# Part 1: Classification Problem

The classification problem is one of the most studied problems in the field of Computer Science with applications in many other fields. It is a problem of identifying which categories of objects the new observation belongs to.

Of the many interesting applications is the classification of email into SPAM and NOT SPAM emails. This is a problem whose application in the real world is clearly identified with it saving countless number of hours for the people every single day. Every email service provided uses classification techniques to separate the emails of their clients.

Of the many classification techniques used is the one with the implementation of classification with the help of a trained neural network model. Neural network models have been successful in separating the handwritten digits for the MNIST dataset.

### Training Data

The training data for training the neural network model has been provided in the P\_train file of the dataset and its target output has been mapped to the T\_train data vector from dataset.

The data provided in the above file is unprocessed and processing needs to be done to convert it into mean normalize the input feature vector. Before feeding the data into the neural network for training and validation the training data is split into two portions for validation and training. The ratio of the data split for each of the individual model has been mentioned in this report.

The neural network model is then trained and the optimum model is decided using the best validation model.

### Test Data

The test data for testing the accuracy of the neural network is provided in the P\_Test file of test dataset and its target output mapping is provided in the T\_Test.

The test data provided is used to check the accuracy of the trained neural network model which is expected to be close to the expected output for the unseen data which has not been used for training.

### Data Pre-processing

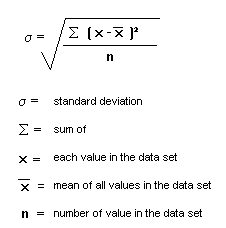
**Training data** is to be pre-processed normalizing the data such that the mean of the input feature vector is 0 and the standard deviation of the data is 1. This will ensure that each of the individual feature in the input feature vector falls in this normalised region.

**Test data** is to be pre-processed using the mean and standard deviation derived after pre-processing the training data. The testing data is not used in the pre-processing stage for the calculation of mean and standard deviation.

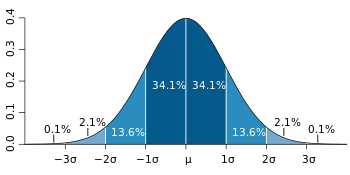
The mean of the feature in an individual feature vector is calculated using this formula,



The standard deviation of the data is calculated using the following formula,



After mean normalizing the data, each of the input feature in the feature vector will fall in region which is represented by the following figure,



## Performance Metric

For the classification problem, out task is to minimize the error of misclassification of the data.

## Training function used

### Lavenberg-Marquardt Algorithm

The LMA is an iterative algorithm used for solving the minimization of the non-linear least squares problem which is used for curve fitting in a multi-dimensional data space. The LMA is first initialized with a starting value then converges to the local minimum. As such LMA reaches different convergence points for different values of initialization. As such, only when this initialization point is closer to the Global Minima, we can obtain the best model for the given dataset.

As such for the purposes of training, the trainlm function in MATLAB initializes the guess parameter vector to the following value,



For each iteration of the learning, the parameter β is updated using the gradient of the cost function using the approximation of the improved function using the linearization,

Firstly, the value of the following function is calculated,



The output change is then approximated with the corresponding change in β using the above equation and the gradient of this change is calculated,

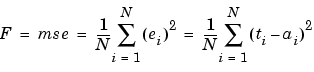
J_i=\frac{\partial f(x_i,\boldsymbol\beta)}{\partial \boldsymbol\beta}

Once the algorithm reaches the optimal value of **β** (vector) then the gradient will become 0 and the learning stops.

## Bayesian Regularization

Regularization is done to improve generalization of the training input data. The regularization is achieved by modifying the performance function of the trained neural network in an iterative process.

The performance function of training a feedforward function can be defined by the following equation,

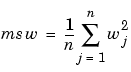


Generalization of this performance function is improved by adding a term that consists of the sum squares of the network weights and biases.

The regularized performance function can be represented by the following equation,

http://radio.feld.cvut.cz/matlab/toolbox/nnet/backp25a.gif

The parameter ϒ is the representation of performance ratio. The sum of weights is represented by the weights equation



# Neural Networks implemented for Classification

## Configuration Settings

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 10
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

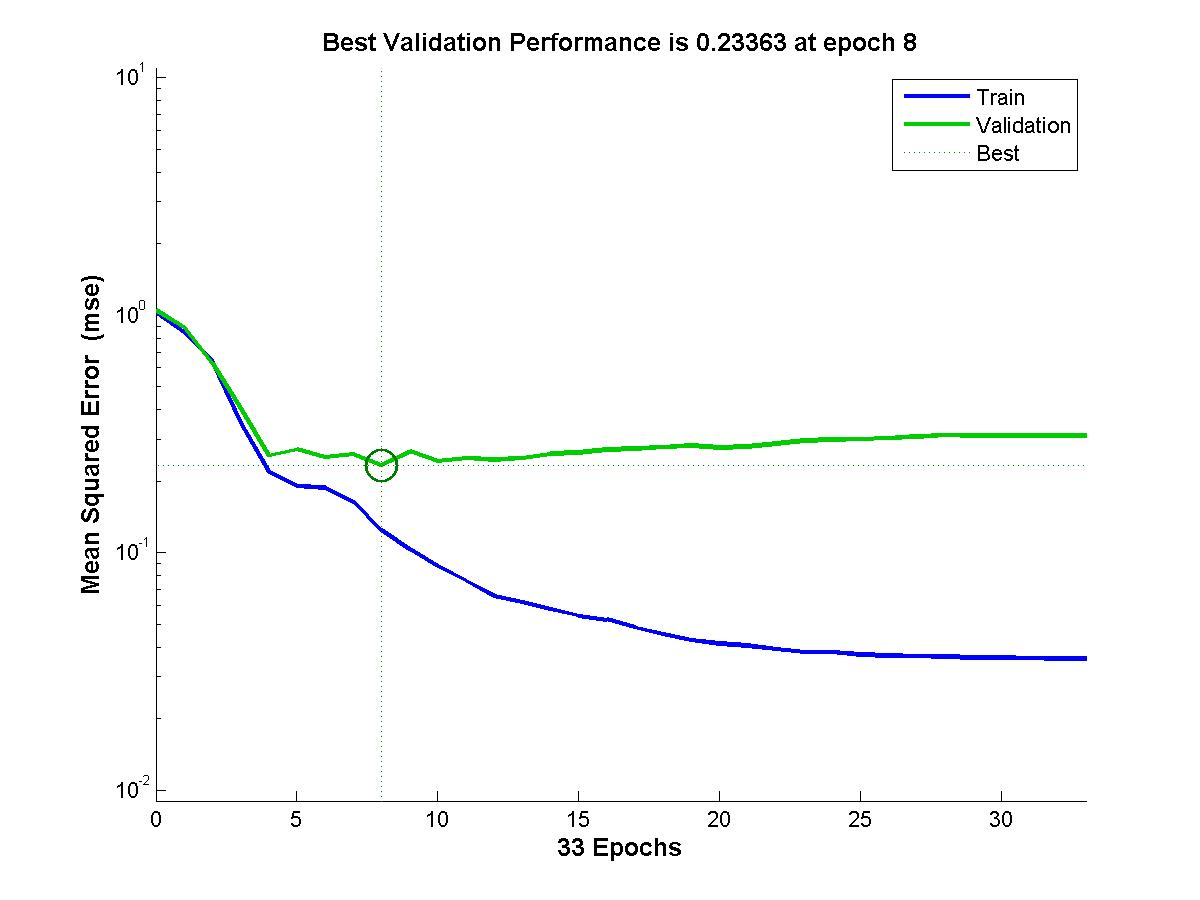


Figure 1: Training and Validation Error vs Number of Epochs

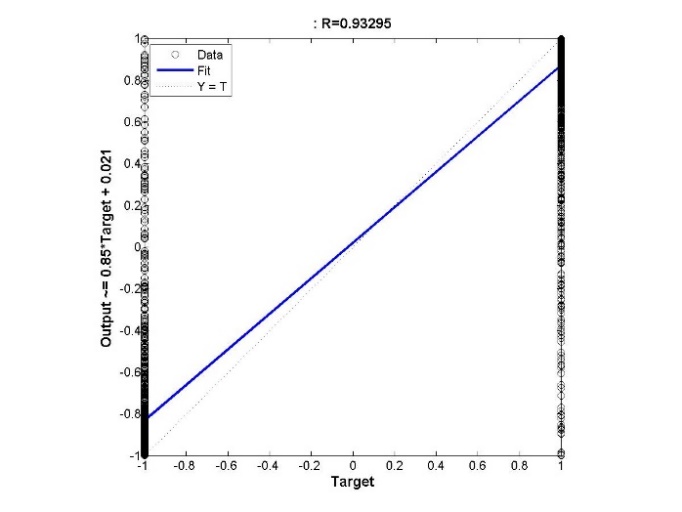
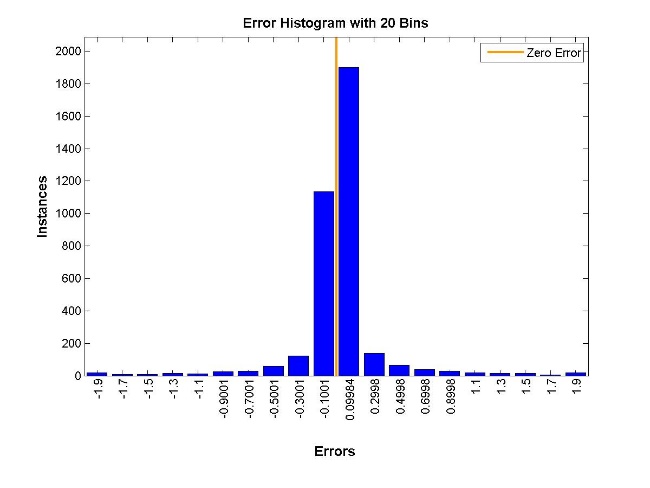
 

Figure 2: (a) Regression figure (b) Histogram for Neural Network with 1 hidden layer with 10 neurons

**Misclassification Error = 0.0771739130434783**

## Configuration Settings

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 46
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

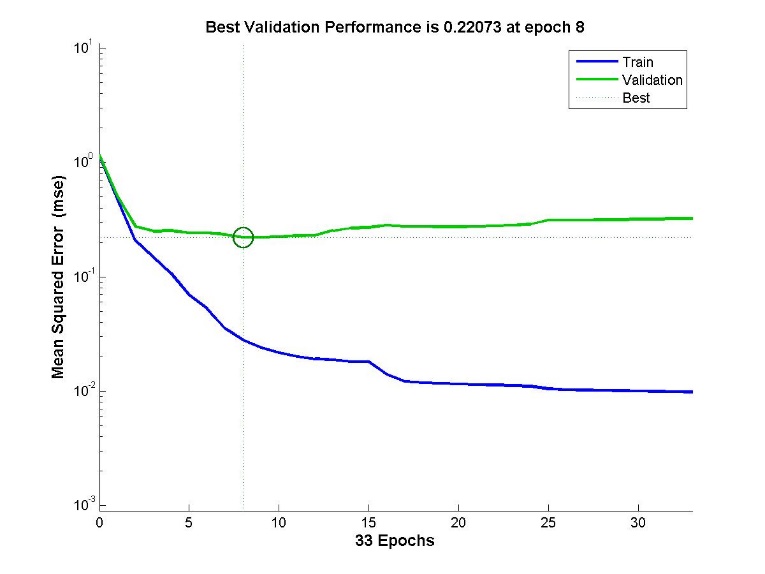
****

Figure 3: Training and Validation Error vs No. of Epochs for Neural Network with 1 hidden layer with 46 neurons

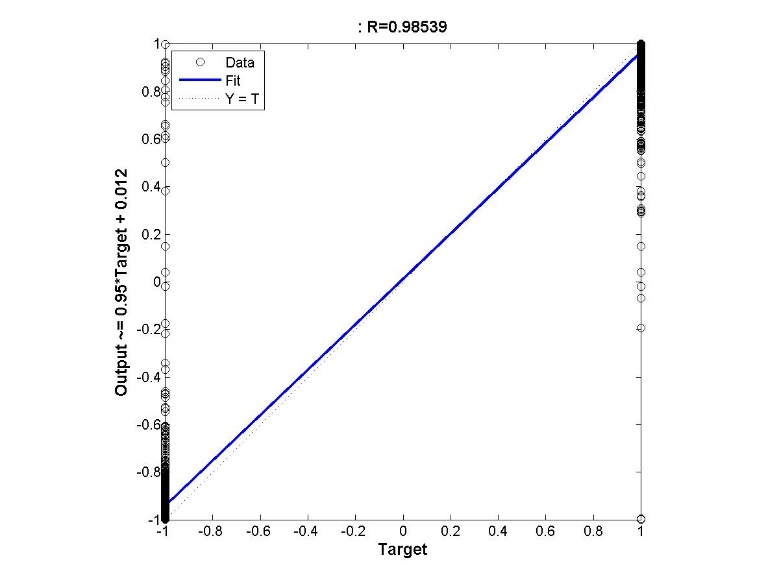
****

Figure 4: (a) Regression figure (b) Histogram for Neural Network with 1 hidden layer with 46 neurons

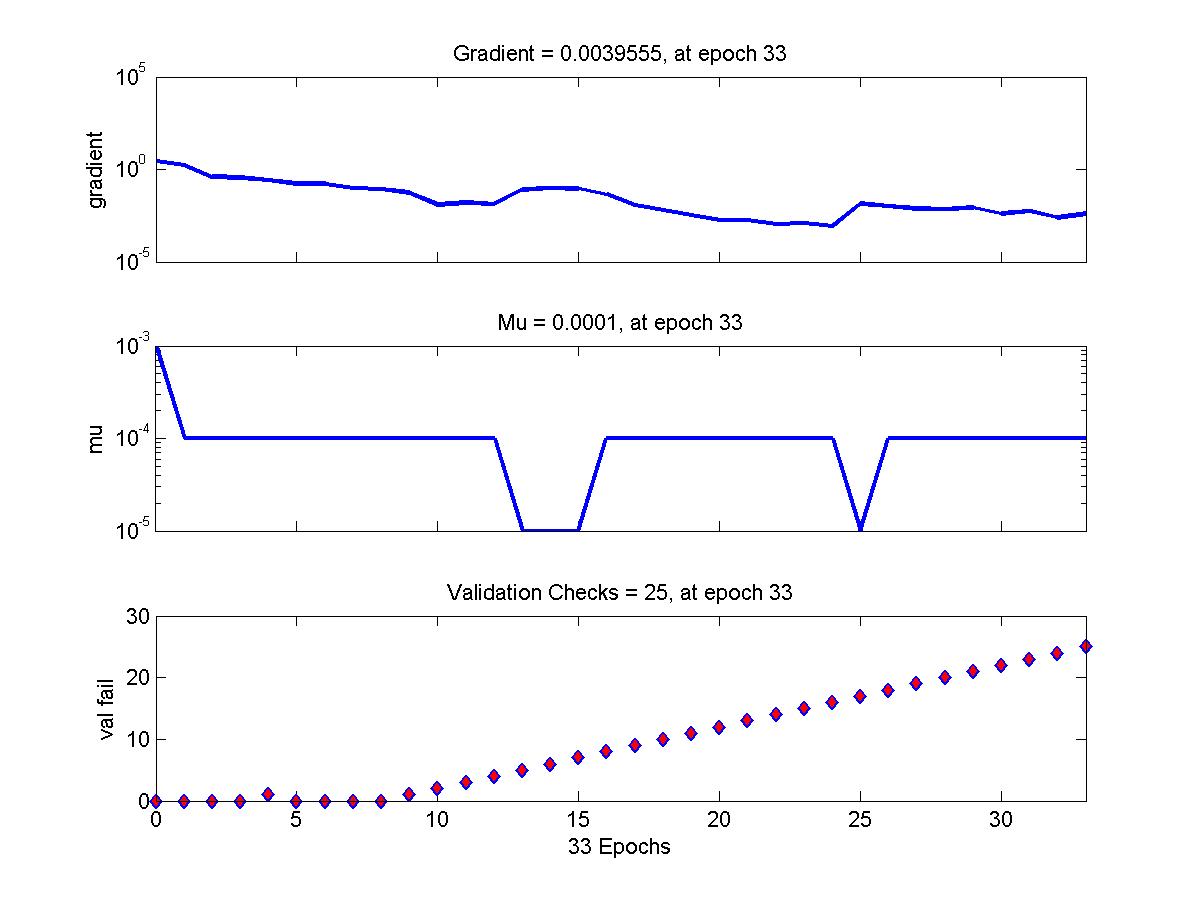
****

Figure 5: Gradient of the Function trained with Neural Network with 1 hidden layer of 46 neurons

**Misclassification error = 0.063043478260870**

## Configuration Settings

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 56
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

