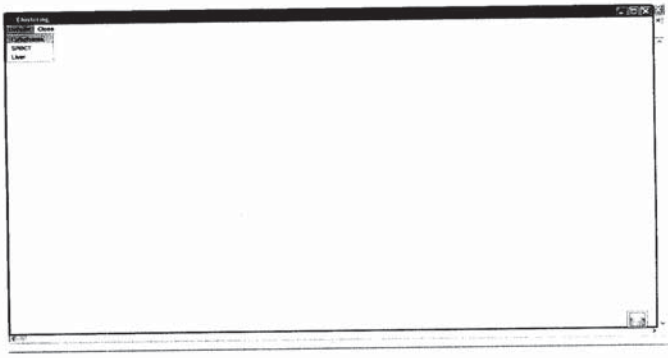
2. Java 2-The complete Reference, fifth edition, Herbert Schildt, Tata McGraw-Hill, 2002

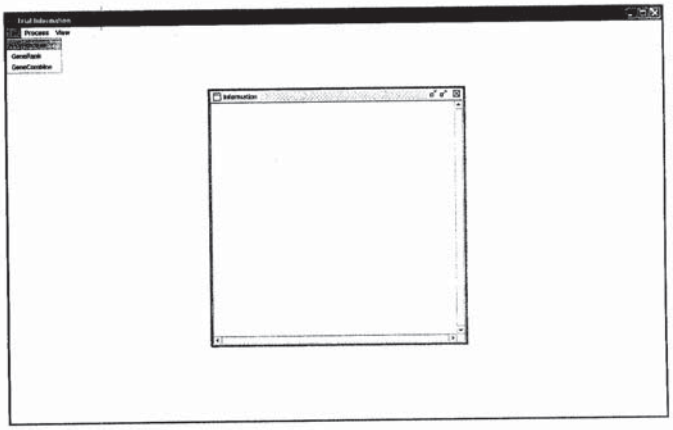
3."Backpropagation",Wikipedia,en.wikipedia.org/wiki/ Backpropagation

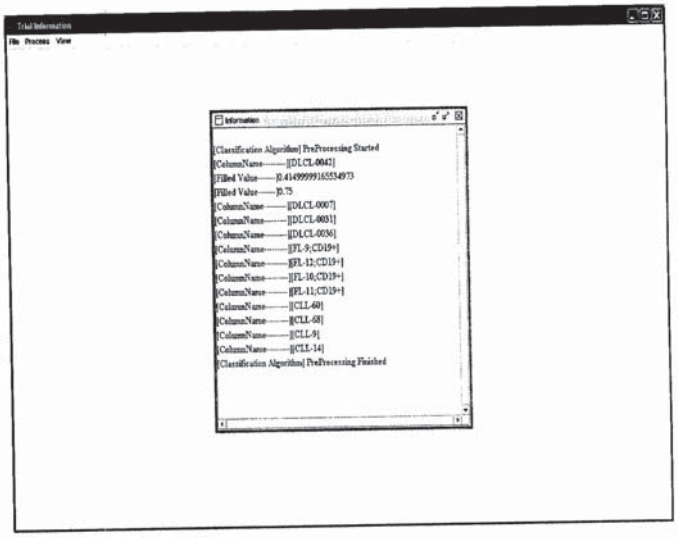
7.2 Screenshots:

Data Preprocessing:

