

REPORT: Neural Networks

1. Introduction:

In this report, we try to analyse the performance (accuracy, fscore, recall and precision) of a neural network that we built using NumPy. The implementation of this network is similar to that of Keras and the whole code has been written in a completely modularized fashion.

2. Data Preprocessing:

The dataset *housepricesdata.csv* contained 1460 data points, each containing 10 input features (representing house features) and one binary target variable (representing if the house price is above or below the median price). In the preprocessing stage, all features were standardized and randomly split into *train_set* and *test_set* (80-20 split ratio).

3. Activation functions:

- linear: $y = x$
- tanh: $y = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
- sigmoid: $y = \frac{1}{1 + e^{-x}}$
- ReLU: $y = \max(0, x)$

4. Loss function:

- $L_n = -t_n \log(y_n) - (1 - t_n) \log(1 - y_n)$

5. Network Architecture:

5.1 Two-Layer NN

The first model we built was a 2-layer neural network (i.e. with one hidden layer). We experimented with activations like linear, ReLU and tanh for the hidden layer. The input layer had 10 neuron units with a ReLU activation (for best results. A slight drop in accuracy was observed when linear activation was used for the input layer). The final layer had an activation of sigmoid for binary classification. Output value greater than or equal to 0.5 was taken to represent positive classes and less than 0.5 to represent negative classes. `np.random.random()` and `np.random.randn()` methods were used to randomly initialize weights and biases associated network to initialize the weights according to a Uniform or Guassian distribution respectively. Best results were obtained using uniformly

distributed weights/biases with a *seed=5* and 4 hidden neurons with ReLU and *learning_rate=0.001* with 20000 training epochs. The loss was observed to have more fluctuations using this configuration after 40000 epochs and thus lowering the accuracy (images attached below). The avg. loss using this configuration was observed to be 0.7 (per data point). The highest accuracy achieved by this model using this configuration was 87.67% and the lowest accuracy was 80% with an average accuracy of 84%.

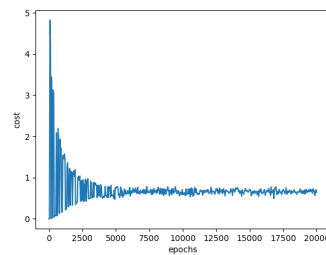


Figure 1: Plot of Loss vs Epochs

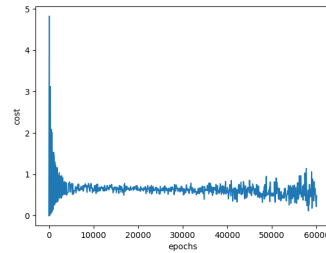


Figure 2: Fluctuations appear from 40000 iterations

Top metrics achieved by the model:

- Best testing accuracy: 87.67%
- Best testing recall: 97.33%
- Best testing precision: 80.11%
- Best testing fscore: 86.50%

5.2 Three-Layer NN

The second model we built and experimented with was a three layer neural network. The accuracy of the three layer neural network was found to be better than that of the two layer network. Both the hidden layers had 5 neurons with ReLU activation for optimal results. Training the network, in this case, required more iterations as there were more number of parameter to be tuned than the two

layer network. The optimal number of neurons for both hidden layers were chosen accordingly as suggested in research article *Review on Methods to Fix Number of Hidden Neurons in Neural Networks* by Gnana Sheela et al. Once again, best results were obtained using uniform distribution of weights with $learning_rate=0.01$ and 40000 iterations. The following combination yielded the highest accuracy of 91.43% with a minimum accuracy of 80% and an average accuracy of 86%. As seen in the figure below,

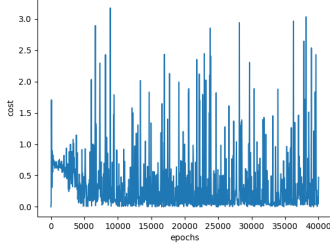


Figure 3: Plot of Loss vs Epochs for 40000 iterations with $lr = 10^{-2}$

a considerably high accuracy was observed even though the loss had significant fluctuations. This is due to the high learning rate provided. Lesser fluctuations were observed when the learning rate was decreased to 0.0005. But to compensate the training, the number of iterations had to increased significantly to around 1000000 to achieve an accuracy of 91.43%

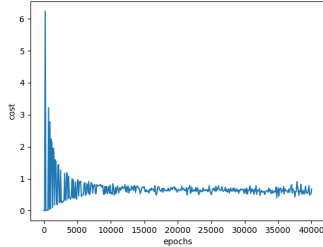


Figure 4: Plot of Loss vs Epochs for 40000 iterations with $lr = 5.10^{-4}$

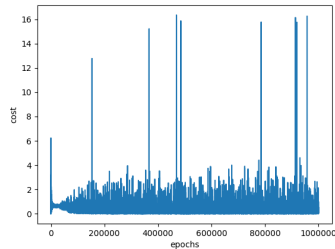


Figure 5: Plot of Loss vs Epochs for 1000000 iterations with $lr = 5.10^{-4}$

Top metrics achieved by the model:

- Best testing accuracy: 91.43%
- Best testing recall: 94.66%
- Best testing precision: 89.30%
- Best testing fscore: 91.90%

6. Conclusion:

- In general, a three layer network performs better than a two layer network. Adding more layers/more neurons to the hidden layers provides more trainable parameters which help the network to fit the model in a better way. But the drawback of adding multiple layers/multiple neurons is that, the model requires more iterations to train. In this case, there is a higher chance that the model might overfit the data, and hence giving a low testing accuracy.
- A lower learning rate requires more iterations to converge, but conversely, using a higher learning rate might lead to fluctuation in the loss/exploding gradient.
- Different initialization of weights leads to convergence of the model at different local minima (initializations mentioned above).
- Different activation functions for the hidden layers provides drastically varying results. We have used ReLU (Rectified Linear) which is by and the far, the de-facto standard activation used for hidden layers. Other activations like tanh or sigmoid gave very low accuracy as compared to that of ReLU. ReLU suffers from the *dying neuron problem*, in which the output of a neuron might be 0 for most of the training (i.e. that neuron is not contributing to the network), and hence we tried using leaky-ReLU, but the accuracy was found to more or less the same.

[results on next page]

7. Additional Work:

- **SGD Momentum and Decay**

$$M_t = \beta M_{t-1} + (1 - \beta) \nabla E$$

$$W = W - \eta M_t$$

- **RMSprop**

$$V_t = \beta V_{t-1} + (1 - \beta) (\nabla E)^2$$

$$W = W - \eta \frac{\nabla E}{\sqrt{V_t} + \epsilon}$$

- **Adam**

$$M_t = \beta M_{t-1} + (1 - \beta_1) \nabla E$$

$$V_t = \beta V_{t-1} + (1 - \beta_2) (\nabla E)^2$$

$$\hat{M}_t = \frac{M_t}{1 - \beta_1^t}; \hat{V}_t = \frac{V_t}{1 - \beta_2^t}$$

$$W = W - \eta \frac{\hat{M}_t}{\sqrt{\hat{V}_t} + \epsilon}$$

8. Results:

| training metrics: | training metrics: |
|---------------------------------------|---------------------------------------|
| tp = 508, tn = 479, fp = 111, fn = 70 | tp = 544, tn = 402, fp = 188, fn = 34 |
| final accuracy: 84.50342465753424 | final accuracy: 80.9931506849315 |
| final recall: 87.88927335640139 | final recall: 94.11764705882352 |
| final precision: 82.06785137318255 | final precision: 74.31693989071039 |
| final fscore: 84.87886382623225 | final fscore: 83.05343511450381 |

| testing metrics: | testing metrics: |
|--------------------------------------|-------------------------------------|
| tp = 133, tn = 123, fp = 19, fn = 17 | tp = 144, tn = 107, fp = 35, fn = 6 |
| final accuracy: 87.67123287671232 | final accuracy: 85.95890410958904 |
| final recall: 88.66666666666667 | final recall: 96.0 |
| final precision: 87.5 | final precision: 80.44692737430168 |
| final fscore: 88.0794701986755 | final fscore: 87.53799392097265 |

2-Layer NN using uniform distribution

| training metrics: | training metrics: |
|---------------------------------------|--------------------------------------|
| tp = 552, tn = 453, fp = 137, fn = 26 | tp = 521, tn = 495, fp = 95, fn = 57 |
| final accuracy: 86.0445205479452 | final accuracy: 86.98630136986301 |
| final recall: 95.50173010380622 | final recall: 90.138408304449827 |
| final precision: 80.11611030478954 | final precision: 84.57792207792207 |
| final fscore: 87.13496448303077 | final fscore: 87.26968174204356 |

| testing metrics: | testing metrics: |
|-------------------------------------|-------------------------------------|
| tp = 143, tn = 111, fp = 31, fn = 7 | tp = 141, tn = 126, fp = 16, fn = 9 |
| final accuracy: 86.98630136986301 | final accuracy: 91.43835616438356 |
| final recall: 95.33333333333334 | final recall: 94.0 |
| final precision: 82.18390804597702 | final precision: 89.80891719745223 |
| final fscore: 88.27160493827162 | final fscore: 91.85667752442997 |

3-Layer NN using uniform distirbution

| training metrics: | training metrics: |
|--|---------------------------------------|
| tp = 393, tn = 432, fp = 158, fn = 185 | tp = 489, tn = 480, fp = 110, fn = 89 |
| final accuracy: 70.63356164383562 | final accuracy: 82.96232876712328 |
| final recall: 67.99307958477509 | final recall: 84.60207612456747 |
| final precision: 71.32486388384754 | final precision: 81.63606010016694 |
| final fscore: 69.6191319751993 | final fscore: 83.09260832625318 |

| testing metrics: | testing metrics: |
|--------------------------------------|--------------------------------------|
| tp = 117, tn = 101, fp = 41, fn = 33 | tp = 130, tn = 115, fp = 27, fn = 20 |
| final accuracy: 74.65753424657534 | final accuracy: 83.9041095890411 |
| final recall: 78.0 | final recall: 86.66666666666667 |
| final precision: 74.0506329113924 | final precision: 82.80254777070064 |
| final fscore: 75.97402597402596 | final fscore: 84.69055374592834 |

