**ASSIGNMENT-2**

**HOUSE PRICE PREDICTION USING**

**NEURAL NETWORKS**

GROUP MEMBERS:

VATSAL GUPTA: 2018A7PS0198H

MIHIR BANSAL: 2018A7PS0215H

BANDARU ABHINAV: 2018A7PS0236H

**FEATURE SCALING:**

In our Neural Networks model, we have scaled the features of our training set by normalizing the data-set.

X:=(x-mean)/(standard deviation)

By normalizing data-set, they will nearly share the same mean,variance and bounds.

----------------------------------------------------------------------------------

**DESIGN:**

The design of the algorithm is based on the neural network classification algorithm which uses 2 layers with each layer having an initial weight vector and a set of neurons(nodes) to compute the final weights and test it on the testing data with an 80:20 train:test split ratio.

The function feedforward() for will compute the dot product of the input data with the 1st layer weight vectors and pass the matrix through an activation function. Next the obtained neurons will be again dot product with the 2nd layer weights and on passing them through the final activation function, they will give us the output layer i.e. the final predicted value.

z1 = tanh(x @ w1)

y\_pred = sigmoid(z1 @ w2)

backprop() function is used to find the gradient of error function w.r.t weight vectors for each layers and is multiplied with learning rate to obtain the subtraction amount from previous weight vector.

The gradients are obtained by the back propagation algorithm.

grad2 = 1/n \* ((z1.transpose()) @ ((y\_pred - y)\*(sigmoid\_derivative(y\_pred))))

grad1 = 1/n \* ((x.transpose()) @ (tanh\_derivative(z1)\*((y\_pred - y) @ (w2.transpose()))))

w1 = w1 - (learnRate \* grad1)

w2 = w2 - (learnRate \* grad2)

As soon as the difference in the error reaches less than 10-6, we will break out of the loop and test the final w vector with the testing set to compute its accuracy and F-score.

----------------------------------------------------------------------------------------

**RESULTS:**

**TRAINING SET-1**

Learning rate=0.1

No of iterations: 493

No of hidden neurons: 5

Activation function for 1st layer: tanh(x)

Activation function for 2nd layer: sigmoid(x)

The final weights for 1st layer:

[[-0.00567152 0.00437654 -0.00432653 -0.00197803 -0.00385769]

[ 0.05616118 0.53538211 0.18856474 0.44461345 0.22731768]

[ 0.14721487 1.21864222 0.45327101 1.0140387 0.53722414]

[ 0.01419201 0.2503264 0.07392726 0.20916892 0.09458546]

[ 0.08938995 0.68468556 0.26478257 0.56633754 0.30989498]

[ 0.10745835 0.83046902 0.32330792 0.69846991 0.38100437]

[ 0.05816967 0.58266894 0.17497097 0.45214107 0.20915425]

[-0.0107221 -0.2299543 -0.08164224 -0.2069334 -0.10462637]

[ 0.05455617 0.36361452 0.13134603 0.28647452 0.1504369 ]

[ 0.05888126 0.46931183 0.15403533 0.35868476 0.17843845]

[ 0.08443488 0.62230724 0.2342364 0.5119368 0.27364978]]

The final weights for 2nd layer:

[[0.13144729]

[1.22301904]

[0.44840297]

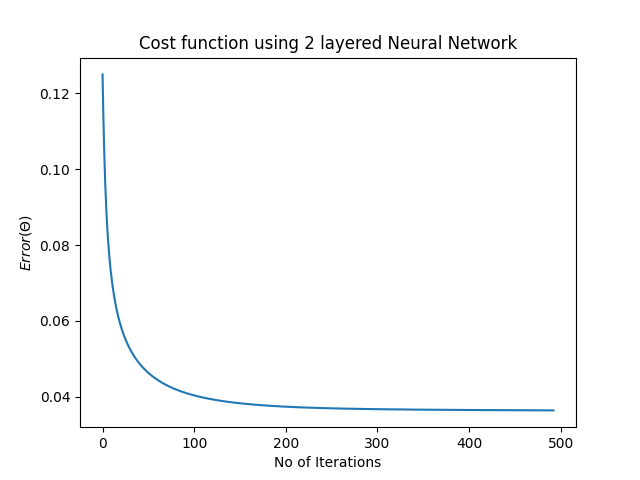
[1.0166417 ]