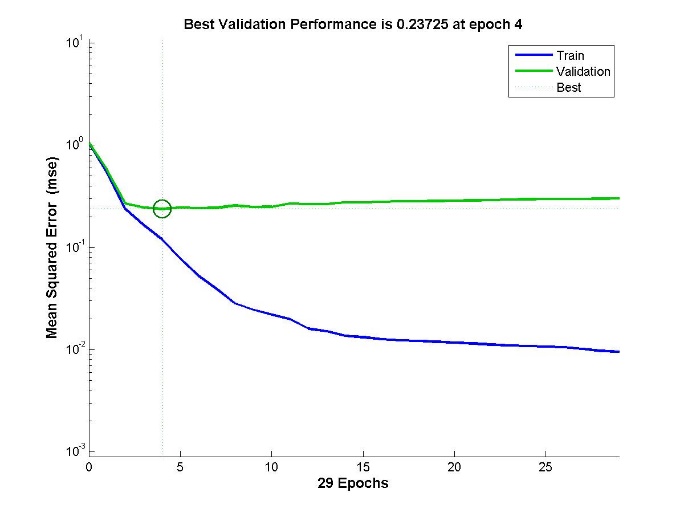


Figure 6: Training and Validation Error vs No. of Epochs for Neural Network with 1 hidden layer with 56 neurons

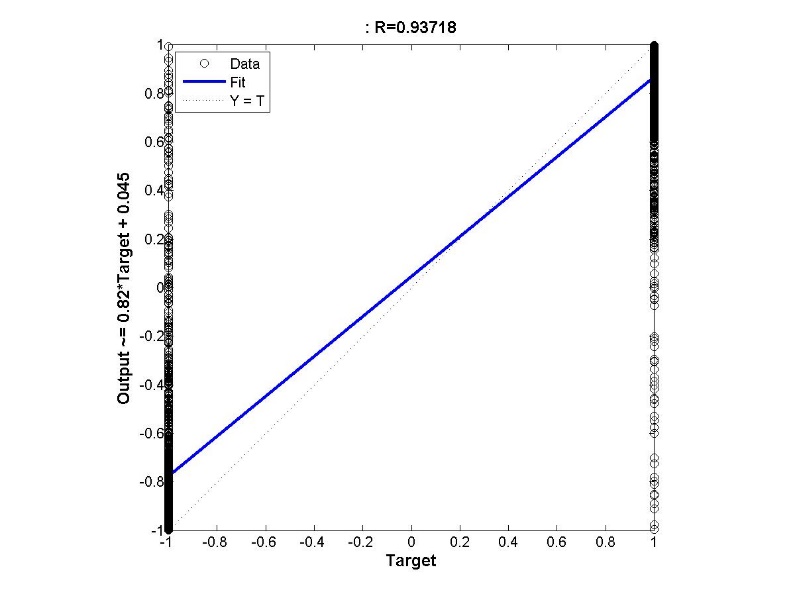
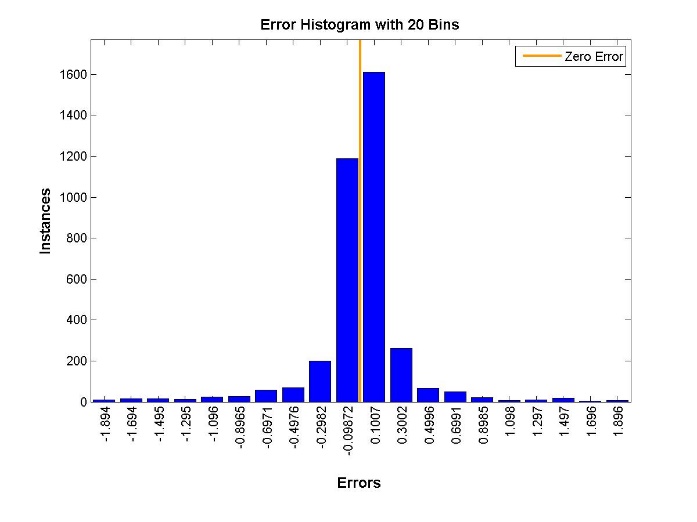


Figure 7: (a) Regression figure (b) Histogram for Neural Network with 1 hidden layer with 56 neurons

**Misclassification Error = 0.0804347826086957**

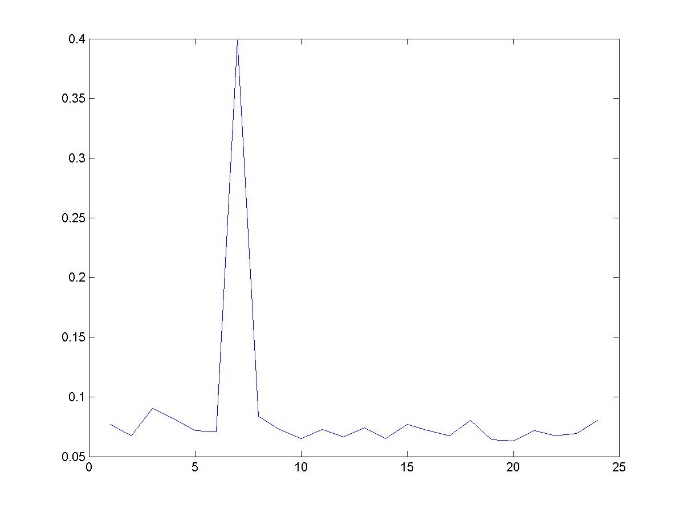
****

Figure 8:

# Changing the Data Split Ration between Training and Validation Data

## Configuration Settings

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 10
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 60:40

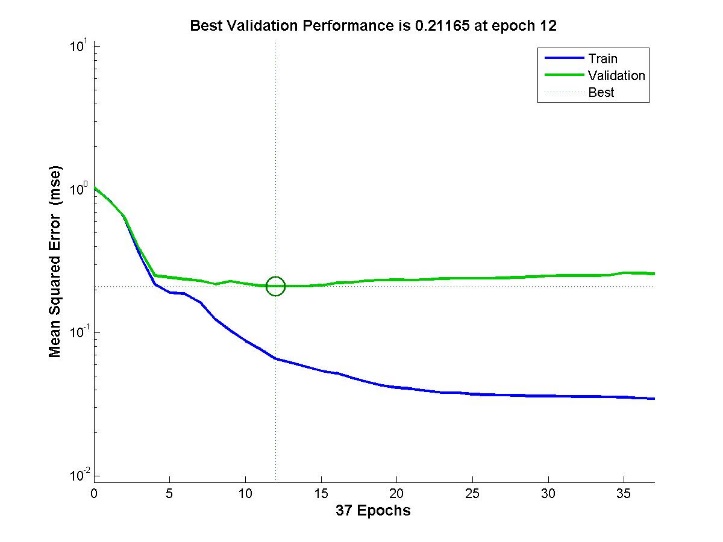
****

Figure 9: Training and Validation Error vs No. of Epochs for Neural Network with 1 hidden layer with 10 neurons with data split ratio of 60:40

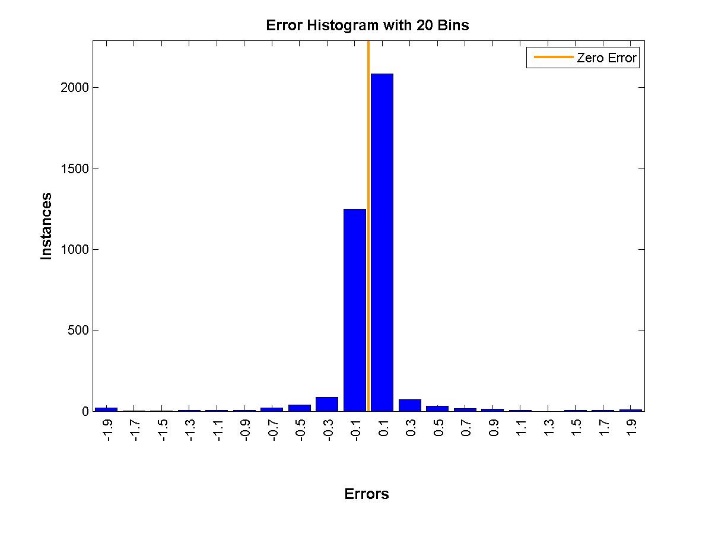
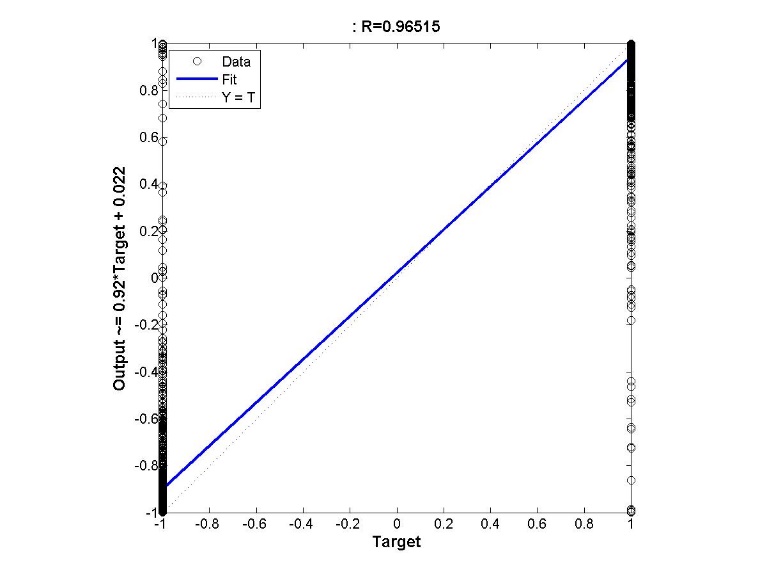
****

Figure 10: (a) Regression figure (b) Histogram for Neural Network with 1 hidden layer with 10 neurons with data split ration of 60:40 for the Training: Validation

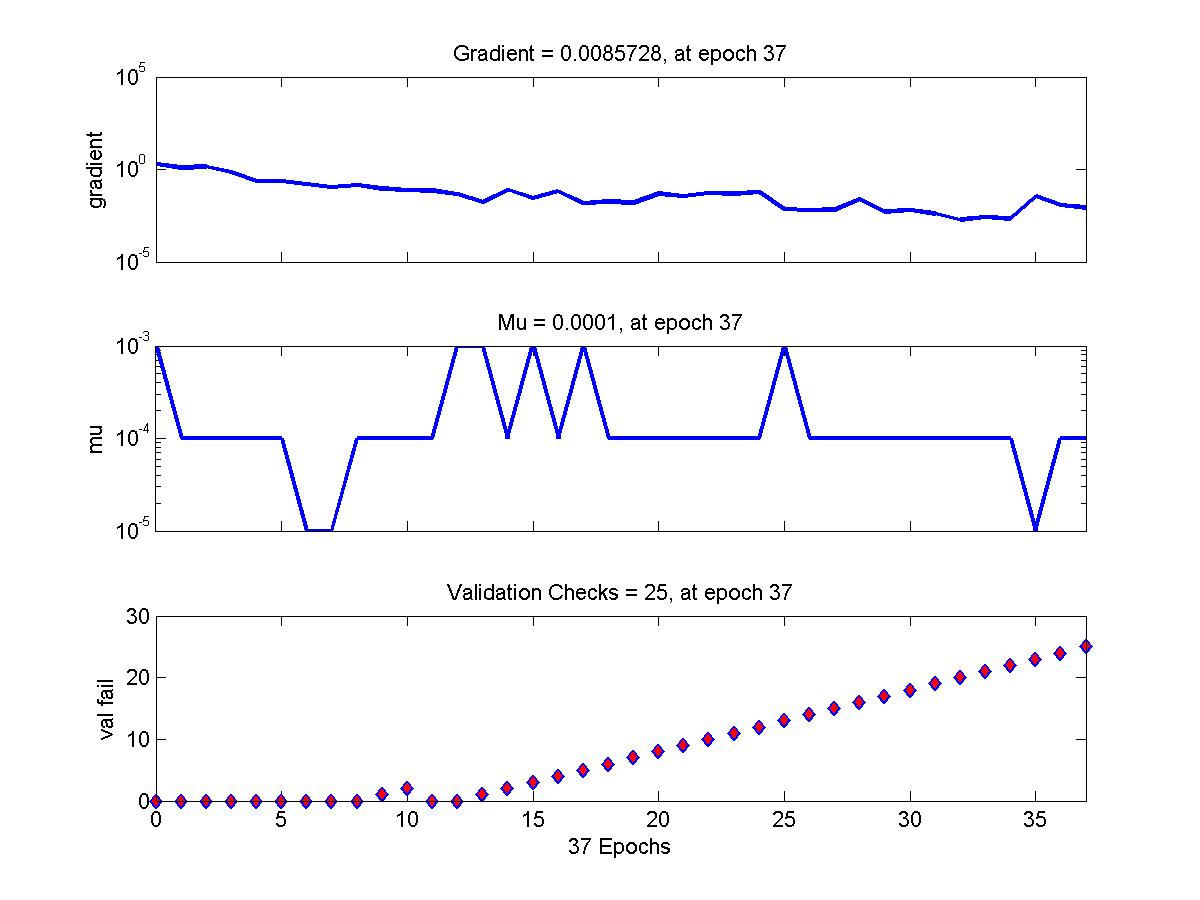
****

Figure 11: Gradient of the Function trained with Neural Network with 1 hidden layer of 10 neurons data split ration of 60:40 for the Training: Validation

## Varying Maximum Validation

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 10
3. Max\_fail = 10
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

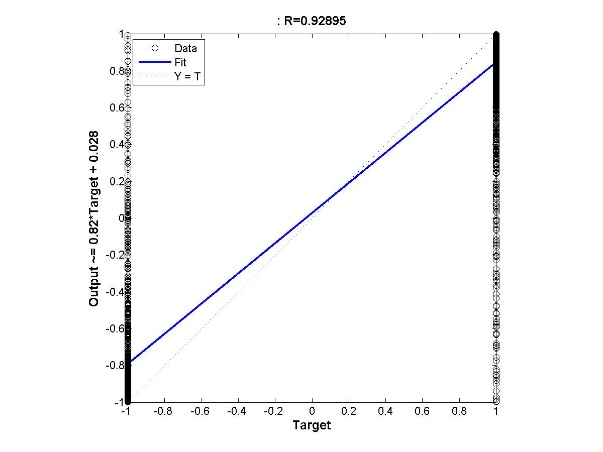
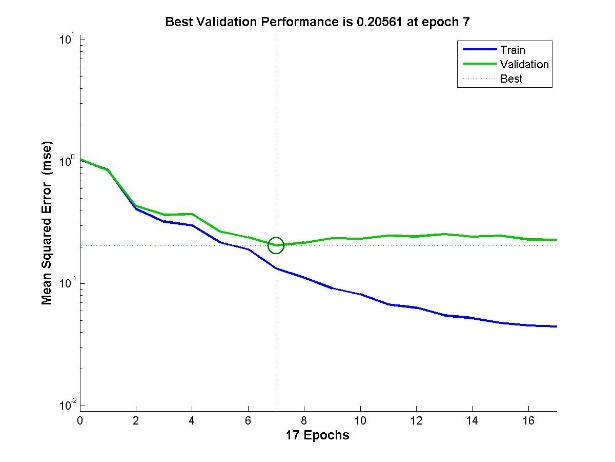


Figure 12: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 1 layer and Maximum Number of continuous Validation increase failures is 10

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 10
3. Max\_fail = 30
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

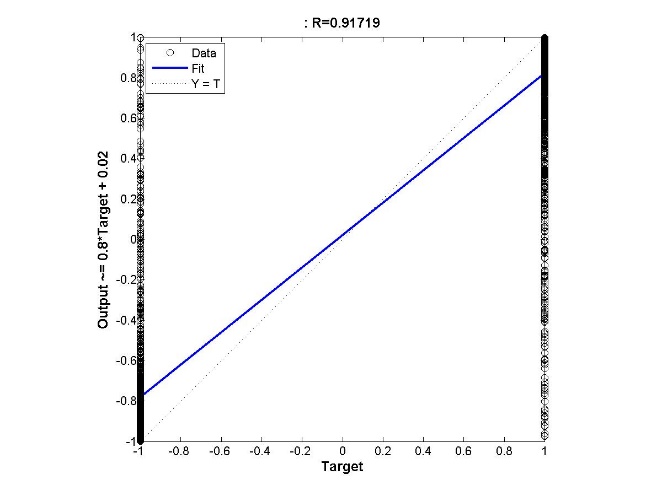
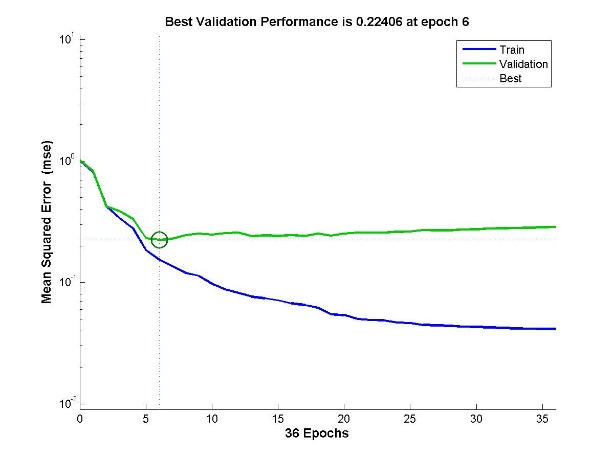


Figure 13: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 1 layer and Maximum Number of continuous Validation increase failures is 30

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 10
3. Max\_fail = 70
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