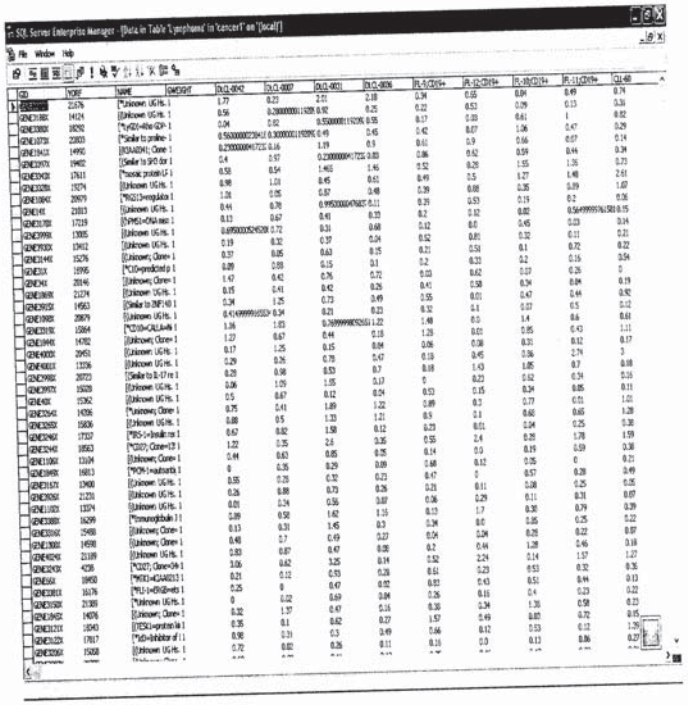






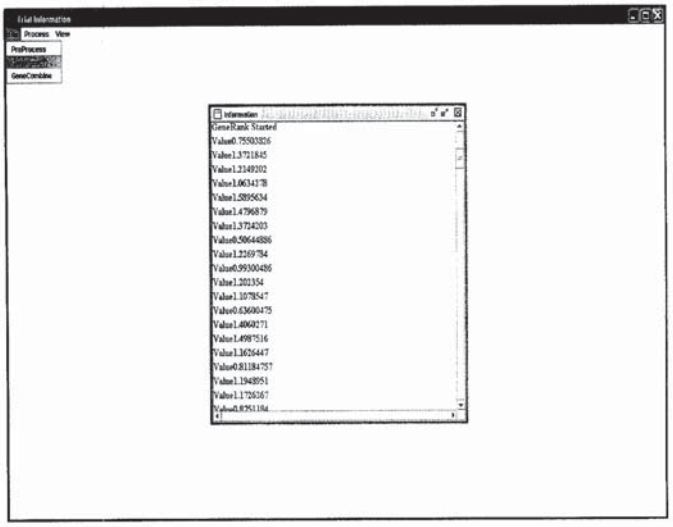


42

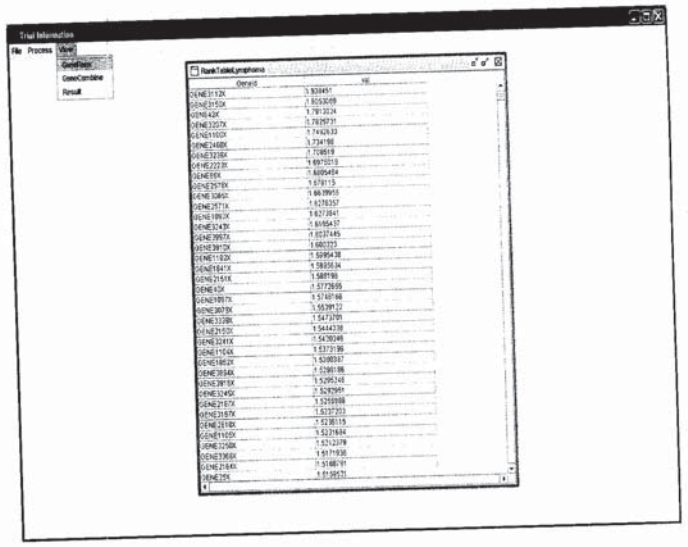


43

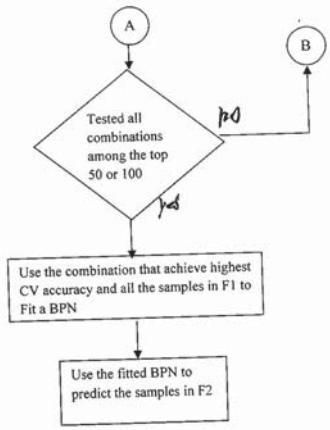
Gene Ranking:



44



45



6. CONCLUSION:

The process described in this project produces an output which has a high degree of efficiency and accuracy. Also the target value i.e., minimum number of genes that classify the cancer, has been obtained by the sequential execution of preprocessing, gene ranking, gene combination and finally through a FNN classifier. Thus we can say that the computational burden has been greatly reduced.

7. FUTURE ENHANCEMENTS:

The output obtained from this project can be further furnished by applying another classifier, Support Vector Machine (SVM). The accuracy of this alternate method can be compared with that of the method utilized in this project. The project calls for further investigation into the possible biological relationship between the small numbers of genes obtained and cancer development and treatment.