[0.5357002 ]]

Testing Accuracy: 90.06849315068493 %

F-Score: 90.16949152542372 %

LOSS FUNCTION GRAPH:



**TRAINING SET-2**

Learning rate=0.1

No of iterations: 1444

No of hidden neurons: 9

Activation function for 1st layer: sigmoid(x)

Activation function for 2nd layer: sigmoid(x)

The final weights for 1st layer:

[[ 0.12131556 -0.33913893 -0.03744388 -0.07811636 0.01116672 -0.96163651

-0.76789705 -1.14932013 0.03013628]

[-0.46494782 0.26767609 -0.14112684 -0.05929597 -0.25242047 0.75507828

0.61643502 0.88492081 -0.2944304 ]

[-1.49456024 0.83813611 -0.43026148 -0.17159805 -0.78709621 2.39294929

1.94759777 2.81054457 -0.92431214]

[-0.21541049 0.04213021 -0.10587699 -0.07114084 -0.15094065 0.10376166

0.09516974 0.10851479 -0.16631011]

[-0.64569429 0.44465042 -0.16796501 -0.0528396 -0.32609566 1.28180545

1.03666049 1.51384215 -0.38720753]

[-1.02795596 0.62748797 -0.27426407 -0.09402527 -0.52316534 1.80798775

1.46491536 2.13143451 -0.61967364]

[-0.63367542 0.36648549 -0.18739008 -0.07687315 -0.33782867 1.06601006

0.86498147 1.25445345 -0.39533724]

[ 0.2385116 -0.02977543 0.09684384 0.06366932 0.14641654 -0.0771811

-0.07196696 -0.07887168 0.16539592]

[-0.27937465 0.27527507 -0.04677533 0.00997873 -0.12177572 0.7952485

0.63741299 0.94655623 -0.1508869 ]

[-0.53247224 0.40495202 -0.13686446 -0.03795116 -0.26863643 1.19838192

0.96453723 1.41964619 -0.3190216 ]

[-0.82016247 0.51971248 -0.21645014 -0.07217842 -0.4153141 1.49975445

1.21367067 1.76993128 -0.49253886]]

The final weights for 2nd layer:

[[-1.12856913]

[ 0.4962004 ]

[-0.43640997]

[-0.22109604]

[-0.70589518]

[ 1.12315378]

[ 0.96473565]

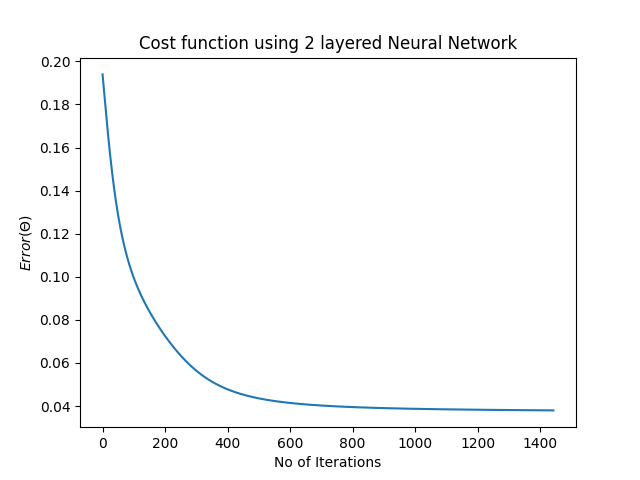
[ 1.26803084]

[-0.79851845]]

Testing Accuracy: 88.6986301369863 %

F-Score: 87.91208791208791 %

LOSS FUNCTION GRAPH:



**TRAINING SET-3**

Learning rate=0.1

No of iterations: 675

No of hidden neurons: 9

Activation function for 1st layer: tanh(x)

Activation function for 2nd layer: sigmoid(x)

The final weights for 1st layer:

[[ 0.03704754 0.05511628 0.08234947 0.00652956 0.03476188 0.00765776

0.00139685 0.07087176 0.02961205]

[ 0.23181371 0.32640454 0.47396555 0.08123143 0.22096767 0.08872277

0.03345115 0.41852288 0.19704262]

[ 0.39347761 0.53563358 0.77009855 0.15141963 0.37701036 0.16438082

0.06705576 0.67788905 0.34036107]

[ 0.03684394 0.08007034 0.18013826 0.00377103 0.03305653 0.00447938

0.00123205 0.13795288 0.02552994]

[ 0.22357947 0.28051085 0.3491543 0.09251852 0.21597441 0.10058534

0.0366653 0.32425449 0.19832974]

[ 0.27453681 0.35774176 0.49331198 0.11273081 0.26437007 0.12201345

0.05121678 0.43825502 0.24129497]

[ 0.1499445 0.20755397 0.32162675 0.06093134 0.14371416 0.06583446

0.02766416 0.27372167 0.1300828 ]

[-0.05241074 -0.10124612 -0.22535187 -0.00808042 -0.04832345 -0.00961422

-0.00158445 -0.17676697 -0.03994441]

[ 0.1007424 0.11375405 0.11269559 0.051465 0.09833769 0.05506746

0.02384524 0.11228923 0.09251436]

[ 0.13025211 0.18123055 0.29214572 0.05759626 0.12524948 0.06199319

0.02579164 0.24702893 0.11448353]

[ 0.18887134 0.2419508 0.33103384 0.08208704 0.1824381 0.08865119

0.03629126 0.29556398 0.16777714]]

The final weights for 2nd layer:

[[0.48379854]

[0.69667607]

[1.06307823]

[0.15818025]

[0.45982723]

[0.17372931]

[0.06475443]

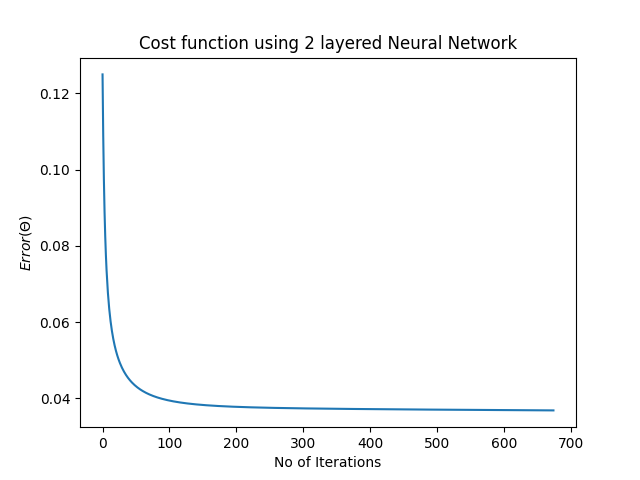
[0.91737092]

[0.4071845 ]]

Testing Accuracy: 92.12328767123287 %

F-Score: 92.45901639344262 %

LOSS FUNCTION GRAPH:



----------------------------------------------------------------------------------

**CONCLUSIONS:**

1.Activation Function:

We observe that selecting tanh(x) as the activation function for the input layer and sigmoid(x) function for the output layer led to better results in terms of accuracy and F-score as compared to using sigmoid(x) for both the layers. The former also had a good computation efficiency giving us the global minima in lesser number of iterations. Altogether, both the activation functions are good as they have a smooth gradient that prevents jumps in output values.

2. No of Hidden Neurons

We observe that on keeping the number of hidden neurons to about 2/3rd of the number of neurons in the previous layer led to better results as evident from training sets-2 & 3 which had 9 hidden neurons in the 1st layer as compared to 5 neurons in the hidden layer in training set-1.