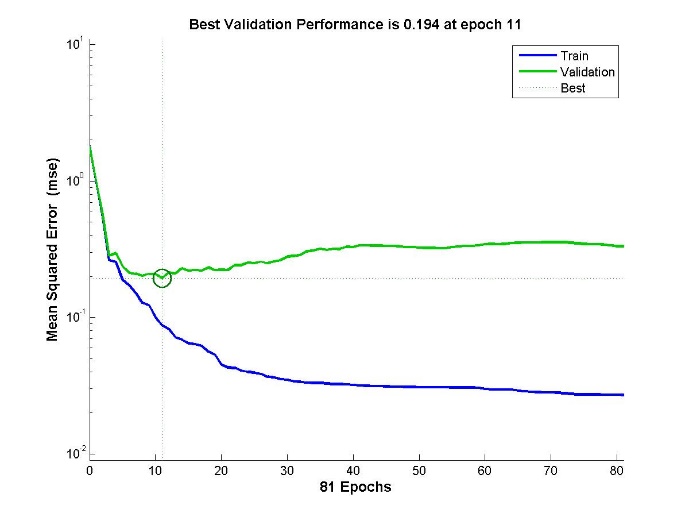


Figure 14: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 1 layer and Maximum Number of continuous Validation increase failures is 70

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 10
3. Max\_fail = 110
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

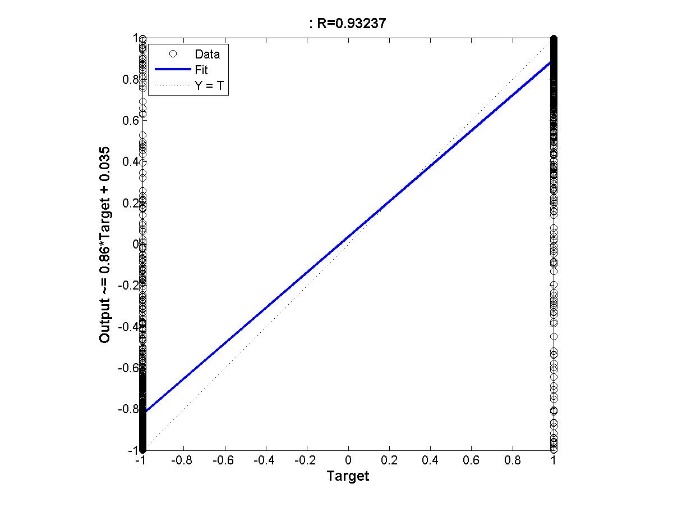
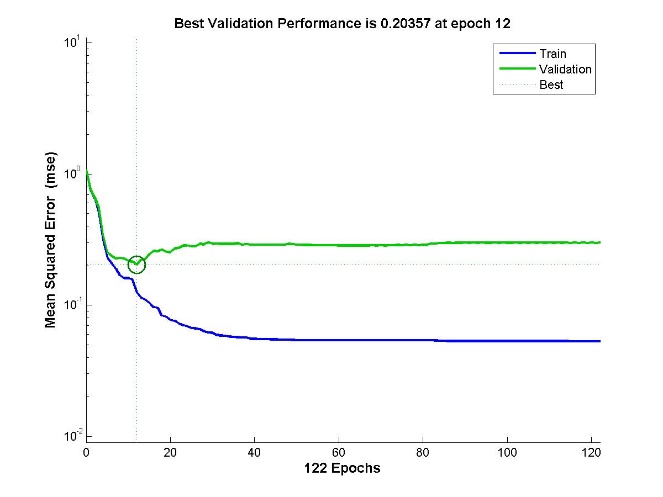


Figure 15: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 1 layer and Maximum Number of continuous Validation increase failures is 110

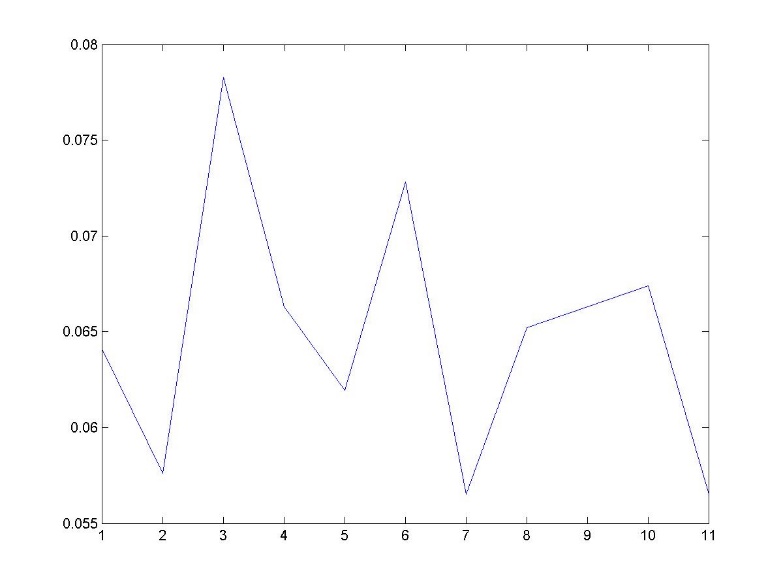


Figure 16: Testing Error vs Max\_fail/10 for the Neural Network with 1 hidden layer of 10 neurons

## Varying the number of hidden layers and the number of neurons in each layer

**Number of Layers = 2**

1. Number of hidden layers = 2
2. Number of neurons in the hidden layer = [20 20]
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

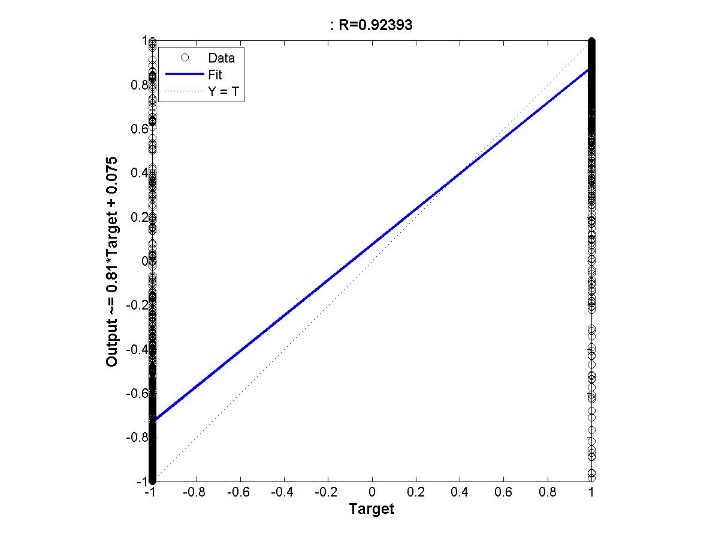
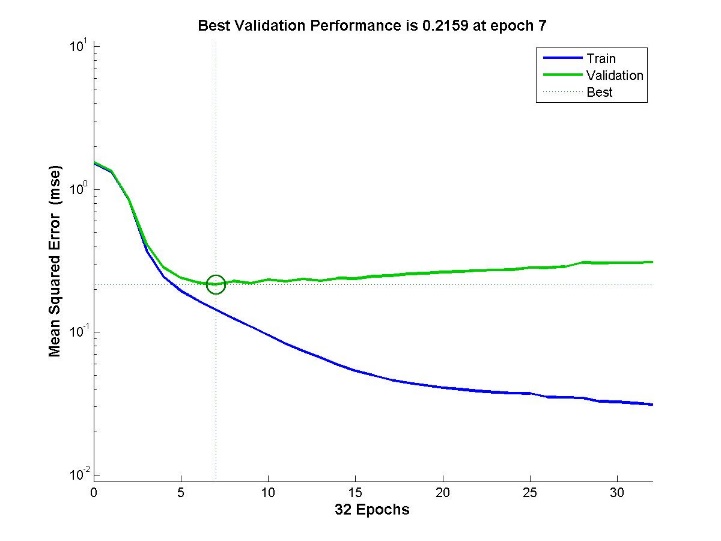
****

Figure 17: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 2 layer with hidden configuration of [20 20]

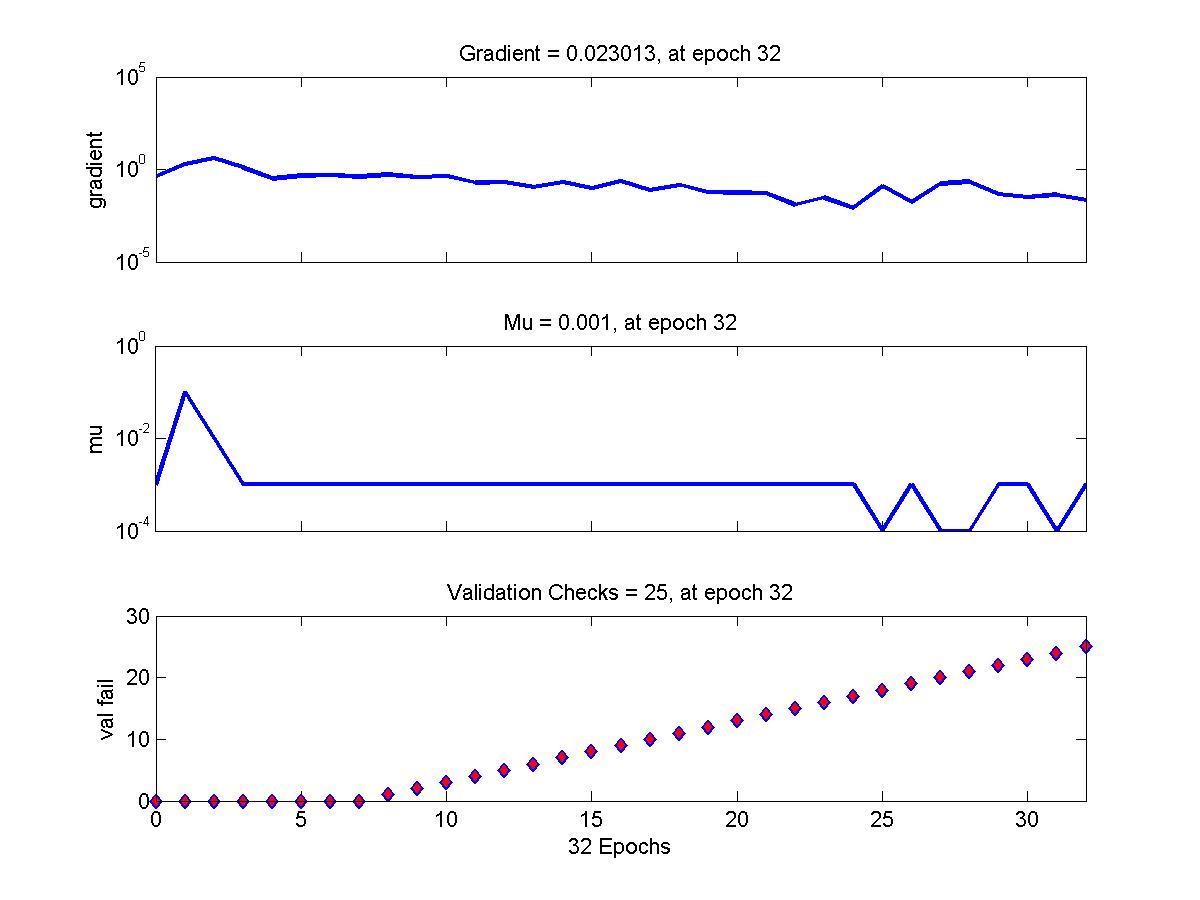
****

Figure 18: Training State of a neural network with 2 layer with hidden configuration of [20 20]

1. Number of hidden layers = 2
2. Number of neurons in the hidden layer = [30 30]
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

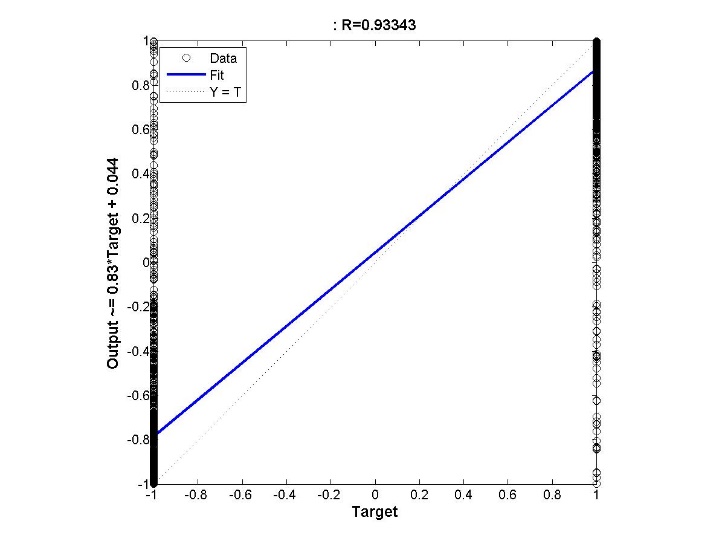
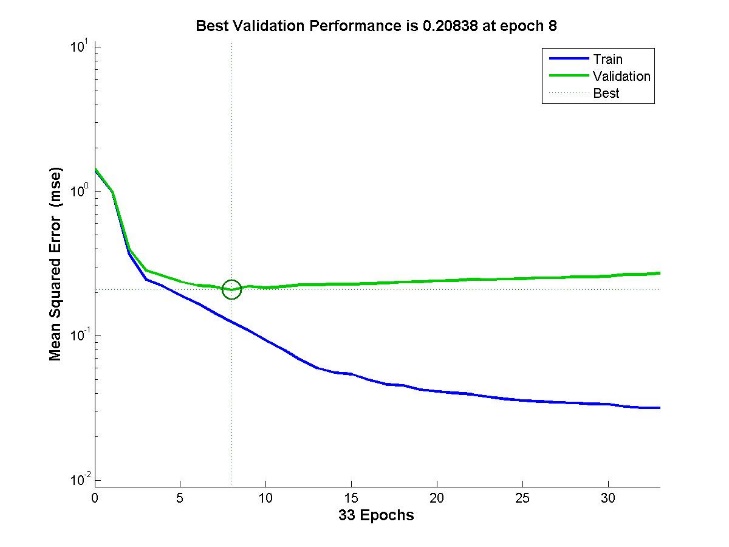


Figure 19: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 2 layer with hidden configuration of [30 30]

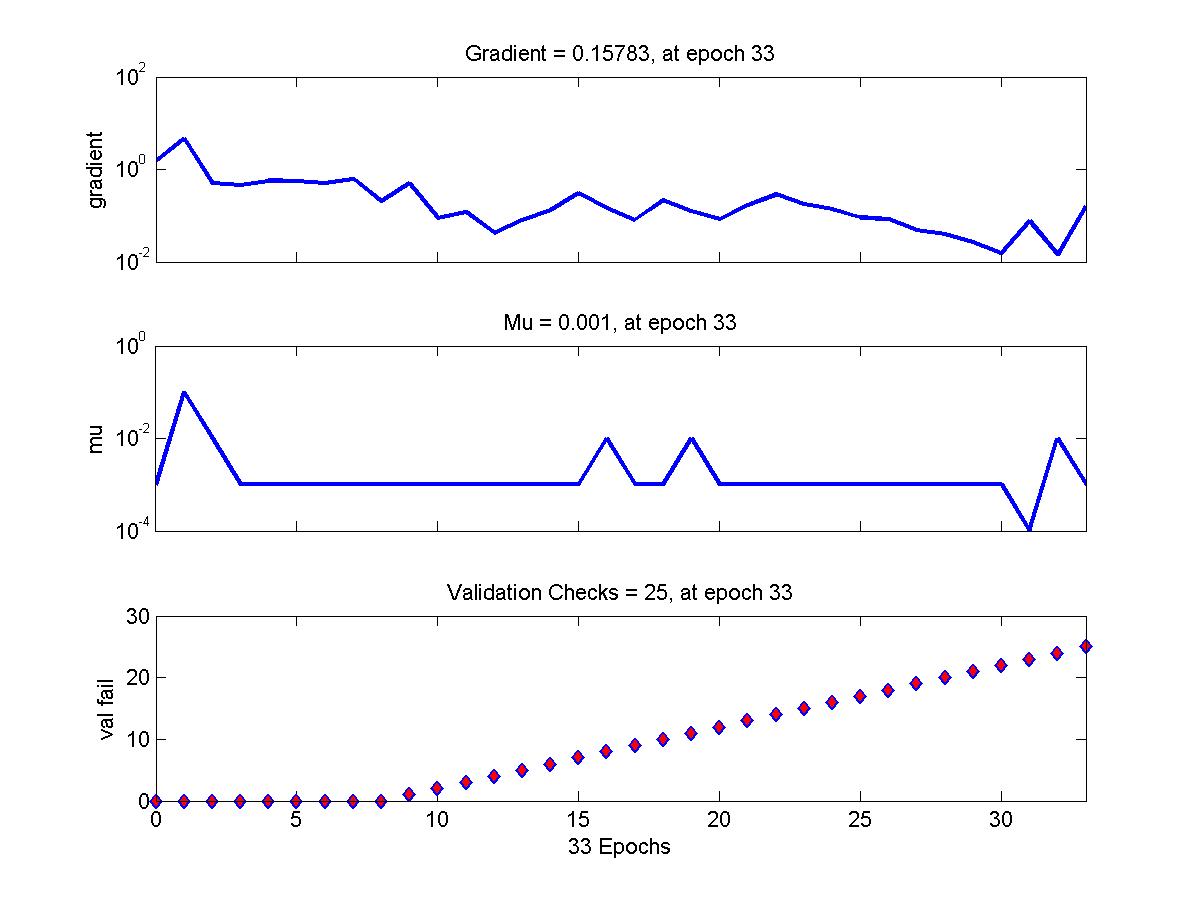


Figure 20: Training State of a neural network with 2 layer with hidden configuration of [30 30]

