CHAPTER 5

DISCUSSION

5.1 Introduction

Computer based support in insurance sector is becoming more significant because of fierce and growing competition. Use of data mining technology with upcoming intelligent techniques is playing a crucial role for an insurance organization. The aim of this dissertation is to develop neural network based prediction models for life insurance sector that can be beneficial for the development of an intelligent decision support system for insurance industry.

In this research, we have developed neural network based prediction models to predict the insurance data which can be beneficial for growth of the organization and policy making decisions. The thesis reveals that it is possible to quickly build the models and train the models with a number of first and second order training algorithms. The developed models and algorithm enable the researchers to employ these models and technique in the development of intelligent decision support systems. On the other hand, the scope of the research is not only limited to insurance domain but it is also applicable to development of decision support systems in other domains or application areas.

In this thesis, we have also proposed and developed a new gradient based algorithm that has the capability to achieve more efficient prediction of datasets and which depicts a faster convergence toward point of minimum error. Also we have compared the novel technique with existing standard gradient based algorithms of first and second order. The study focuses on the development of appropriate neural network configurations based upon MLP architecture, using the supervised learning method along with new and existing gradient based training algorithms.

The present study investigates the predictive performance and accuracy of the novel algorithm for the insurance datasets with help of a number of statistical parameters and plots. The aim of this dissertation was to present an efficient adaptive technique which avoids the second order computations of complex hessian matrix of second
order derivatives. Since the traditional techniques have failed to cope up with large datasets and are not capable to handle the non-linear relationships that are present in real word situations. To represent and explore these non-linear relationships with older statistical techniques is either a very complex task or not possible at all. This study is set out to assess the impact of ANN learning capabilities to develop the prediction models which can prove to be a useful asset for the development of intelligent decision support systems. We also discussed and compared their performance with other predictive methods.

This is the first study, to our knowledge, to examine the predictive capabilities of ANN in life insurance sector. Thereresults achieved during the researchdescribe for the first time the predictive capabilities of a novel training algorithm based on the normalized adaptive technique. Other studies in this area to our knowledge have examined only the direct application of existing first and second order algorithms in a wide range of application areas. But here we have presented a novel training algorithm for neural networks and also tested its performance and compared with existing popular algorithms.

5.2 Comparison of Training Algorithms Employed for the Development of Perfection Models with the Proposed Algorithm

As shown in the precious chapter through result tables and performance plots that second order algorithms and the proposed algorithm based prediction models are showing very good performance and predictive accuracy. The ANN models produced overall results that were very good for forecasting, but with poor results for a few models based on first order and traditional methods. The models were evaluated using the performance (MSE) parameters, final gradient values, regression plots, error histograms and confusion matrix.

Comparative charts shown below in figures from 5.1 to 5.4 present a comparative investigation for the various employed training methods.
We can clearly observe from the above chart for comparison of performance values (MSE) that second order methods have shown the best performance values like CGM reached 0.039, SCGM reached 0.0375 and Levenberg Marquardet method has shown the best performance of 0.0138. On the other hand first order methods and adaptive methods have shown poor performance values like simple gradient descent (GD) reached till 0.0581, gradient descent with adaptive momentum (GDM) reached till 0.0584.

Only gradient descent with adaptive learning (GDA) method could achieve a slightly better performance value of 0.0464 but finally proves to be a poor method because it could not converge toward the final set target even in 1000 epochs and final gradient value is 0.0816 as shown in the next chart. Finally the proposed method could reach a performance value of 0.0499 and was able to achieve the set target of minimum gradient in 209 epochs and the final error gradient value is 9.95E-03 and is satisfactory. We can say from the above results that performance of proposed normalized adaptive method falls in between the second order methods and first order methods.
While comparing the final gradient values for the employed methods, we can clearly observe from the above chart for final error gradient values that second order methods were able to achieve the gradient value of the order of $10^{-5}$ but first order and traditional adaptive methods could not do that even 1000 epoch. On the other hand, proposed method achieved the final gradient of the order of $10^{-3}$.
Epoch and time graph clearly show that Levenberg Marquardet method has shown the fastest convergence in 71 epochs and taken minimum time and first order methods were not able to converge even in 1000 epochs. The proposed algorithm proves to be very good and converged toward set target in 209 epochs while taking 0:03:41 minutes.