2-Layer NN using normal distirbution

| training metrics: | training metrics: |
|---------------------------------------|--|
| tp = 479, tn = 460, fp = 130, fn = 99 | tp = 468, tn = 470, fp = 120, fn = 110 |
| final accuracy: 80.39383561643835 | final accuracy: 80.3082191780822 |
| final recall: 82.8719723183391 | final recall: 80.96885813148789 |
| final precision: 78.65353037766832 | final precision: 79.59183673469387 |
| final fscore: 80.70766638584666 | final fscore: 80.27444253859348 |

| testing metrics: | testing metrics: |
|--------------------------------------|--------------------------------------|
| tp = 131, tn = 112, fp = 30, fn = 19 | tp = 128, tn = 113, fp = 29, fn = 22 |
| final accuracy: 83.21917808219177 | final accuracy: 82.53424657534246 |
| final recall: 87.33333333333333 | final recall: 85.33333333333334 |
| final precision: 81.36645962732919 | final precision: 81.52866242038218 |
| final fscore: 84.2443729903537 | final fscore: 83.38762214983714 |

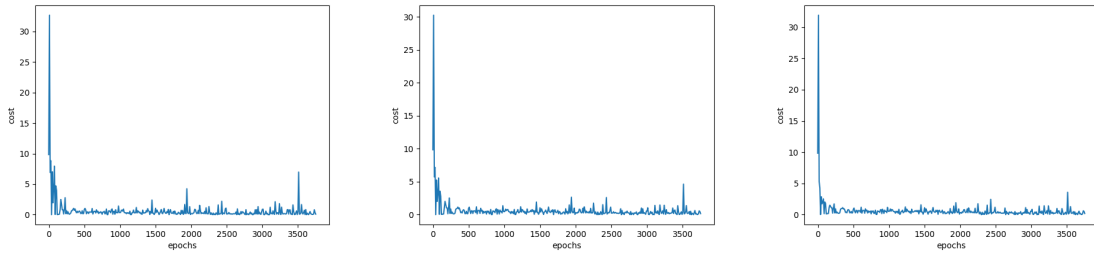
3-Layer NN using normal distirbution

| training metrics: | training metrics: |
|--------------------------------------|--------------------------------------|
| tp = 519, tn = 492, fp = 98, fn = 59 | tp = 505, tn = 504, fp = 86, fn = 73 |
| final accuracy: 86.5582191780822 | final accuracy: 86.38698630136986 |
| final recall: 89.79238754325259 | final recall: 87.37024221453287 |
| final precision: 84.11669367909238 | final precision: 85.44839255499154 |
| final fscore: 86.86192468619247 | final fscore: 86.39863130881095 |

| testing metrics: | testing metrics: |
|-------------------------------------|--------------------------------------|
| tp = 141, tn = 126, fp = 16, fn = 9 | tp = 138, tn = 127, fp = 15, fn = 12 |
| final accuracy: 91.43835616438356 | final accuracy: 90.75342465753424 |
| final recall: 94.0 | final recall: 92.0 |
| final precision: 89.80891719745223 | final precision: 90.19607843137256 |
| final fscore: 91.85667752442997 | final fscore: 91.0891089108911 |

3-Layer NN using uniform distirbution with momentum and decay

9. Additional Results:



Loss vs Epochs for Adam, RMSprop and Momentum respectively

| | | |
|---|---|---|
| <pre>training metrics: tp = 492, tn = 516, fp = 74, fn = 86 final accuracy: 86.3013698630137 final recall: 85.12110726643598 final precision: 86.92579505300353 final fscore: 86.01398601398601 testing metrics: tp = 129, tn = 130, fp = 12, fn = 21 final accuracy: 88.6986301369863 final recall: 86.0 final precision: 91.48936170212765 final fscore: 88.65979381443297</pre> | <pre>training metrics: tp = 529, tn = 483, fp = 107, fn = 49 final accuracy: 86.64383561643835 final recall: 91.52249134948097 final precision: 83.17610062893081 final fscore: 87.14991762767708 testing metrics: tp = 141, tn = 121, fp = 21, fn = 9 final accuracy: 89.72602739726028 final recall: 94.0 final precision: 87.03703703703704 final fscore: 90.38461538461539</pre> | <pre>training metrics: tp = 528, tn = 486, fp = 104, fn = 50 final accuracy: 86.81506849315068 final recall: 91.34948096885813 final precision: 83.54430379746836 final fscore: 87.27272727272728 testing metrics: tp = 140, tn = 122, fp = 20, fn = 10 final accuracy: 89.72602739726028 final recall: 93.33333333333333 final precision: 87.5 final fscore: 90.3225806451613</pre> |
|---|---|---|

3-Layer NN with Adam, RMSprop and Momentum respectively for 3750 epochs