**Misclassification rate for**

**Number of neurons in second layer = 10**

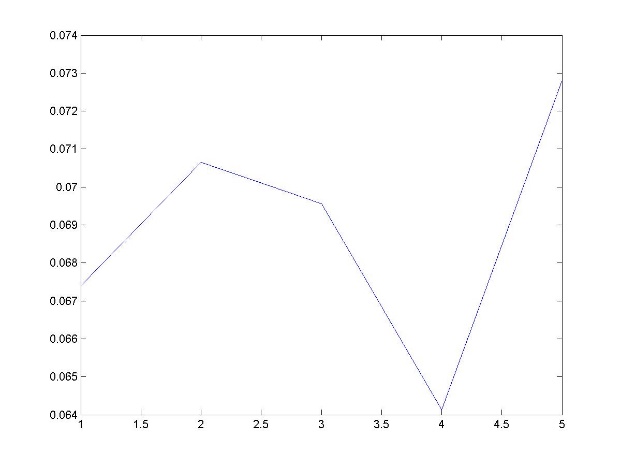
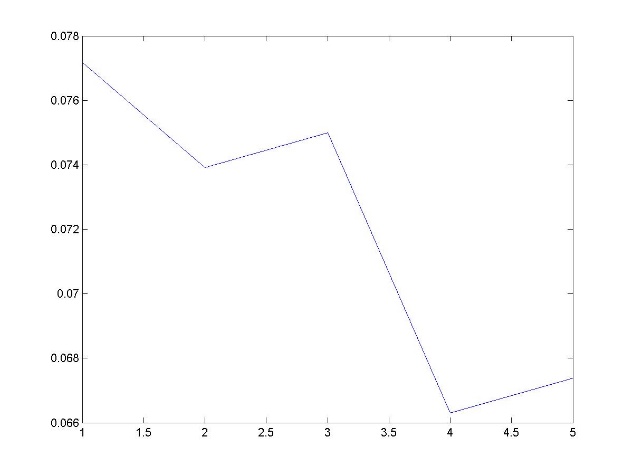
****

Figure 21: Testing Error (Misclassification Rate) vs Number of neurons in the first layer (a) when the number of neurons in the second layer is fixed at 10 (b) when the number of neurons in the second layer is fixed at 20

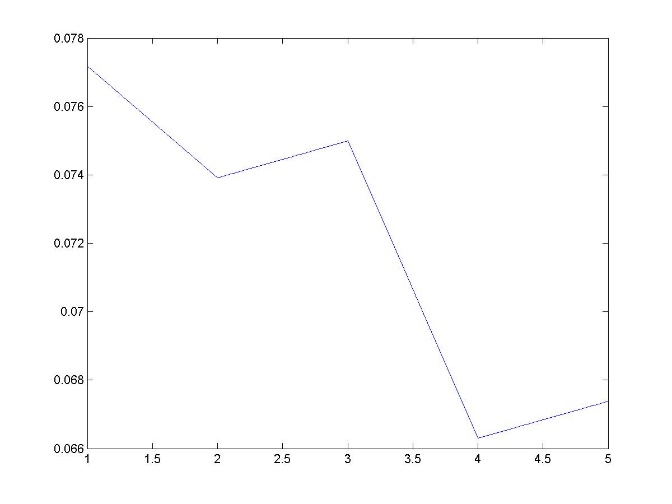
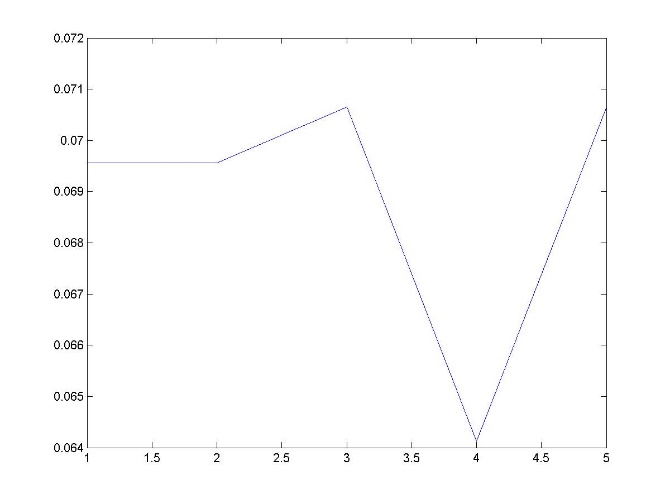
****

Figure 22: Testing Error (Misclassification Rate) vs Number of neurons in the first layer (a) when the number of neurons in the second layer is fixed at 30 (b) when the number of neurons in the second layer is fixed at 40

## Bayesian Regularization Training

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = [10]
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

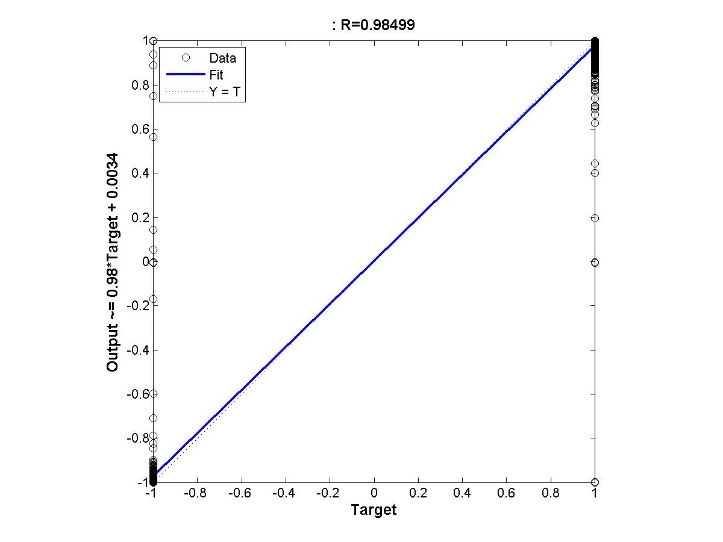
****

Figure 23: Training Error vs No. of Epochs for Neural Network with 1 hidden layer with 10 neurons and Bayesian Regularization training

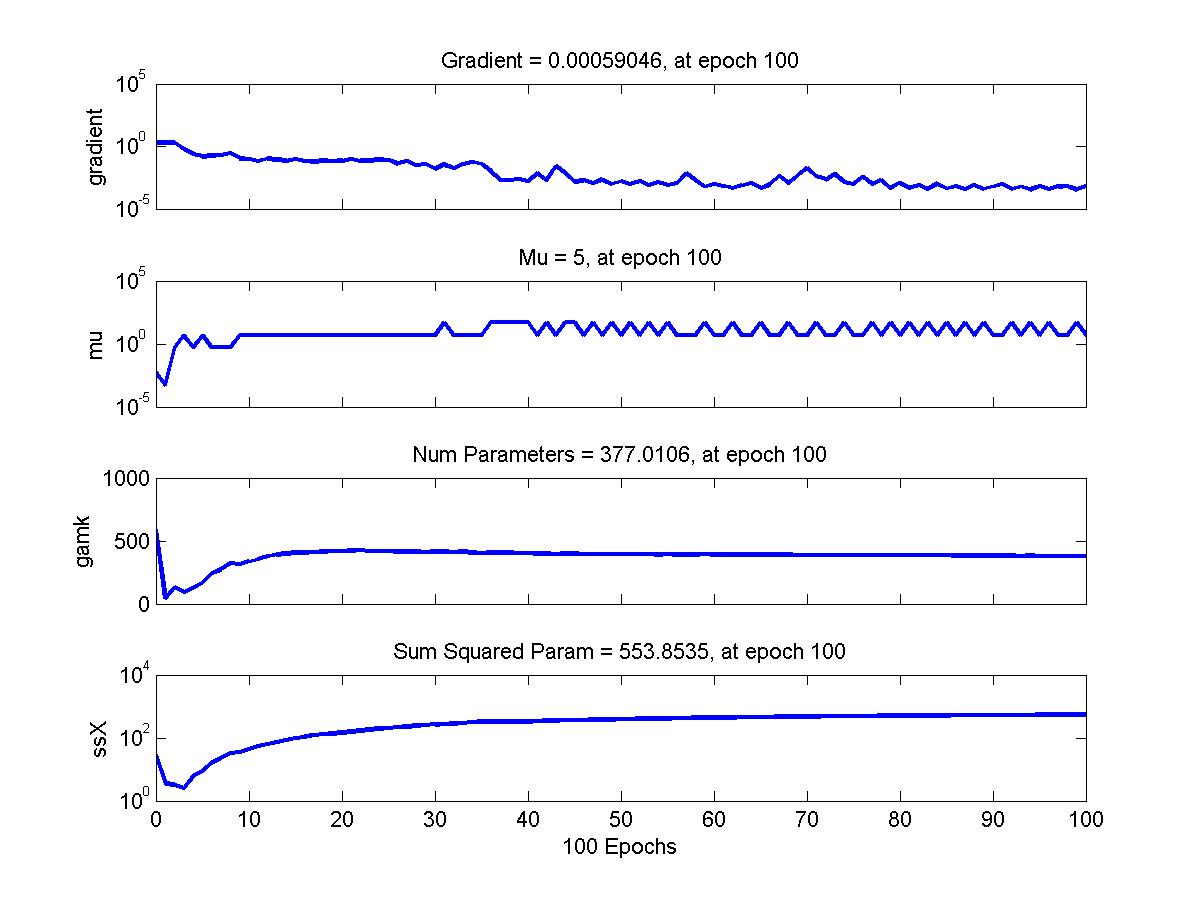
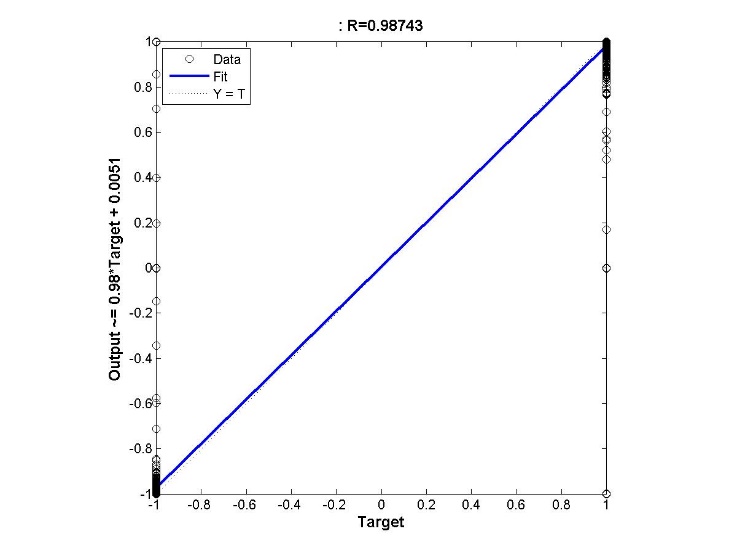
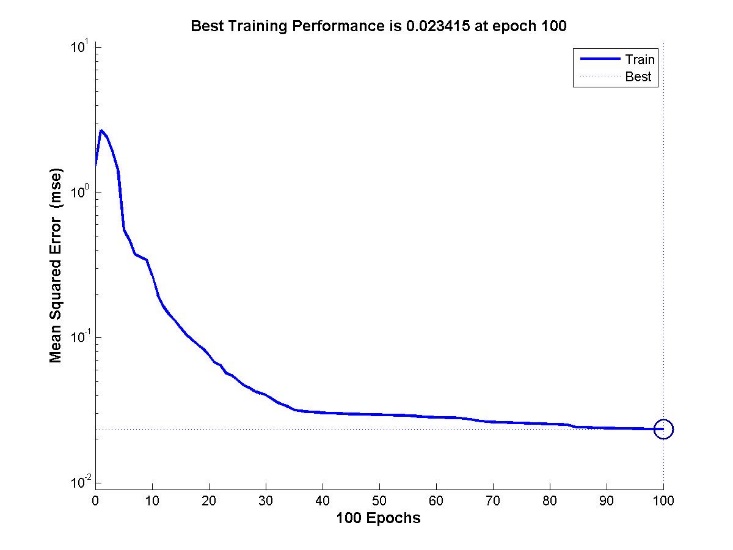
****

Figure 24: Gradient of the Function trained with Neural Network with 1 hidden layer of 10 neurons and training function of Bayesian Regularization

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = [40]
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

****Figure 25: Training Error vs No. of Epochs for Neural Network with 1 hidden layer with 40 neurons and Bayesian Regularization training

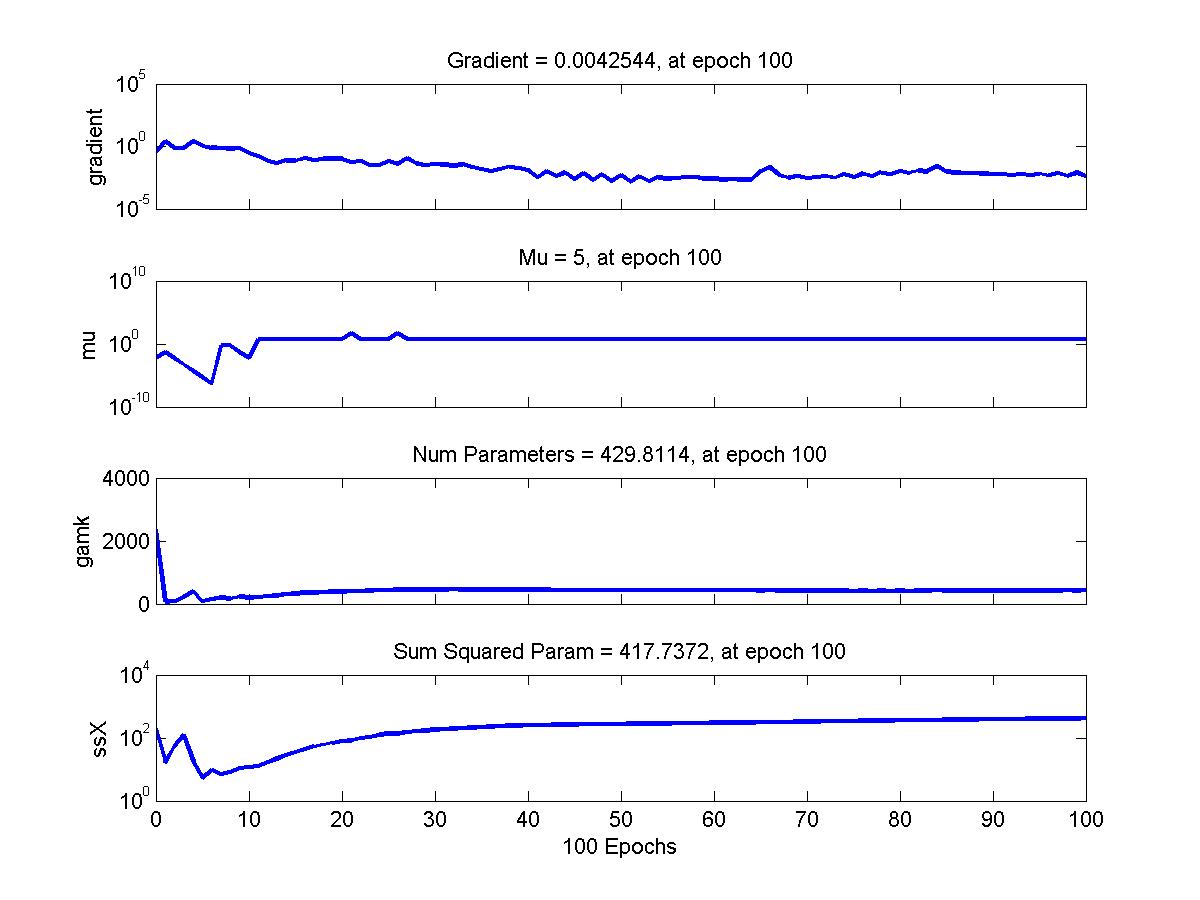
****

Figure 26: Gradient of the Function trained with Neural Network with 1 hidden layer of 10 neurons and training function of Bayesian Regularization

# Training Function – Gradient Descent

Since in the Lavenberg-Marquardt Algorithms and the Bayesian Regularization training algorithms, the training rate parameter cannot be controlled, testing that parameter with the gradient descent learning rule.