7. APPENDIX:

7.1 Sample Coding:

```
/*Pre-processing Lymphoma*/
import java.sql.*;
import java.util.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class LymphomaMain extends JFrame implements ActionListener
{
        JMenuBar mb;
        JMenu mnualg,mnufile,mnuview;
        JMenuItem
        mipre,migenerank,migenecom,mitrail,mireal,migeneranktab,migeneco
        mtab,misvm,milymres;
        JDesktopPane dp;
        database_conn db;
```



```
LymphomaMain()
{
        super("Trial Information");
        mb=new JMenuBar();
        mnufile=new JMenu("File");
        mipre=new JMenuItem("PreProcess");
        migenerank=new JMenuItem("GeneRank");
        migenecom=new JMenuItem("GeneCombine");
        mnufile.add(mipre);
        mnufile.add(migenerank);
        mnufile.add(migenecom);
        mnualg=new JMenu("Process");
        mitrail=new JMenuItem("Train");
        mireal=new JMenuItem("Test ");
        mnualg.add(mitrail);
        mnualg.add(mireal);
        mnuview=new JMenu("View");
        migeneranktab=new JMenuItem("GeneRank");
        migenecomtab=new JMenuItem("GeneCombine");
```

```
milymres=new JMenuItem("Result");
mnuview.add(migeneranktab);
mnuview.add(migenecomtab);
mnuview.add(milymres);
mb.add(mnufile);
mb.add(mnualg);
mb.add(mnuview);
setJMenuBar(mb);
mitrail.addActionListener(this);
mireal.addActionListener(this);
mipre.addActionListener(this);
migenerank.addActionListener(this);
migenecom.addActionListener(this);
migeneranktab.addActionListener(this);
migenecomtab.addActionListener(this);
milymres.addActionListener(this);
db=new database_conn();
Dimension ss=Toolkit.getDefaultToolkit().getScreenSize();
dp=new JDesktopPane();
```



```
dp.putClientProperty("JDsektopPane.dragMode","outline");
setContentPane(dp);
setSize(ss.width,ss.height);
setVisible(true);
textArea obj4=new textArea();
display(obj4);
}
public void actionPerformed(ActionEvent ae)
{
    if(ae.getSource()==mipre)
        {
            PreProcessingLymphoma pp=new
            PreProcessingLymphoma();
        }
    if(ae.getSource()==migenerank)
        {
            try
            {
```



```
                db.stat.executeUpdate("delete from
LymphomaRank");
                GeneRankLymphoma gg=new
GeneRankLymphoma();
            }
            catch(Exception e)
            {
                e.printStackTrace();
            }
        }
    if(ae.getSource()==migenecom)
        {
            try
            {
            db.stat.executeUpdate("truncate table genecombination");
                GeneCombineLymphoma gg=new
GeneCombineLymphoma();
            }
```



```
            catch(Exception e)
            {
                e.printStackTrace();
            }
        }
    if(ae.getSource()==mitrail)
        {
            try
            {
            db.stat.executeUpdate("truncate table temp");
            BackpropagationTrailLymphoma bb=new
BackpropagationTrailLymphoma();
            display(bb);
            }
            catch(Exception e)
            {
                e.printStackTrace();
            }
        }
```

```
if(ae.getSource()==mireal)
        {
            try
            {
            db.stat.executeUpdate("truncate table temp1");
            db.stat.executeUpdate("delete from graph");
            new BackpropagationRealLymphoma().show();
          }
            catch(Exception e)
            {
                e.printStackTrace();
            }
        }
    if(ae.getSource()==migeneranktab)
        {
            RankTableLymphoma rt=new RankTableLymphoma();
            display(rt);
        }
```



```
    if(ae.getSource()==migenecomtab)
        {
            GeneCombineTableLymphoma rt=new
GeneCombineTableLymphoma();
            display(rt);
        }
        if(ae.getSource()==milymres)
        {
            ViewTable vv=new ViewTable("ResultLymphoma");
            display(vv);
        }
    }
void display(JInternalFrame obj)
        {
            new CenterFrame(obj);
            obj.setVisible(true);
            dp.add(obj);
            try
            {
```



```
                obj.setSelected(true);
            }
            catch(java.beans.PropertyVetoException e2)
            {
            }
        }
}
```


4. METHODOLOGY:

4.1. Data Preprocessing:

In all data sets, a small part of the data is missing. A k-nearest neighbor algorithm should be applied to fill those missing values. 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 Euclidean distance, though other distance measures, such as the Manhattan distance could in principle be used instead. The *k*-nearest neighbor algorithm is sensitive to the local structure of the data.