5.3 Limitations of the Research Work

Neural network are becoming popular for their powerful learning and non-linear mapping capabilities and is a rapidly growing technology. We have successfully applied artificial neural networks to develop the prediction models based on MLP networks along with a number of training algorithms. But still there are a number of limitations of the research work which are described as follows:

- **Global Convergence:** We have applied the first order and second order gradient based methods which are prone to get entangled in small local minima present on the multi-dimensional surface and are less suitable for global optimization. Therefore to achieve more accurate prediction model we need to train the model
again and again with different initial parameters. On the other hand, other intelligent techniques like genetic algorithms, Evolutionary algorithms, simulated annealing etc.offer a global search solution in a large multi-dimensional search space.

• **GUI Development:** We have developed the prediction models using the procedures and GUI available in MATLAB toolbox. Development of an independent GUI is possible for applying the methods under consideration. However it was not required since the available GUIs were sufficient to meet the research objectives.

• **IDSS Implementation:** We have suggested data mining techniques for prediction modeling and proposed the training algorithms, which are crucial parts of development of an intelligent support system (IDSS). But actual development of IDSS software has not been done. However it is out of the scope of the research work and whenever required can be achieved in any kind of application area and the algorithms suggested can be applied in its development and decision making.

• **Black Box Nature of Working:** Processing of large weight vectors, large datasets and thus large number of epochs and iterations, cause the Black Box nature working for ANN which makes it difficult to interpret what is happening inside. The complex computations involved during training phase of the prediction model by learning algorithms obstruct us from clear understanding how a decision is being concluded by the neural networks. Hence, due to their black box nature of working, we have to blindly accept the output of the network, which, however, does not endear the neural network technology to users.

• **Other Neural Network Architectures:** We have tested the research only on MLP architecture with a number of configuration and a wide variety of training algorithms. MLP networks are found to be very much suitable for prediction modeling, but other architectures of neural networks and variations of the traditional techniques can also be tested for the feasibility of prediction modeling.

• **Unstable Nature of Problem Solving:** ANNs are also criticized because of the unstable nature of problem solving and the often inability to repeat the process
and obtain the same results. Because the data are constantly changing so it becomes difficult to repeat the solution to a problem.

5.4 Other Areas of Application

Neural networks have been successfully applied for the solution of a variety of problems. Neural networks have shown remarkable progress in the recognition of visual images, handwritten and printed characters, speech recognition, face detection, finger prints detection etc. which come under the domain of pattern recognition. ANNs also have been applied to solve problems in manufacturing domain like scheduling, defect recognition etc. and for the solution the problems related to control systems and branch of robotics. As seen during the current research ANNs exhibit the capability to predict situations from past trends. They have found ample applications in areas like finance, stock market, banking, insurance, econometrics with high success rates. In the light of changing business environments managers are feeling the need for more flexible forecasting and decision making models. More specifically it is becoming increasingly important for forecasting models today to be able to detect non-linear relationships while allowing for high levels of noisy data and chaotic components. ANNs fit well in these kind of situations and therefore highly acceptable. ANNs are very much suitable for the development of expert systems and problems related to medical domain like early diagnosis of a number of health problems. They are also suitable for problem in a variety of miscellaneous domains like development of intrusion detection system (IDS), software reliability estimation, calamity prediction systems, and a number of other important intelligent systems. Since ANN think and behave like humans so the concept can be developed and applied in new application areas where humans have no reach.

In the following section we have tested and discussed the application of proposed algorithm on two other application areas with their different datasets – for heart disease prediction and software estimation.

5.4.1 Testing the Algorithms for Other Datasets

The algorithms under consideration are not only applied for insurance datasets. But their performance has also been tested for other datasets as described in the following
sections. The algorithms have been successfully tested for heart disease dataset, software estimation datasets and to judge their learning behavior and performance.

5.4.1.1 Testing the Algorithm for Heart Disease Dataset

Datasets for heart disease prediction have been extracted from ‘UCI Machine Learning Repository’, a publically available datasets repository for heart disease diagnosis in the text format. This is available online at ‘http://mlr.cs.umass.edu/ml/support/Heart+Disease’. We have selected 8 important input predictor variables to predict 1 output variable for classifying the state of heart disease. Best results after applying the training algorithms and the proposed algorithm have been shown in table 5.1.