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = [10]
3. Max\_fail = 25
4. Maximum number of epochs = 200
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30
7. Learning Rate = 0.0001

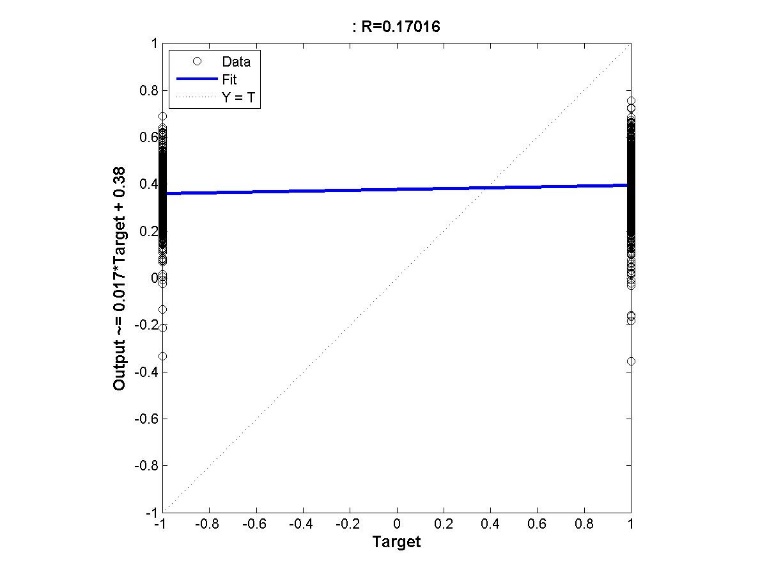
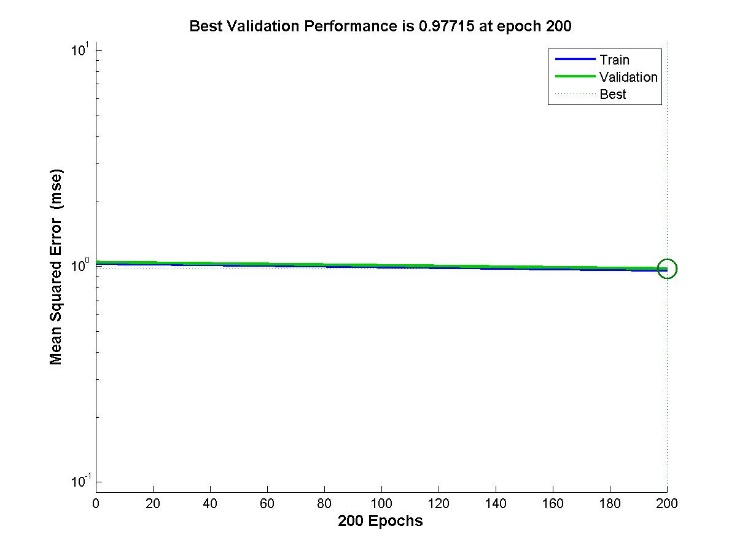


Figure 27: Training Error vs No. of Epochs for Neural Network with 1 hidden layer with 10 neurons and Gradient Descent Learning with learning rate of 0.0001

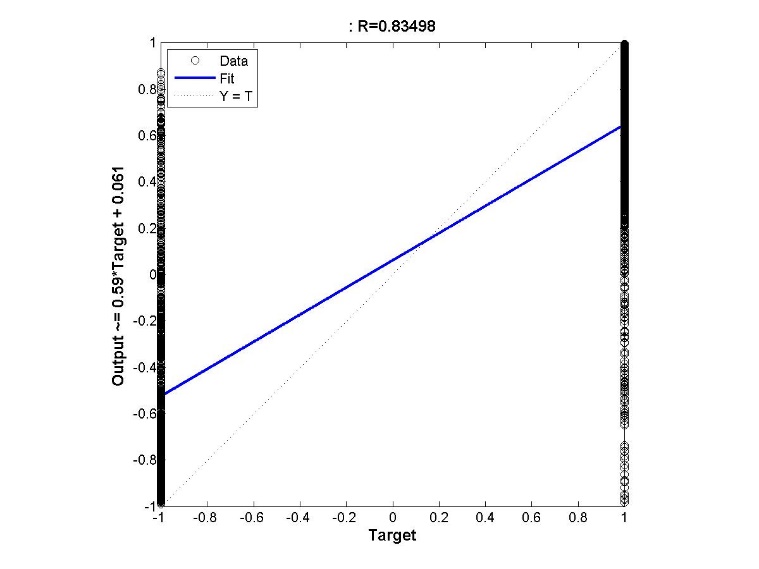
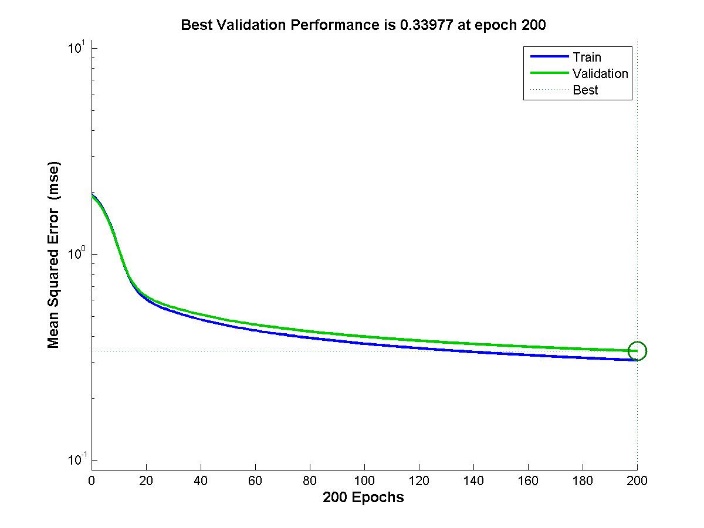


Figure 28: Training Error vs No. of Epochs for Neural Network with 1 hidden layer with 10 neurons and Gradient Descent Learning with learning rate of 0.05

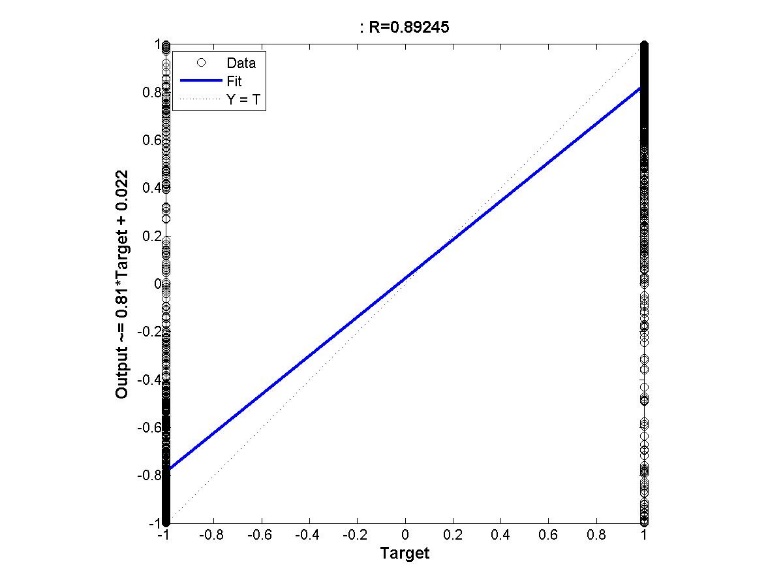
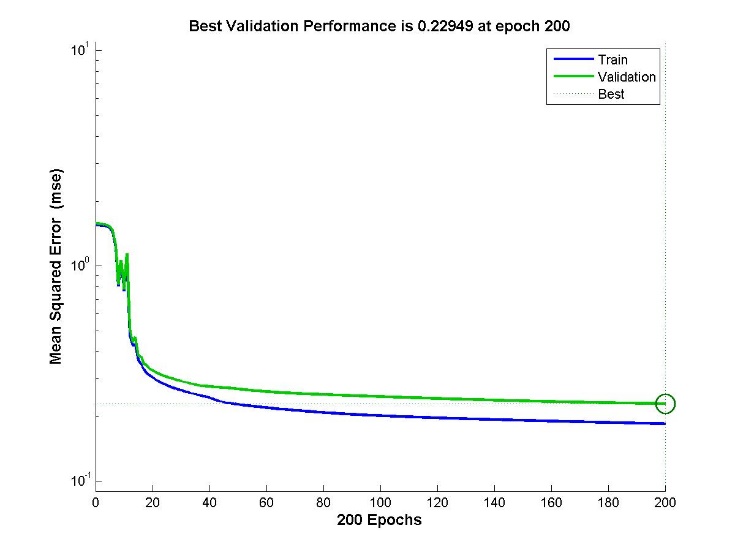


Figure 29: Training Error vs No. of Epochs for Neural Network with 1 hidden layer with 10 neurons and Gradient Descent Learning with learning rate of 0.26

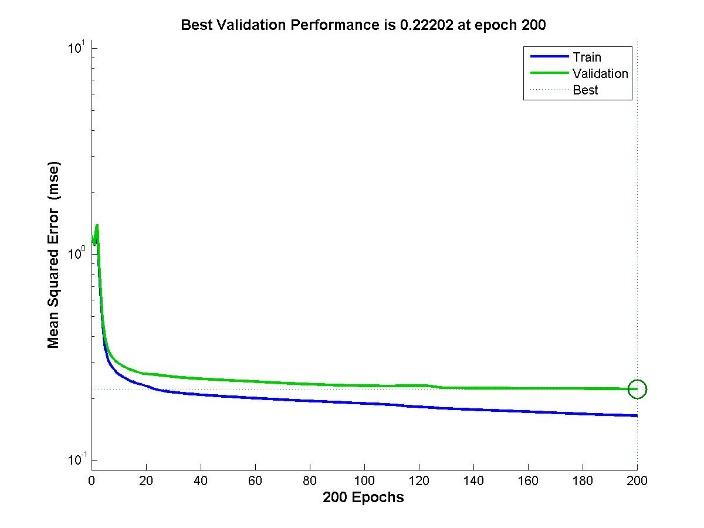
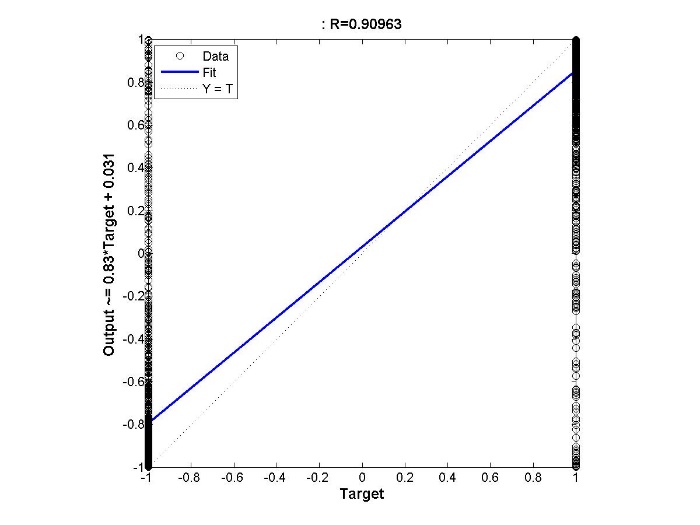
 

Figure 30: Training Error vs No. of Epochs for Neural Network with 1 hidden layer with 10 neurons and Gradient Descent Learning with learning rate of 0.51

## Multi-Layer Networks – Number of hidden layers 2

1. Number of hidden layers = 2
2. Number of neurons in the hidden layer = [10 10]
3. Max\_fail = 25
4. Maximum number of epochs = 200
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

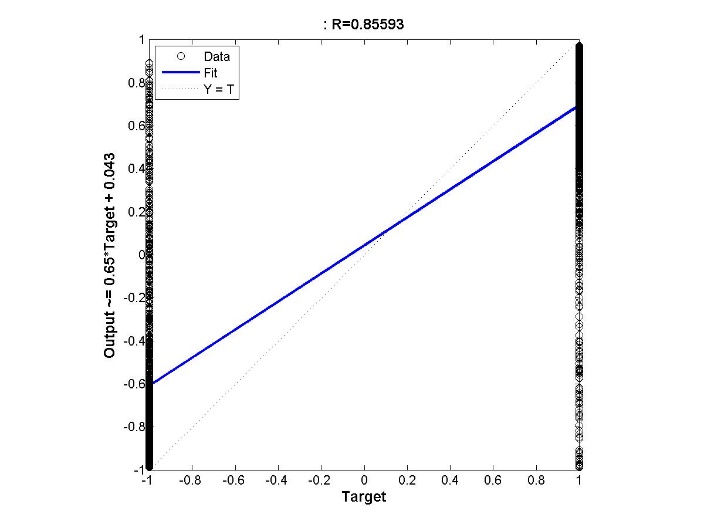
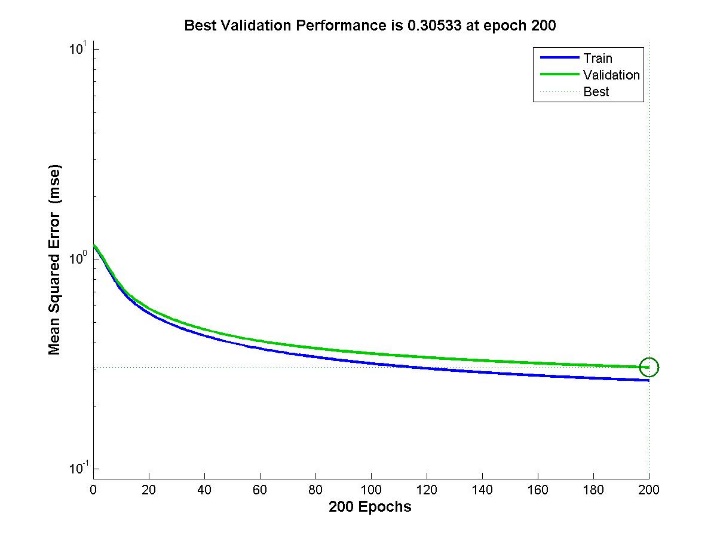


Figure 31: Training Error vs No. of Epochs for Neural Network with 2 hidden layers configuration of [10 10] and Gradient Descent Learning with learning rate of 0.01

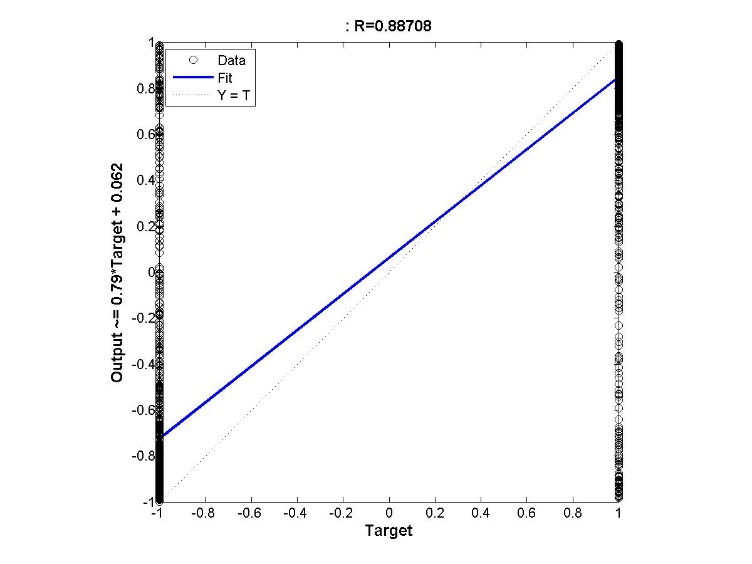
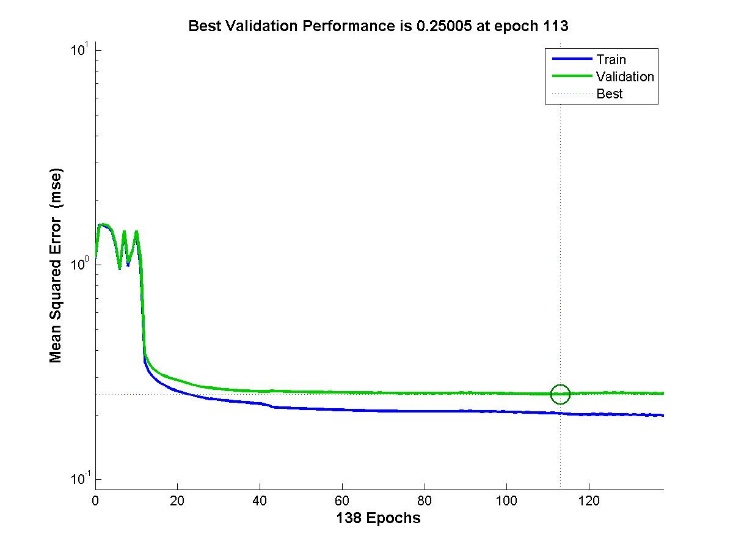


Figure 32: Training Error vs No. of Epochs for Neural Network with 2 hidden layers configuration of [10 10] and Gradient Descent Learning with learning rate of 0.26