The accuracy of the *k*-NN algorithm can be severely degraded by the presence of noisy or irrelevant features, or if the feature scales are not consistent with their importance.

4.2. Gene Importance Ranking:

In this module, all genes in the training data set are ranked using a scoring scheme. Then, we retain the genes with high scores. The importance ranking of each gene is computed using a feature ranking measure. We use Class separability method for gene importance ranking.

The CS of gene i is defined as

$$CSi = SB_i/SW_i$$

where SB is the sum of squares of the interclass distances (the distances between samples of different classes). SWi is the sum of squares of the intraclass distances (the distances of samples within the same class).A larger CS indicates a greater ratio of the interclass distance to the intraclass distance and, therefore, can be used to measure the capability of genes to separate different classes.

4.3. Finding the Minimum Gene Subset:

After selecting some top genes in the importance ranking list, we attempt to classify the data set with only one gene. We input each selected gene into our classifier. If no good accuracy is obtained, we go on classifying the data set with all the possible 2-gene combinations within the selected genes. If still no good accuracy is obtained, we repeat this procedure with all of the 3-gene combinations and so on until we obtain a good accuracy..

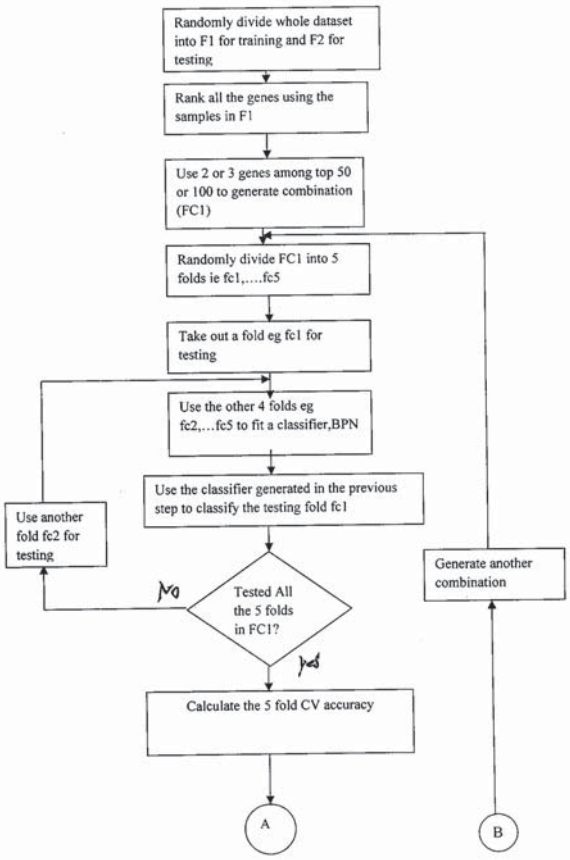
4.4. Applying BPN:

In this module, we use Backpropogation classifier to test gene combinations. BPN has the following steps to classify cancer datasets. we carry out 5-fold cross-validation (CV) in the training data set to tune their parameters. We have included CV accuracy for all of the data sets It includes the following steps:

1. Present a training sample to the neural network.
2. Compare the network's output to the desired output from that sample. Calculate the error in each output neuron.

3. For each neuron, calculate what the output should have been, and a scaling factor, how much lower or higher the output must be adjusted to match the desired output. This is the local error.
4. Adjust the weights of each neuron to lower the local error.
5. Assign "blame" for the local error to neurons at the previous level, giving greater responsibility to neurons connected by stronger weights.
6. Repeat the steps above on the neurons at the previous level, using each one's "blame" as its error.