<table>
<thead>
<tr>
<th>Training Method Employed</th>
<th>Hidden Layer Neurons</th>
<th>Minimum Gradient</th>
<th>Final Epochs</th>
<th>Training Performance</th>
<th>Initial Gradient</th>
<th>Final Gradient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steepest (Gradient) Descent</td>
<td>30</td>
<td>0.001</td>
<td>7</td>
<td>6.48</td>
<td>12.7</td>
<td>1.65e+03</td>
</tr>
<tr>
<td>Conjugate Gradient</td>
<td>30</td>
<td>0.001</td>
<td>145</td>
<td>0.547</td>
<td>6.45</td>
<td>0.167</td>
</tr>
<tr>
<td>Scaled Conjugate Gradient</td>
<td>30</td>
<td>0.001</td>
<td>105</td>
<td>0.534</td>
<td>6.36</td>
<td>0.240</td>
</tr>
<tr>
<td>Levenberg Marquardet</td>
<td>30</td>
<td>0.001</td>
<td>459</td>
<td>0.000554</td>
<td>12.7</td>
<td>.000252</td>
</tr>
<tr>
<td>Normalized Adaptive</td>
<td>30</td>
<td>0.001</td>
<td>54</td>
<td>1.21</td>
<td>13.7</td>
<td>0.687</td>
</tr>
</tbody>
</table>

Figures from 5.5 to 5.10 show the snapshots for training process, performance graphs, error gradient graphs, validation fail graph and histogram plot for the proposed method applied on heart disease dataset. Validation plot shows the stopping criterion due to validation fail for validation data during the training process. The training process stops if validation fail reaches a value of 6 (default value). Performance of the proposed algorithm is found good and in between first and second order methods.
Fig. 5.5 Neural network training process snapshot for heart disease dataset

Fig. 5.6 Neural network training process snapshot for heart disease dataset
Fig. 5.7 Training performance graph with proposed normalized adaptive learning algorithm for heart disease dataset.

Best Validation Performance is 1.2386 at epoch 48

Fig. 5.8 Error gradient graph with proposed normalized adaptive learning algorithm for heart disease dataset.

Gradient = 0.68726, at epoch 54

Fig. 5.9 Validation fail graph with proposed normalized adaptive learning algorithm for heart disease dataset.

Validation Checks = 6, at epoch 54
Fig. 5.10 Histogram plot with proposed normalized adaptive learning algorithm for heart disease dataset

Note: [Figures 5.5 to 5.10 have been plotted in MATLAB Neural Network Toolbox R2012a, V.7.14.0.739.]

5.4.1.2 Testing the Algorithm for Software Reliability Estimation Dataset

Datasets for software reliability estimation have been extracted from ‘PROMISE Software Engineering Repository’, a publically available datasets repository for software project development for 93 different projects. This is available online at ‘http://promise.site.uottawa.ca/SERepository’. We have selected 3 important input predictor variables to predict 1 output variable for estimating software reliability in different classes. Best results after applying the training algorithms and the proposed algorithm have been shown in Table 5.2. Performance of the proposed algorithm is found good and in between first and second order methods.

Table 5.2 Optimal results when employing different training methods for software reliability estimation dataset

<table>
<thead>
<tr>
<th>Training Algorithm Employed</th>
<th>Gradient Descent</th>
<th>Levenberg Marquardt</th>
<th>Conjugate Gradient Descent</th>
<th>Proposed Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of epoch used for validation performance</td>
<td>Val. Failed in epoch 0</td>
<td>116</td>
<td>102</td>
<td>104</td>
</tr>
<tr>
<td>Neurons Used</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>Gradient Value</td>
<td>3.23e+16</td>
<td>8.38e-13</td>
<td>0.419</td>
<td>11.3</td>
</tr>
<tr>
<td>Performance Value</td>
<td>8.44e+03</td>
<td>9.49e-06</td>
<td>.00684</td>
<td>1.48</td>
</tr>
</tbody>
</table>
Figures from 5.11 and 5.12 show performance graphs and error gradient graphs, for the proposed method and applied on software estimation dataset.

### 5.5 Improving the Accuracy of Results

If the network is not sufficiently accurate, then we can try improving the results in following ways:

- We can try initializing the network and then train again. Each time we initialize a feed-forward network; the network parameters are different and might produce...
different solutions. We can make use of ‘init’ and ‘train’ function as shown below.

\[ net = \text{init}(net); \]

\[ net = \text{train}(net, \text{Inputs, Targets}); \]

- As a second approach, we can try increase the number of hidden neurons. Larger the numbers of neurons in the hidden layer give the network more flexibility because the network has more parameters and it can optimize the situation in a better way. However, we should increase the layer size gradually because if we make the hidden layer too large, we might cause the problem of overfitting. Overfitting occurs when a model becomes excessively complex, such as having too many network parameters relative to the number of observations. A model that has been overfit will show poor predictive performance, as it overreacts to minor fluctuations in the training data.

- A third option is to try a different training function. We have applied some selected training functions of first and second order but there exist different training functions which can sometimes produce better results.

- Finally, we can try using additional training data. Providing additional data for the network is more likely to produce a network that generalizes well to new data.

But in our case there is no need to try for the above mentioned approaches because we are already getting sufficiently good results for our proposed method and the results have been discussed in the previous chapter in detail.