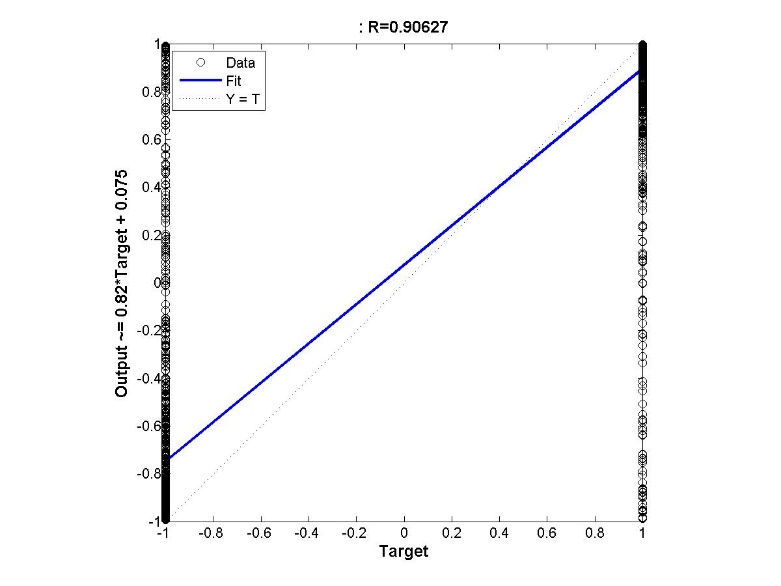
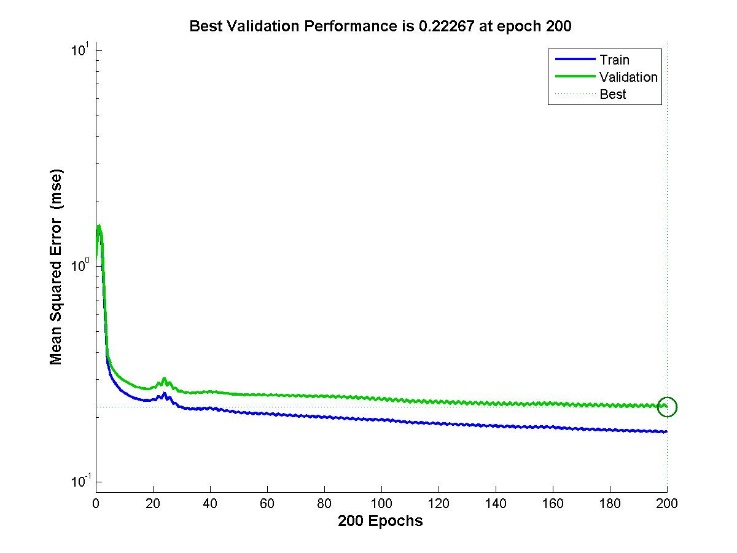


Figure 33: Training Error vs No. of Epochs for Neural Network with 2 hidden layers configuration of [10 10] and Gradient Descent Learning with learning rate of 0.51

# Part 2: Function Approximation

### Lavenberg-Marquardt Algorithm

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 10
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

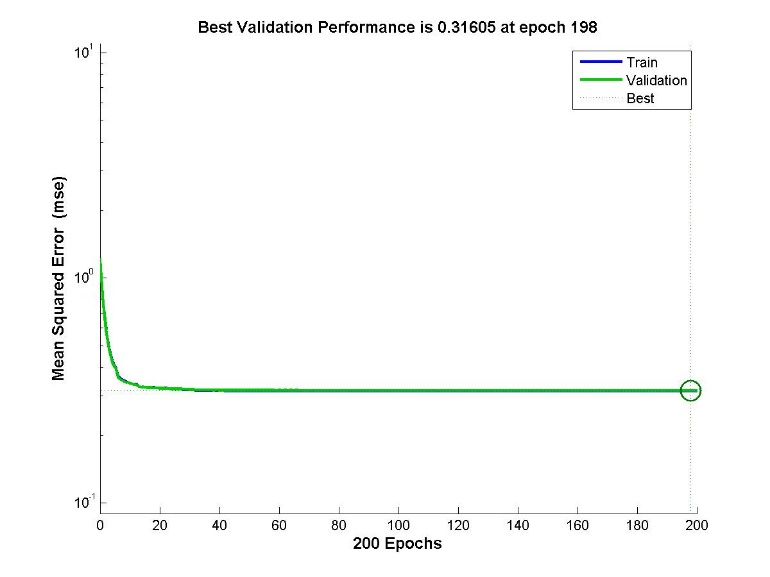


Figure 34: Training and Validation Error vs Number of Epochs

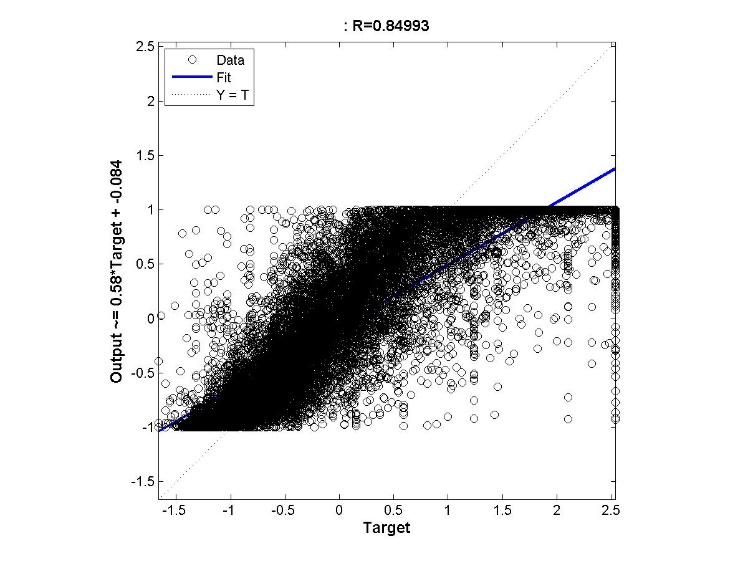
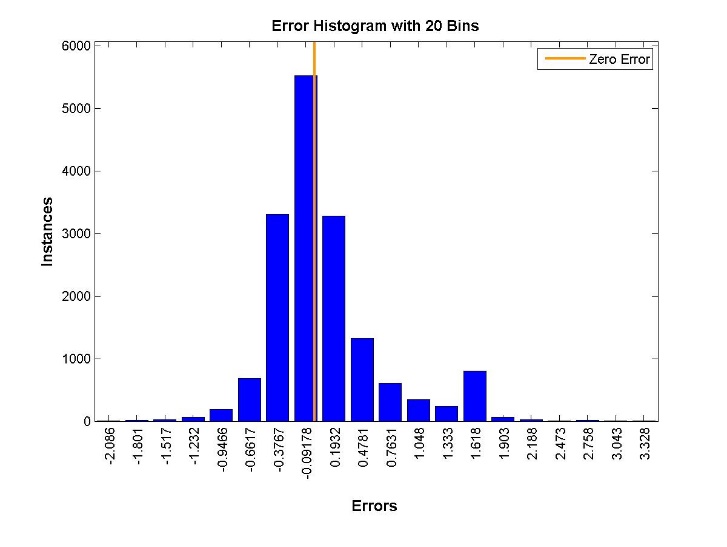
 

Figure 35: (a) Regression figure (b) Histogram for Neural Network with 1 hidden layer with 10 neurons

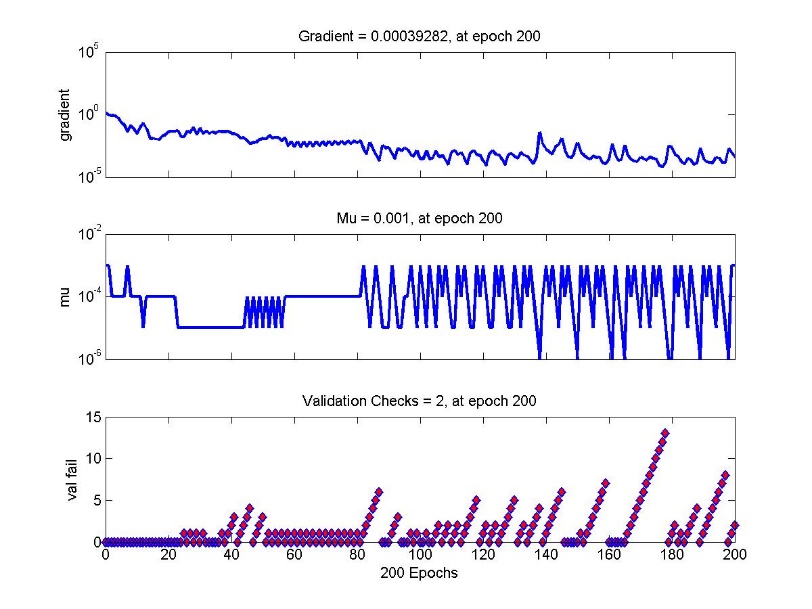


Figure 36: Gradient of the Function trained with Neural Network with 1 hidden layer of 10 neurons

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 30
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

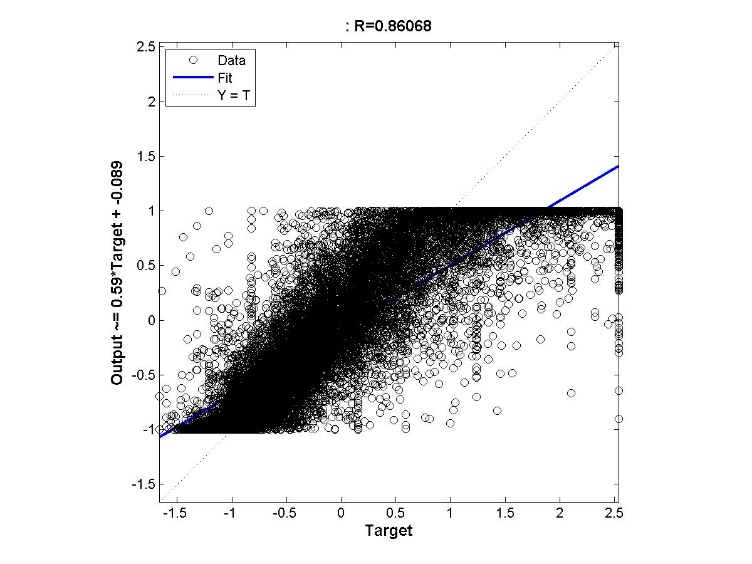
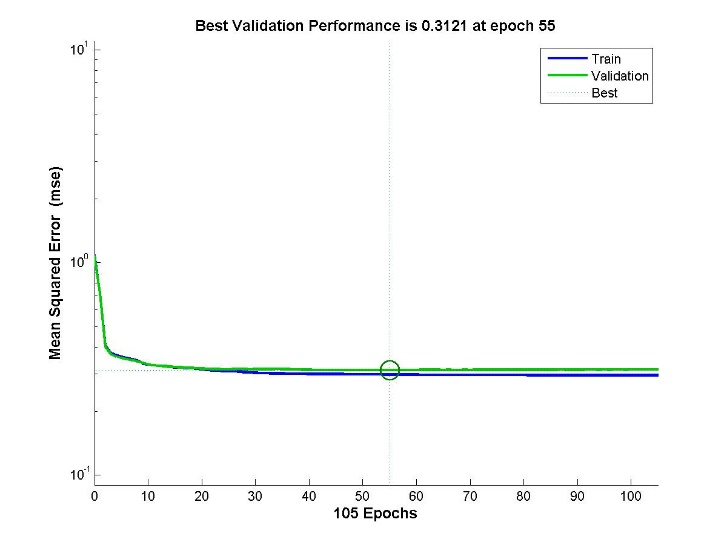


Figure 37: (a) Training and Validation Error vs No. of Epochs (b) Regression Model for function approximation of a neural network with 30 neurons in the single hidden layer

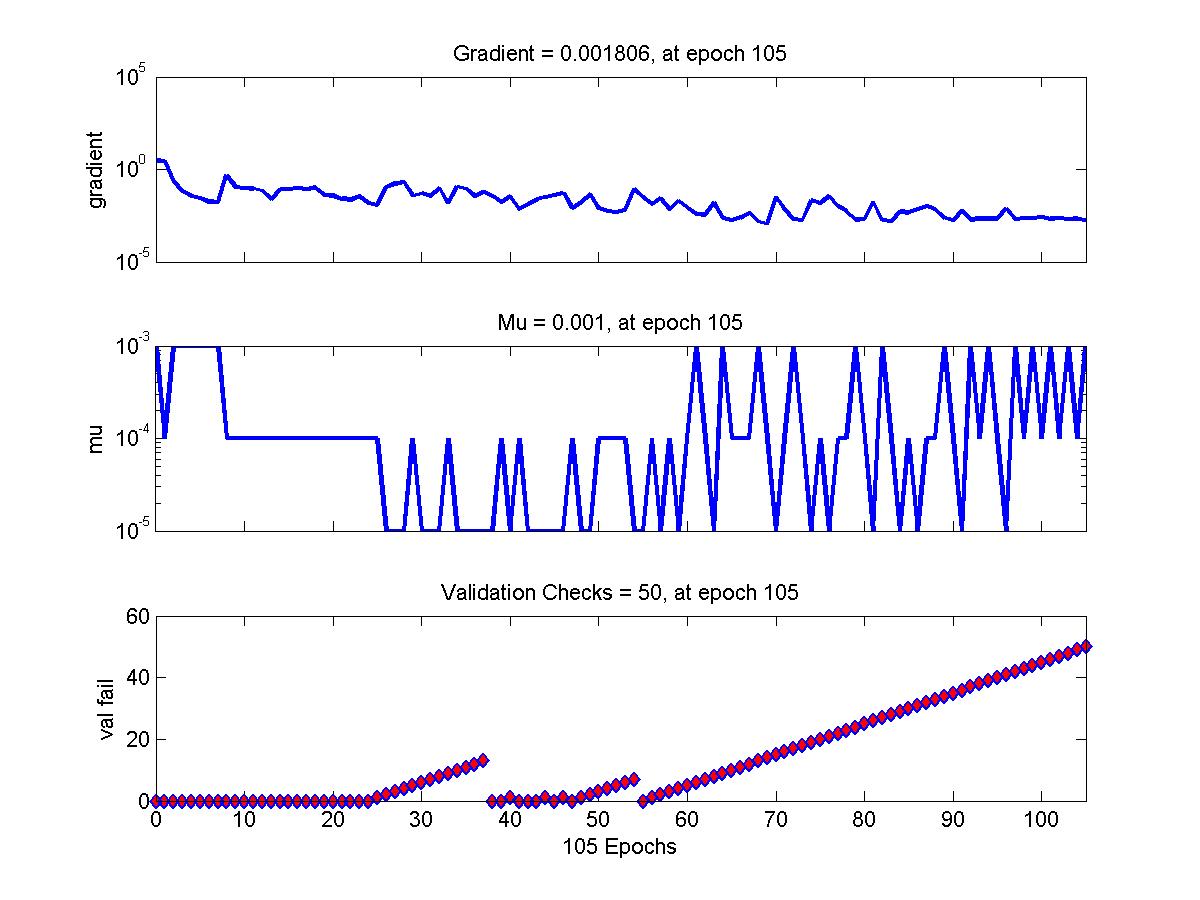


Figure 38: Gradient of the Function trained with Neural Network with 1 hidden layer of 30 neurons

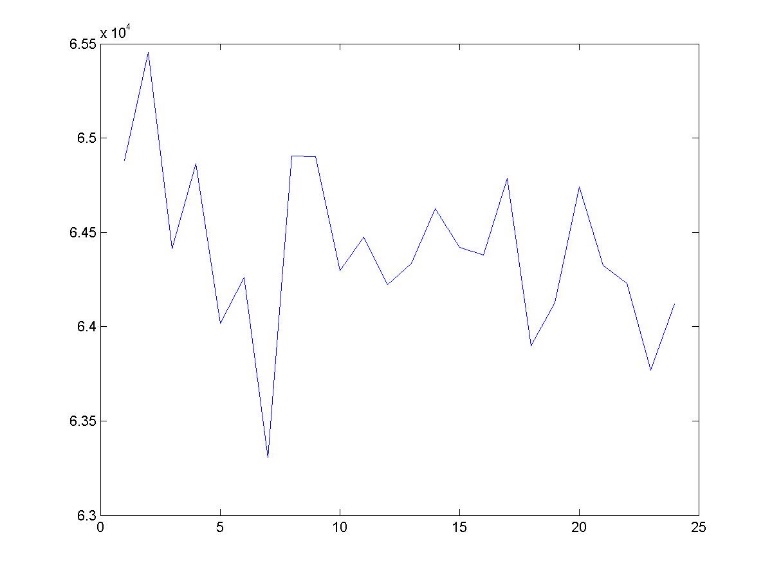
****

Figure 39: Test Error vs (Number of Neurons in the hidden layer – 10)/2 + 1

# Changing the Data Split Ration between Training and Validation Data

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 10
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 60:40