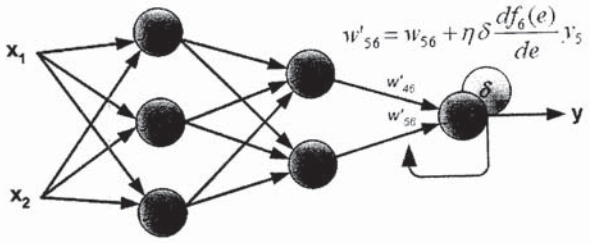
5. SYSTEM FLOW CHART:



represents derivative of neuron activation function (which weights are modified).

$$w'_{46} = w_{46} + \eta\delta\frac{df_6(e)}{de}.y_4$$

$$w'_{56} = w_{56} + \eta\delta\frac{df_6(e)}{de}.y_5$$



Coefficient affects network teaching speed. There are a few techniques to select this parameter. The first method is to start teaching process with large value of the parameter. While weights coefficients are being established the parameter is being decreased gradually. The second, more complicated, method starts teaching with small parameter value. During the teaching process the parameter is being increased when the teaching is advanced and then decreased again in the final stage. Starting teaching process with low parameter value enables to determine weights coefficients signs.

3. LITERATURE REVIEW:

When talking about cancer, specialists often distinguish only two forms: hereditary (or inherited) and sporadic cancer. In hereditary cancer, every body cell (somatic + germline) in the person's body has a mutation in some highly penetrant gene (e.g. BRCA1 or 2). Because we inherit two copies of genes (or alleles) on separate chromosomes from our parents, the second copy usually is normal. But people who have inherited a constitutional germline mutation in tumor suppressor genes are one step closer to cancer than those who haven't, because they generally follow the Knudson 'two-hit hypothesis" and need inactivation of a second gene copy for cancer to develop. So, these persons are "one step closer to cancer" and the term hereditary cancer is mostly applied when we know the responsible gene, which commonly is tumor suppressor and disease phenotype is transmitted in an autosomal dominant manner with partial penetrance (but on the genetic level you'll get recessive inheritance, cause disease is manifested when two copies of gene are faulty or so called loss of heterozygosity (LoH) is present). So, by the term of hereditary cancer we often mean "predisposition to cancer" (because we don't inherit cancer - we inherit only predisposition to it).

Mutations in the currently identified inherited cancer predisposition genes are relatively rare and probably play a major role in the development of about 5-10% of all solid tumors and a smaller proportion of hematological malignancies. This is not big but well defined and very important proportion.

In sporadic cancer there are mutations only in somatic cells (diploid) of affected organ. These mutations usually are not inherited and are caused by environment or other factors. The genetic testing in sporadic cancer cases usually gives prognostic information about the course of disease, but don't say anything about the recurrence in the future generations

3.1. Goals of classification of the tumours:

The classification of the tumours has several goals :

- · Predicting prognosis,
- · Adapting therapy to the clinical situation,
- · Comparing therapeutic results between relatively homogeneous groups of patients,
- · Enabling therapeutic studies which are essential in proving any therapeutic progress.

Classification enables the definition of therapeutic groups, for which therapeutic protocols can be elaborated, taking into account all treatment possibilities. It is essential for physicians to establish the classification of a tumour before any treatment can be administered to the patient in order to:

- · avoid proposing unnecessary treatment (for instance mutilating surgery when the patient unfortunately has metastases),
- · propose the most appropriate treatment (for instance: general treatment when localized treatment might be more appropriate).

3.2. Criteria used for tumour classifications:

Very early in the twentieth century, physicians who exchanged data about their therapeutic results felt the necessity to base their common classification on objective, easy to understand and easy to implement factors.

Most classifications are based on clinical data. However, other criteria are sometimes considered.

The most determining factors are :

- the degree of local invasion,
- the degree of remote invasion,
- histological types of cancer with specific grading for each type of cancer,
- possibly various tumour markers,
- (in the near future, gene markers and other proteomic abnormalities may become determining factors),
- general status of the patient.

In cases like these a vote of the 9 or 15 nearest neighbors would provide a better prediction accuracy for the system than would just the single nearest neighbor. Usually this is accomplished by simply taking the majority or plurality of predictions from the K nearest neighbors if the prediction column is a binary or categorical or taking the average value of the prediction column from the K nearest neighbors.

2.4. Neural Network:

When data mining algorithms are talked about these days most of the time people are talking about either decision trees or neural networks. Of the two neural networks have probably been of greater interest through the formative stages of data mining technology. As we will see neural networks do have disadvantages that can be limiting in their ease of use and ease of deployment, but they do also have some significant advantages. Foremost among these advantages is their highly accurate predictive models that can be applied across a large number of different types of problems.