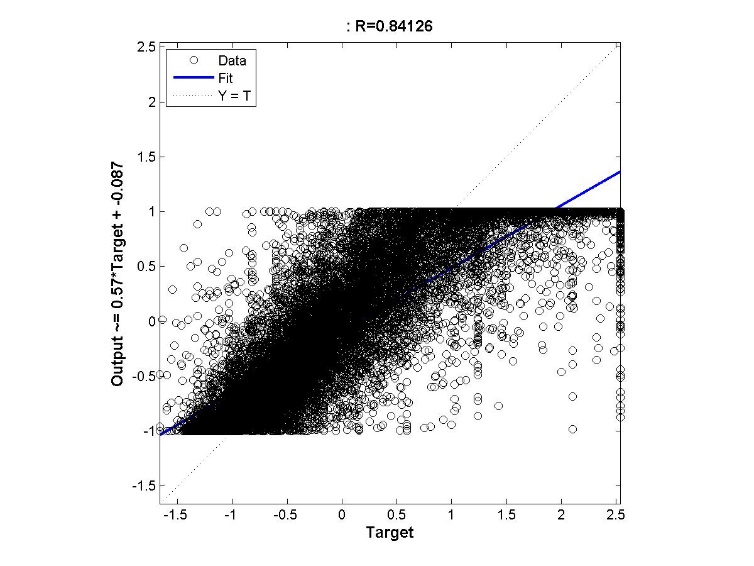
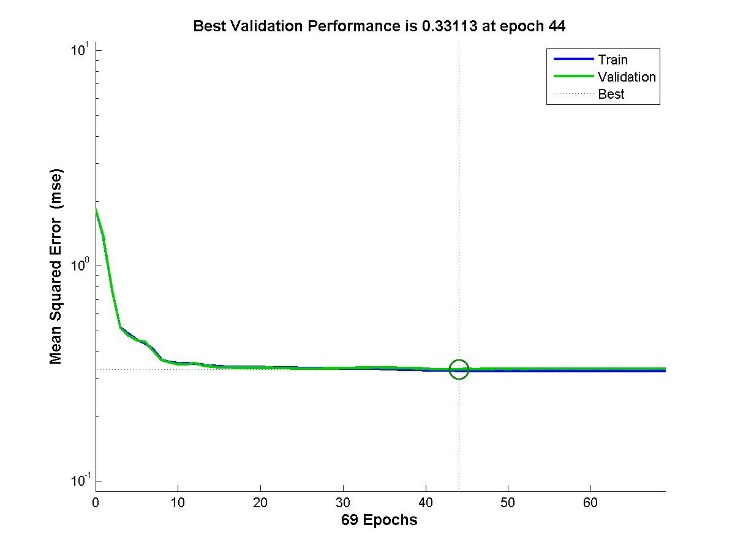


Figure 40: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 1 layer and Maximum Number of continuous Validation increase failures is 10

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 10
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 60:40

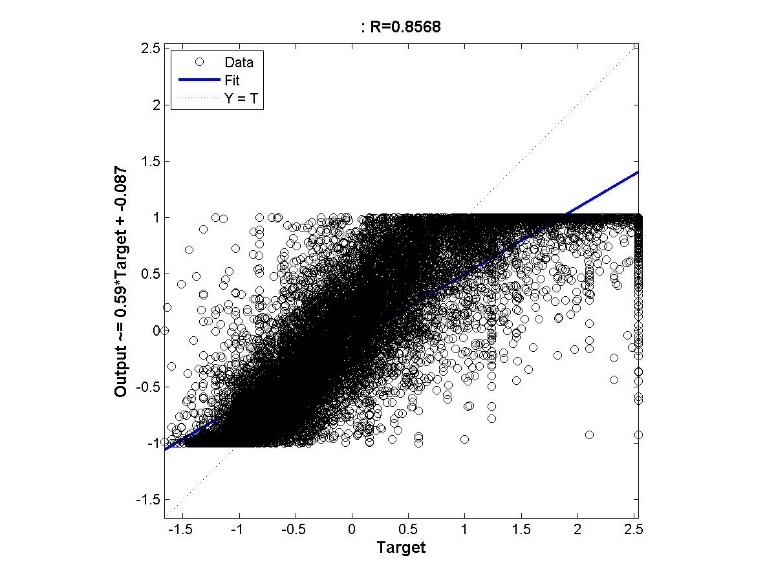
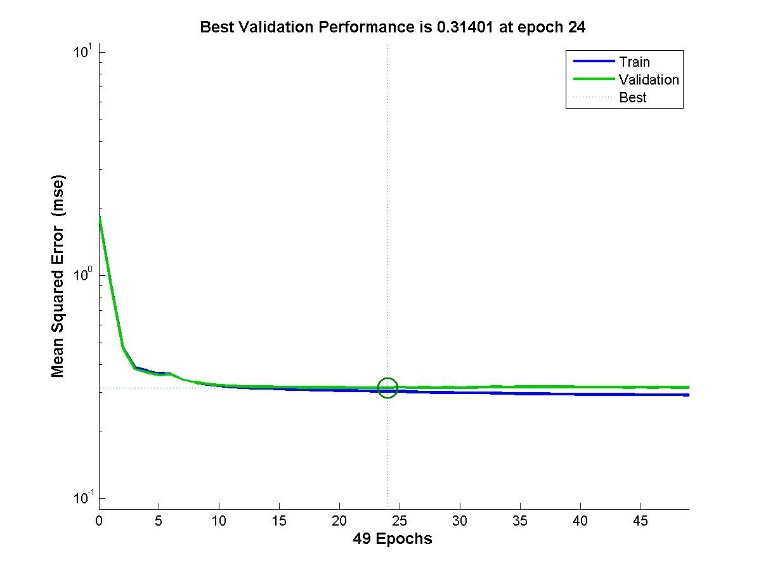


Figure 41: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 1 layer and Maximum Number of continuous Validation increase failures is 50

## Varying Maximum Validation

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 30
3. Max\_fail = 30
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

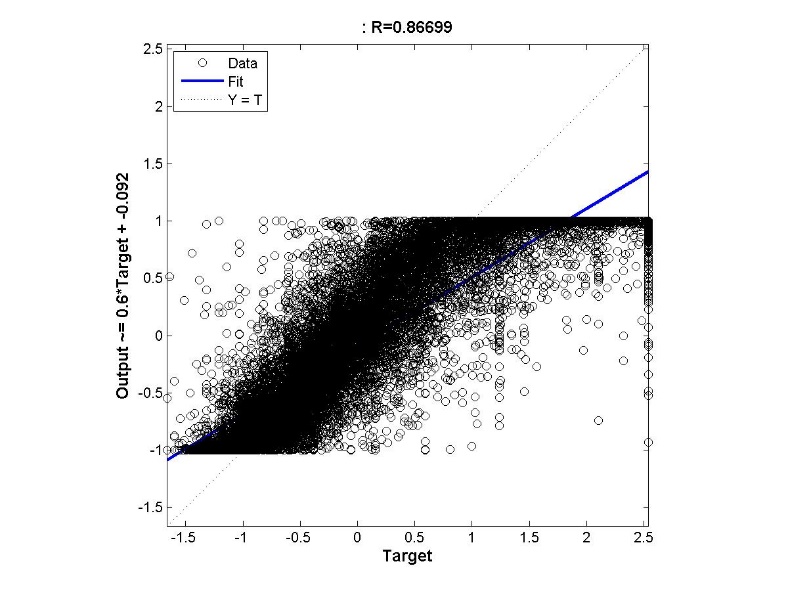
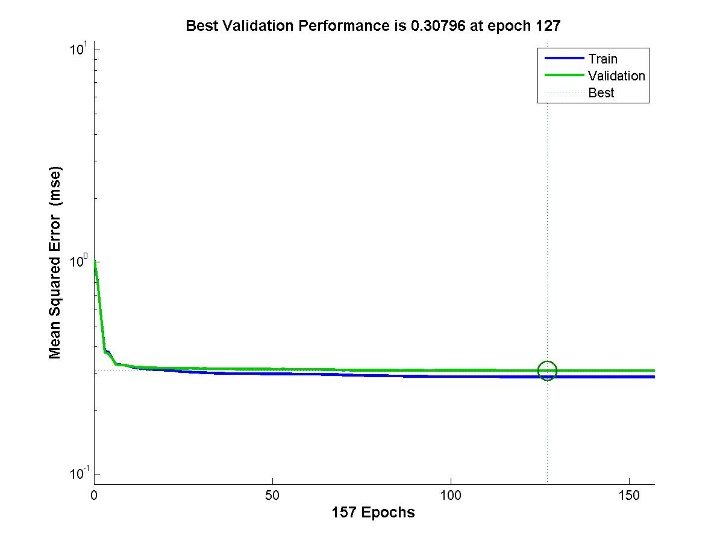


Figure 42: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 1 layer and Maximum Number of continuous Validation increase failures is 30

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = 30
3. Max\_fail = 60
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

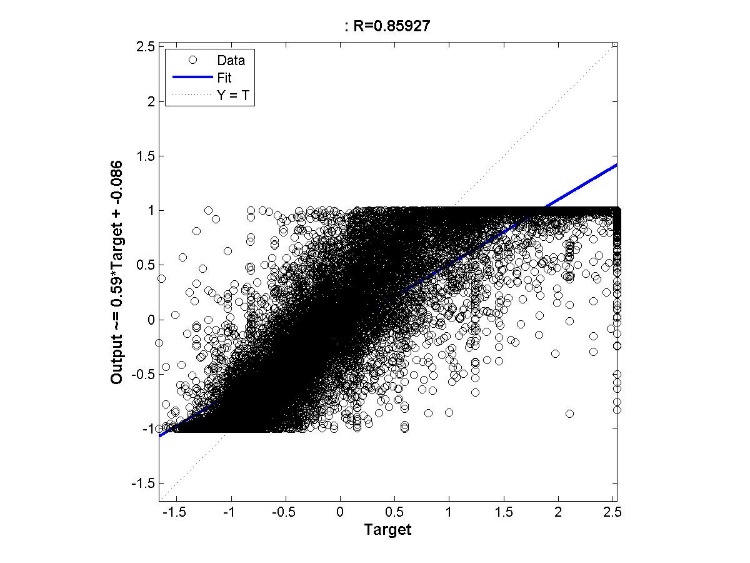
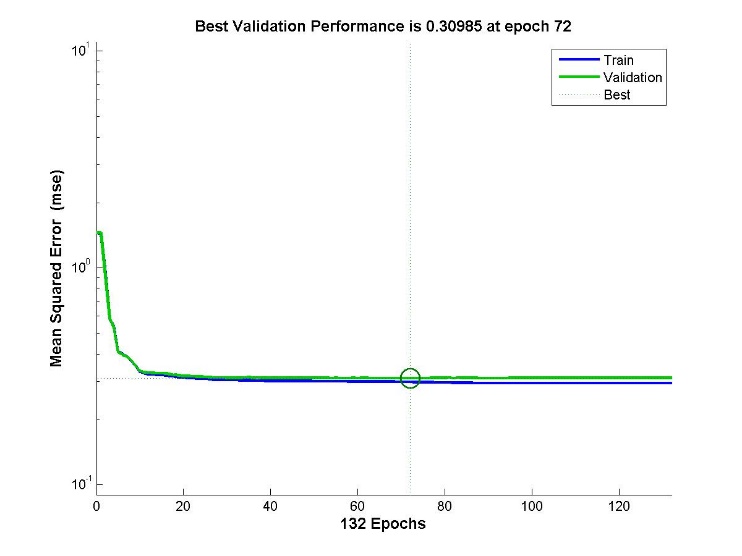


Figure 43: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 1 layer and Maximum Number of continuous Validation increase failures is 100

## Varying the number of hidden layers and the number of neurons in each layer

**Number of Layers = 2**

1. Number of hidden layers = 2
2. Number of neurons in the hidden layer = [20 20]
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

## C:\Users\Madhavan\Desktop\CZ4042_total\CZ4042_Given COde\Function Approximation\Trainlm_2_layers\performance_maxfail20_20.jpgC:\Users\Madhavan\Desktop\CZ4042_total\CZ4042_Given COde\Function Approximation\Trainlm_Varylayers\plotregression20_20.jpg

Figure 44: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 2 layer with hidden configuration of [20 20]



Figure 45: Training State of a neural network with 2 layer with hidden configuration of [20 20]

1. Number of hidden layers = 2
2. Number of neurons in the hidden layer = [30 30]
3. Max\_fail = 25
4. Maximum number of epochs = 100
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

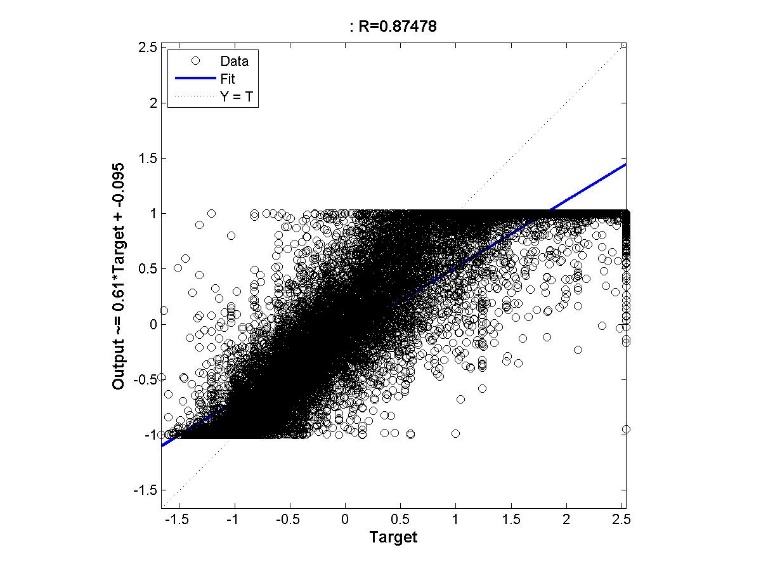
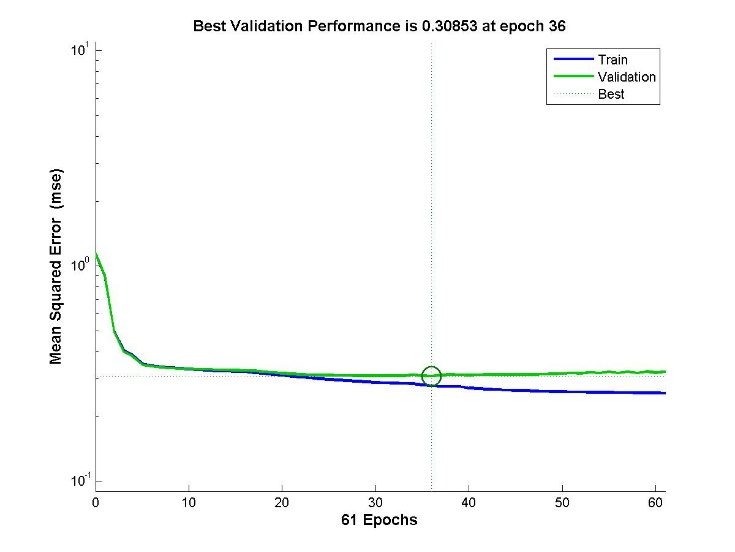


Figure 46: (a) Performance Measure Training and Validation Error vs No. of Epochs (b) Regression output for a neural network with 2 layer with hidden configuration of [30 30]



Figure 47: Training State of a neural network with 2 layer with hidden configuration of [20 20]

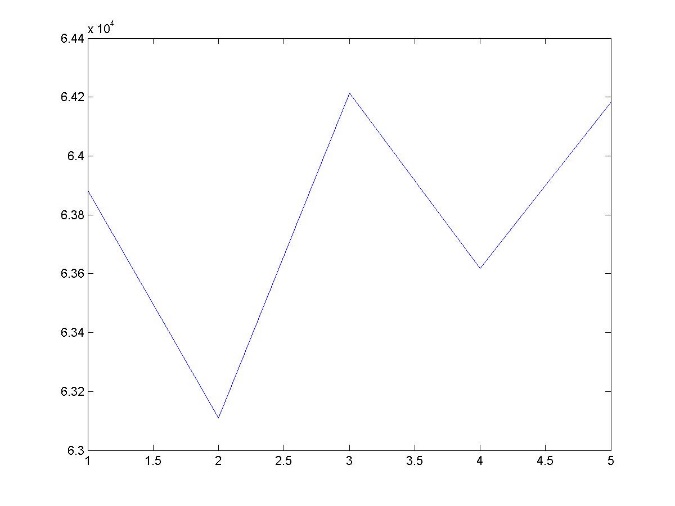
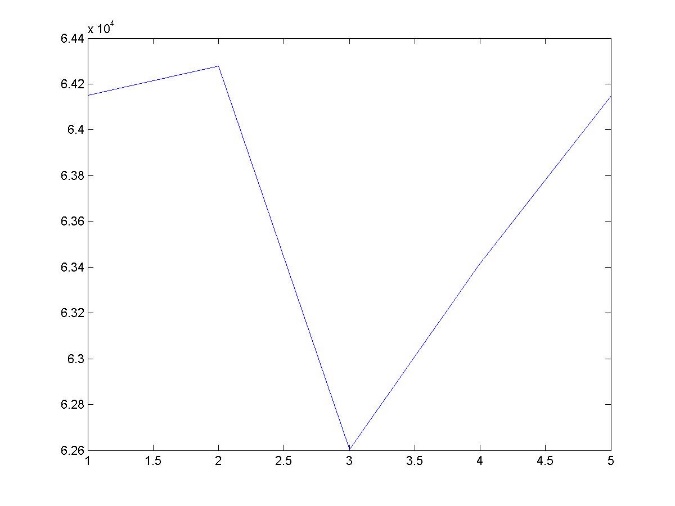


Figure 48: Testing Error (Misclassification Rate) vs Number of neurons/10 in the first layer (a) when the number of neurons in the second layer is fixed at 10 (b) when the number of neurons in the second layer is fixed at 20