To be more precise with the term "neural network" one might better speak of an "artificial neural network". True neural networks are biological systems (a k a brains) that detect patterns, make predictions and learn. The artificial ones are computer programs implementing sophisticated pattern detection and machine learning algorithms on a computer to build predictive models from large historical databases. Artificial neural networks derive their name from their historical development which started off with the premise that machines could be made to "think" if scientists found ways to mimic the structure and functioning of the human brain on the computer.

Thus historically neural networks grew out of the community of Artificial Intelligence rather than from the discipline of statistics. Despite the fact that scientists are still far from understanding the human brain let alone mimicking it, neural networks that run on computers can do some of the things that people can do.

It is difficult to say exactly when the first "neural network" on a computer was built. During World War II a seminal paper was published by McCulloch and Pitts which first outlined the idea that simple processing units (like the individual neurons in the human brain) could be connected together in large networks to create a system that could solve difficult problems and display behavior that was much more complex than the simple pieces that made it up. Since that time much progress has been made in finding ways to apply artificial neural networks to real world prediction problems and in improving the performance of the algorithm in general. In many respects the greatest breakthroughs in neural networks in recent years have been in their application to more mundane real world problems like customer response prediction or fraud detection rather than the loftier goals that were originally set out for the techniques such as overall human learning and computer speech and image understanding.

Where to Use Neural Networks:

Neural networks are used in a wide variety of applications. They have been used in all facets of business from detecting the fraudulent use of credit cards and credit risk prediction to increasing the hit rate of targeted mailings. They also have a long history of application in other areas such as

the military for the automated driving of an unmanned vehicle at 30 miles per hour on paved roads to biological simulations such as learning the correct pronunciation of English words from written text.

2.4.1. An Introduction to Back-Propagation Neural Networks :

Introduction :

This article focuses on a particular type of neural network model, known as a "feed-forward back-propagation network". This model is easy to understand, and can be easily implemented as a software simulation.

First we will discuss the basic concepts behind this type of NN, then we'll get into some of the more practical application ideas.

Complex Problems:

The field of neural networks can be thought of as being related to artificial intelligence, machine learning, parallel processing, statistics, and other fields. The attraction of neural networks is that they are best suited to solving the problems that are the most difficult to solve by traditional computational methods.

Consider an image processing task such as recognizing an everyday object projected against a background of other objects. This is a task that even a small child's brain can solve in a few tenths of a second. But building a conventional serial machine to perform as well is incredibly complex. However, that same child might NOT be capable of calculating 2+2=4, while the serial machine solves it in a few nanoseconds.

A fundamental difference between the image recognition problem and the addition problem is that the former is best solved in a parallel fashion, while simple mathematics is best done serially. Neurobiologists believe that the brain is similar to a massively parallel analog computer, containing about 10^10 simple processors which each require a few milliseconds to respond to input. With neural network technology, we can use parallel processing methods to solve some real-world problems where it is very difficult to define a conventional algorithm.

2.4.2. The Feed-Forward Neural Network Model:

If we consider the human brain to be the 'ultimate' neural network, then ideally we would like to build a device which imitates the brain's functions. However, because of limits in our technology, we must settle for a much simpler design. The obvious approach is to design a small electronic device which has a transfer function similar to a biological neuron, and then connect each neuron to many other neurons, using RLC networks to imitate the dendrites, axons, and synapses. This type of electronic model is still rather complex to implement, and we may have difficulty 'teaching' the network to do anything useful. Further constraints are needed to make the design more manageable. First, we change the connectivity between the neurons so that they are in distinct layers, such that each neruon in one layer is connected to every neuron in the next layer. Further, we define that signals flow only in one direction across the network, and we simplify the neuron and synapse design to behave as analog comparators being driven by the other neurons through simple resistors. We now have a feed-forward neural network model that may actually be practical to build and use.

Referring to figures 1 and 2, the network functions as follows: Each neuron receives a signal from the neurons in the previous layer, and each of those signals is multiplied by a separate weight value. The weighted inputs are summed, and passed through a limiting function which scales the output to a fixed range of values. The output of the limiter is then broadcast to all of the neurons in the next layer. So, to use the network to solve a problem, we apply the input values to the inputs of the first layer, allow the signals to propagate through the network, and read the output values.

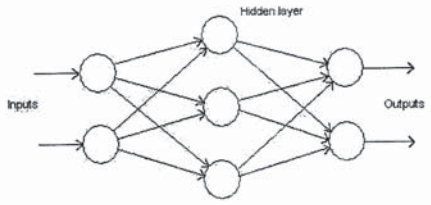


Figure 1. A Generalized Network. Stimulation is applied to the inputs of the first layer, and signals propagate through the middle (hidden) layer(s) to the output layer. Each link between neurons has a unique weighting value.

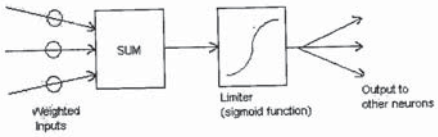


Figure 2. The Structure of a Neuron. Inputs from one or more previous neurons are individually weighted, then summed. The result is non-linearly scaled between 0 and +1, and the output value is passed on to the neurons in the next layer.

Since the real uniqueness or 'intelligence' of the network exists in the values of the weights between neurons, we need a method of adjusting the weights to solve a particular problem. For this type of network, the most common learning algorithm is called Back Propagation (BP). A BP network learns by example, that is, we must provide a learning set that consists of some input examples and the known-correct output for each case. So, we use these input-output examples to show the network what type of behavior is expected, and the BP algorithm allows the network to adapt.

The BP learning process works in small iterative steps: one of the example cases is applied to the network, and the network produces some output based on the current state of it's synaptic weights (initially, the output will be random). This output is compared to the known-good output, and a mean-squared error signal is calculated. The error value is then propagated backwards through the network, and small changes are made to the weights in each layer. The weight changes are calculated to reduce the error signal

for the case in question. The whole process is repeated for each of the example cases, then back to the first case again, and so on. The cycle is repeated until the overall error value drops below some pre-determined threshold. At this point we say that the network has learned the problem "well enough" - the network will never exactly learn the ideal function, but rather it will asymptotically approach the ideal function.

Here are some situations where a BP NN might be a good idea:

- A large amount of input/output data is available, but you're not sure how to relate it to the output.
- The problem appears to have overwhelming complexity, but there is clearly a solution.
- It is easy to create a number of examples of the correct behavior.
- The solution to the problem may change over time, within the bounds of the given input and output parameters (i.e., today 2+2=4, but in the future we may find that 2+2=3.8).
- Outputs can be "fuzzy", or non-numeric.

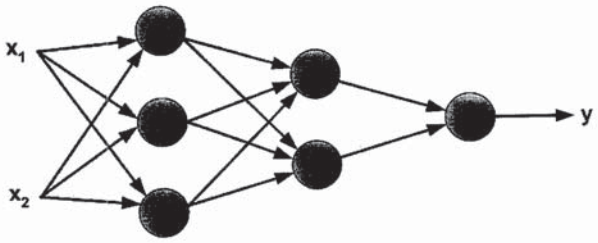
One of the most common applications of NNs is in image processing. Some examples would be: identifying hand-written characters; matching a photograph of a person's face with a different photo in a database; performing data compression on an image with minimal loss of content. Other applications could be: voice recognition; RADAR signature analysis; stock market prediction. All of these problems involve large amounts of data, and complex relationships between the different parameters.

It is important to remember that with a NN solution, you do not have to understand the solution at all! This is a major advantage of NN approaches. With more traditional techniques, you must understand the inputs, and the algorithms, and the outputs in great detail, to have any hope of implementing something that works. With a NN, you simply show it: "this is the correct output, given this input". With an adequate amount of training, the network will mimic the function that you are demonstrating. Further, with a NN, it is OK to apply some inputs that turn out to be irrelevant to the solution - during the training process, the network will learn to ignore any inputs that don't contribute to the output. Conversely, if you leave out some critical inputs, then you will find out because the network will fail to converge on a solution.

If your goal is stock market prediction, you don't need to know anything about economics, you only need to acquire the input and output data (most of which can be found in the Wall Street Journal).