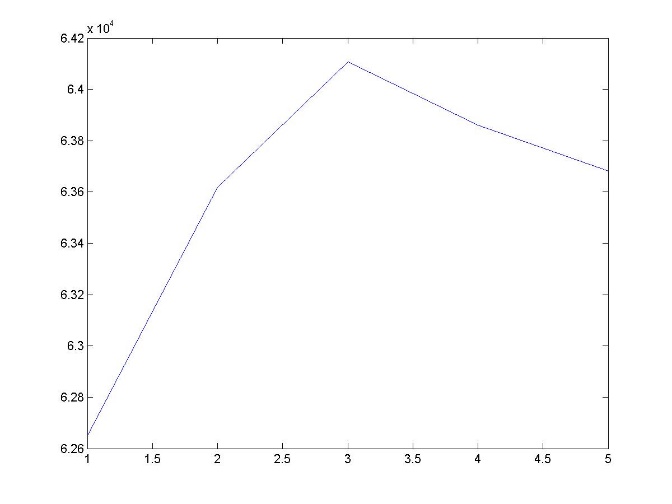
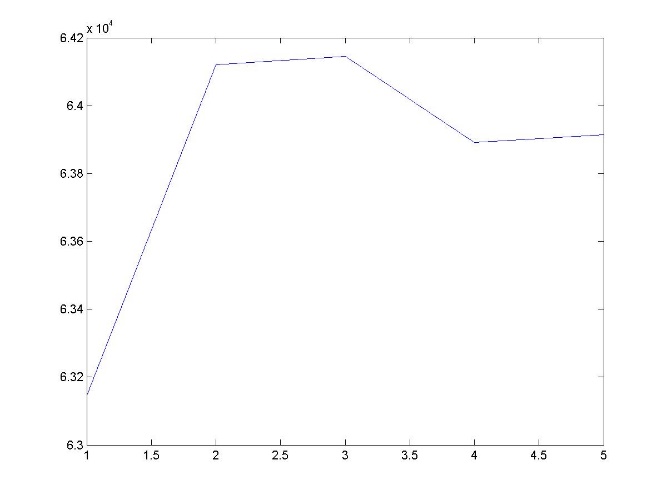
 

Figure 49: Testing Error (Misclassification Rate) vs Number of neurons/10 in the first layer (a) when the number of neurons in the second layer is fixed at 20 (b) when the number of neurons in the second layer is fixed at 40

### Number of layers = 3

1. Number of hidden layers = 3
2. Number of neurons in the hidden layer = [20 20 20]
3. Max\_fail = 25
4. Maximum number of epochs = 200
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30

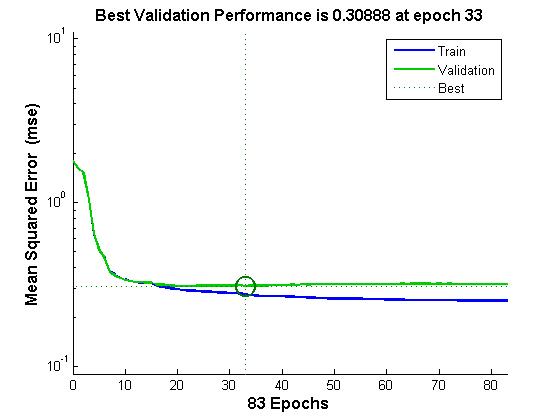


Figure 50: Performance Measure Training and Validation Error vs No. of Epochs for a neural network with 3 layer with hidden configuration of [20 20 20]

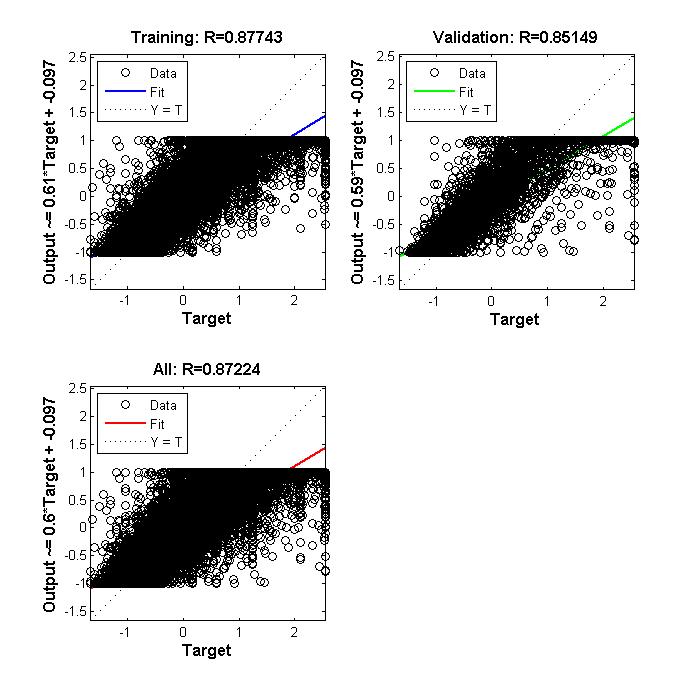


Figure 51: (a) Training Regression (b) Validation Regression for a neural network with 3 layer with hidden configuration of [20 20 20]

# Training Function – Gradient Descent

Since in the Lavenberg-Marquardt Algorithms and the Bayesian Regularization training algorithms, the training rate parameter cannot be controlled, testing that parameter with the gradient descent learning rule.

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = [10]
3. Max\_fail = 25
4. Maximum number of epochs = 200
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30
7. Learning Rate = 0.01

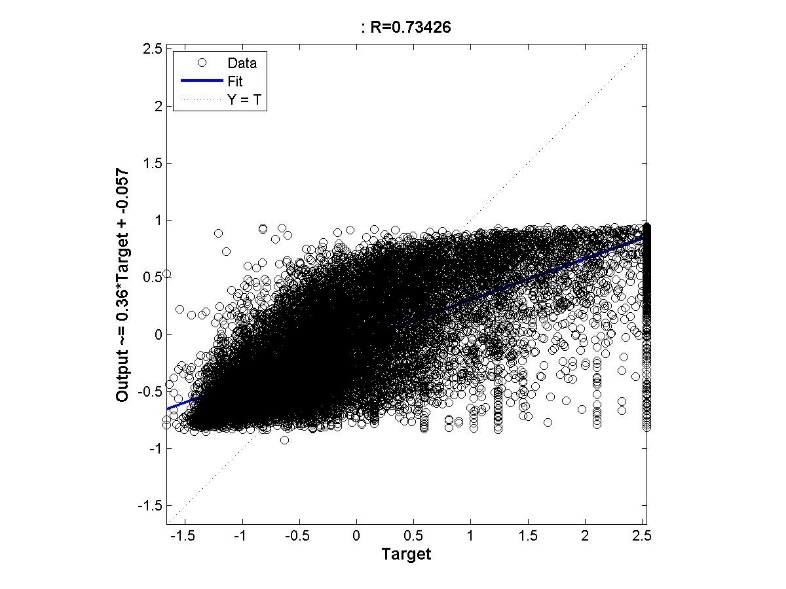
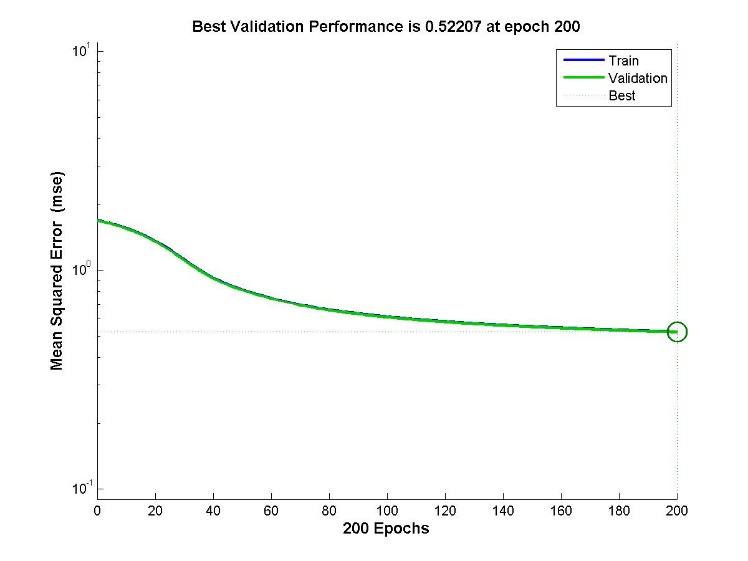


Figure 52: Training Error vs No. of Epochs for Neural Network with 1 hidden layer with 10 neurons and Gradient Descent Learning with learning rate of 0.01

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = [10]
3. Max\_fail = 25
4. Maximum number of epochs = 200
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30
7. Learning Rate = 0.26

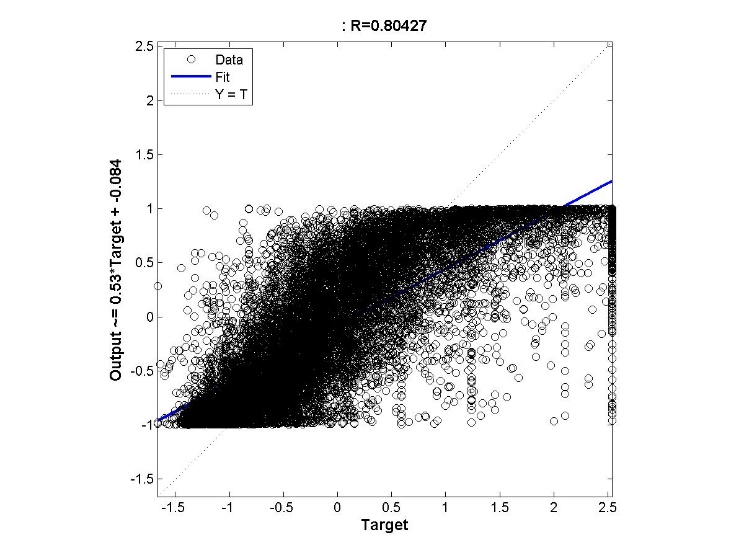
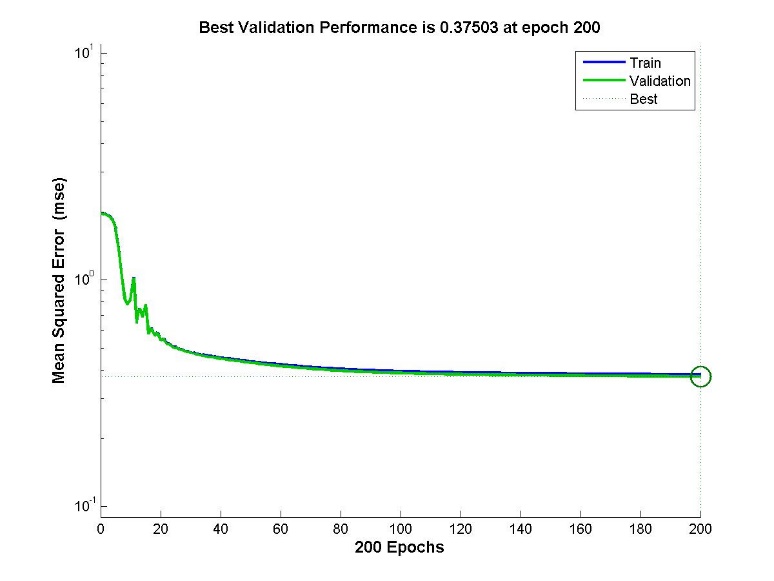


Figure 53: Training Error vs No. of Epochs for Neural Network with 1 hidden layer with 10 neurons and Gradient Descent Learning with learning rate of 0.26

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = [10]
3. Max\_fail = 25
4. Maximum number of epochs = 200
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30
7. Learning Rate = 0.51

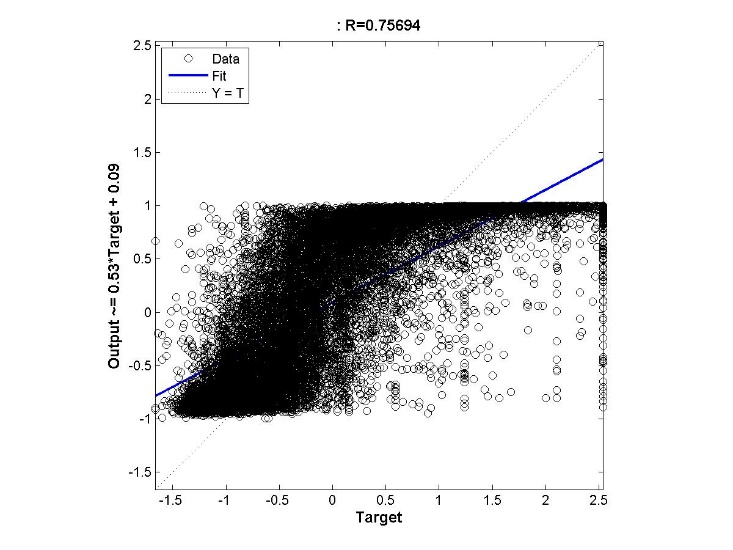
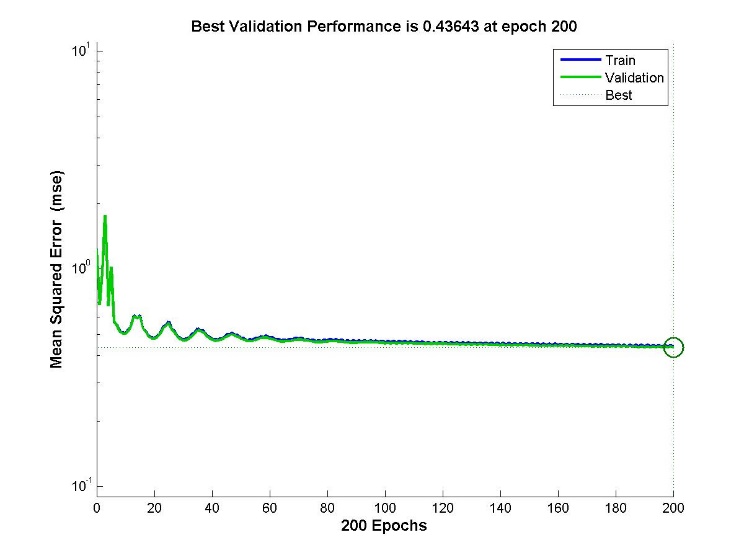


Figure 54: Training Error vs No. of Epochs for Neural Network with 1 hidden layer with 10 neurons and Gradient Descent Learning with learning rate of 0.51

1. Number of hidden layers = 1
2. Number of neurons in the hidden layer = [10]
3. Max\_fail = 25
4. Maximum number of epochs = 200
5. Minimum Gradient = 1e-20
6. Training Data Split = 70:30
7. Learning Rate = 0.75

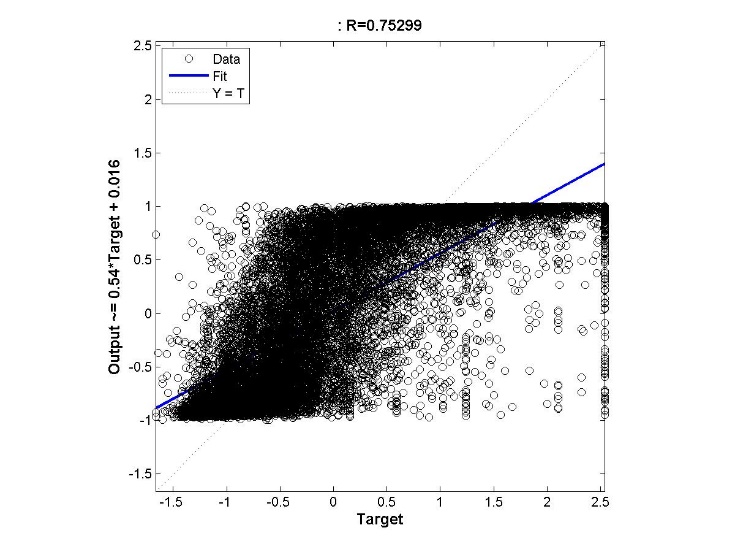
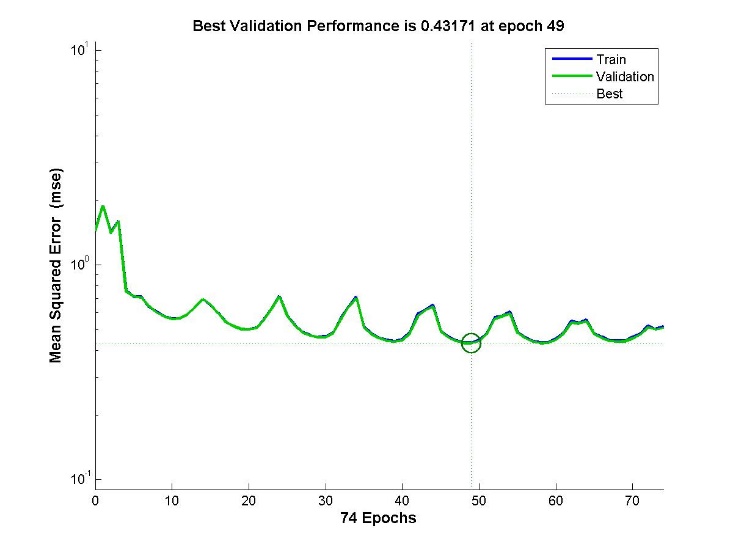


Figure 55: Training Error vs No. of Epochs for Neural Network with 1 hidden layer with 10 neurons and Gradient Descent Learning with learning rate of 0.76