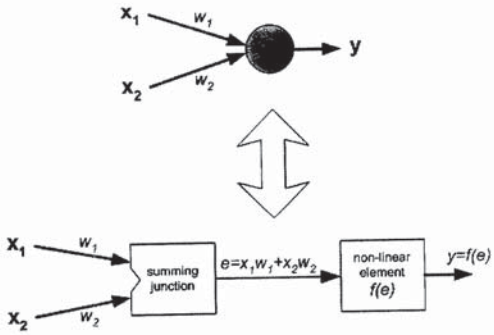
2.4.3. Principles of training multi-layer neural network using backpropagation algorithm :

The project describes teaching process of multi-layer neural network employing backpropagation algorithm. To illustrate this process the three layer neural network with two inputs and one output, which is shown in the picture below, is used:

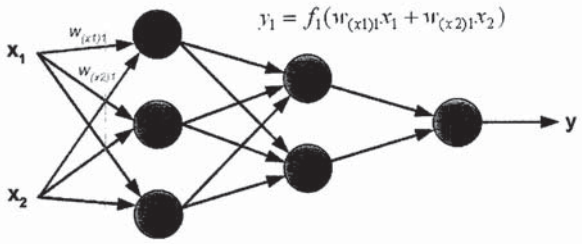


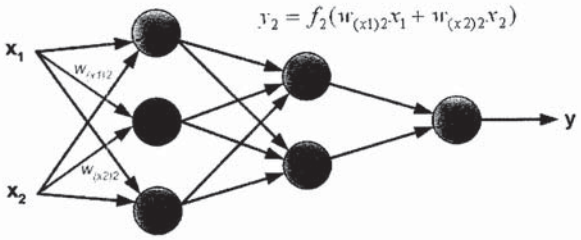
Process of multi-layer neural network

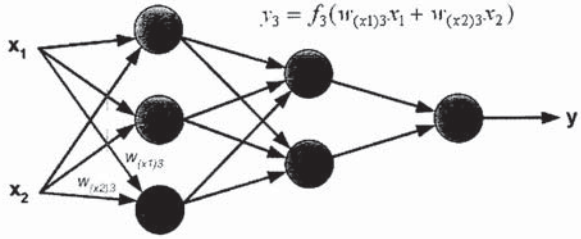
Each neuron is composed of two units. First unit adds products of weights coefficients and input signals. The second unit realise nonlinear function, called neuron activation function. Signal e is adder output signal, and y = f(e) is output signal of nonlinear element. Signal y is also output signal of neuron.



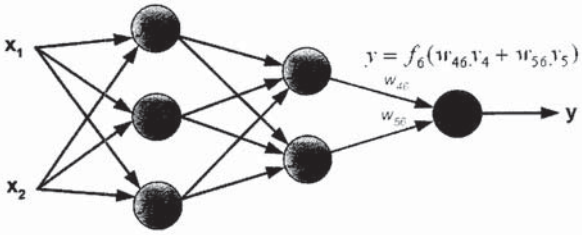
To teach the neural network we need training data set. The training data set consists of input signals (x_1 and x_2) assigned with corresponding target (desired output) z. The network training is an iterative process. In each iteration weights coefficients of nodes are modified using new data from training data set. Modification is calculated using algorithm described below: Each teaching step starts with forcing both input signals from training set. After this stage we can determine output signals values for each neuron in each network layer. Pictures below illustrate how signal is propagating through the network, Symbols $w_{(xm)n}$ represent weights of connections between network input x_m and neuron n in input layer. Symbols y_n represents output signal of neuron n.



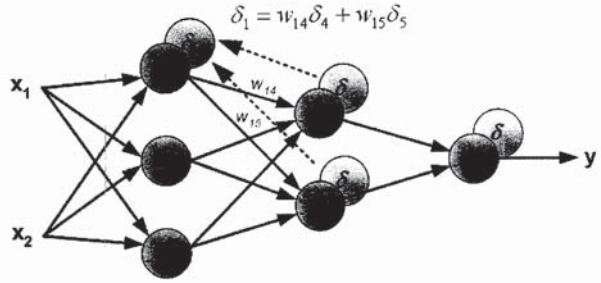




Propagation of signals through the hidden layer. Symbols w_{mn} represent weights of connections between output of neuron m and input of neuron n in the next layer.



The weights' coefficients w_{mn} used to propagate errors back are equal to this used during computing output value. Only the direction of data flow is changed (signals are propagated from output to inputs one after the other). This technique is used for all network layers. If propagated errors came from few neurons they are added. The illustration is below:



When the error signal for each neuron is computed, the weights coefficients of each neuron input node may be modified. In formulas below df(e)/de