Ishaan Chawla

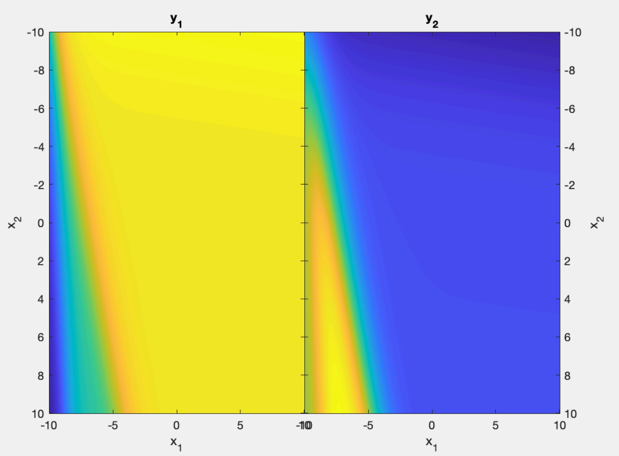
Prof. Chunming Wang

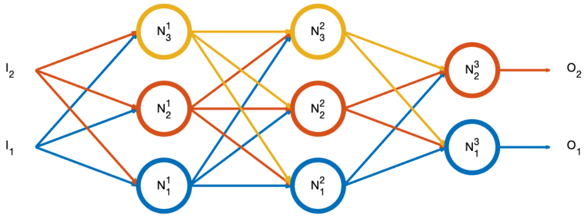
Math 467

3 October 2022

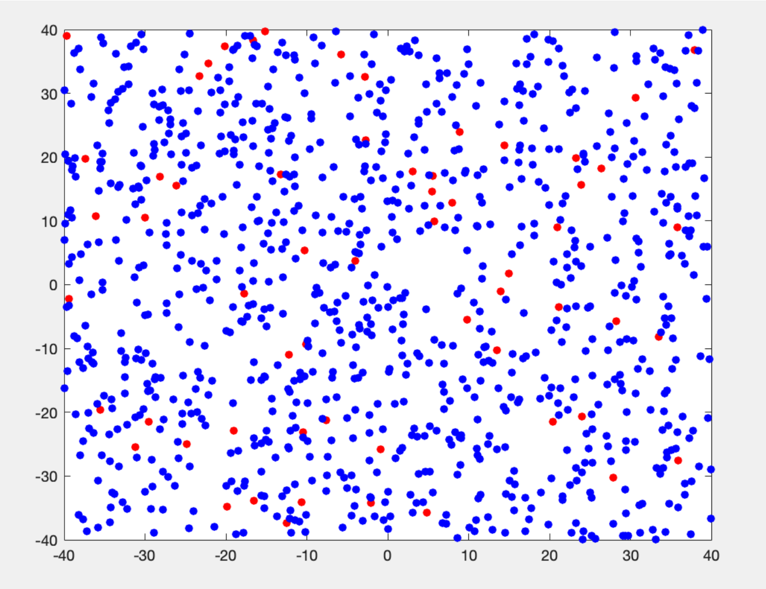
Project 1

1. For the first task, I duplicated the test code and modified line 7's call to the createNetwork function input to (2, [3,3,2]). I did the same on line 62, except modified that one to (2, [3,3,1]). I further modified the input for getData to use my Student ID number, 2827439557. The resulting figures were as follows:

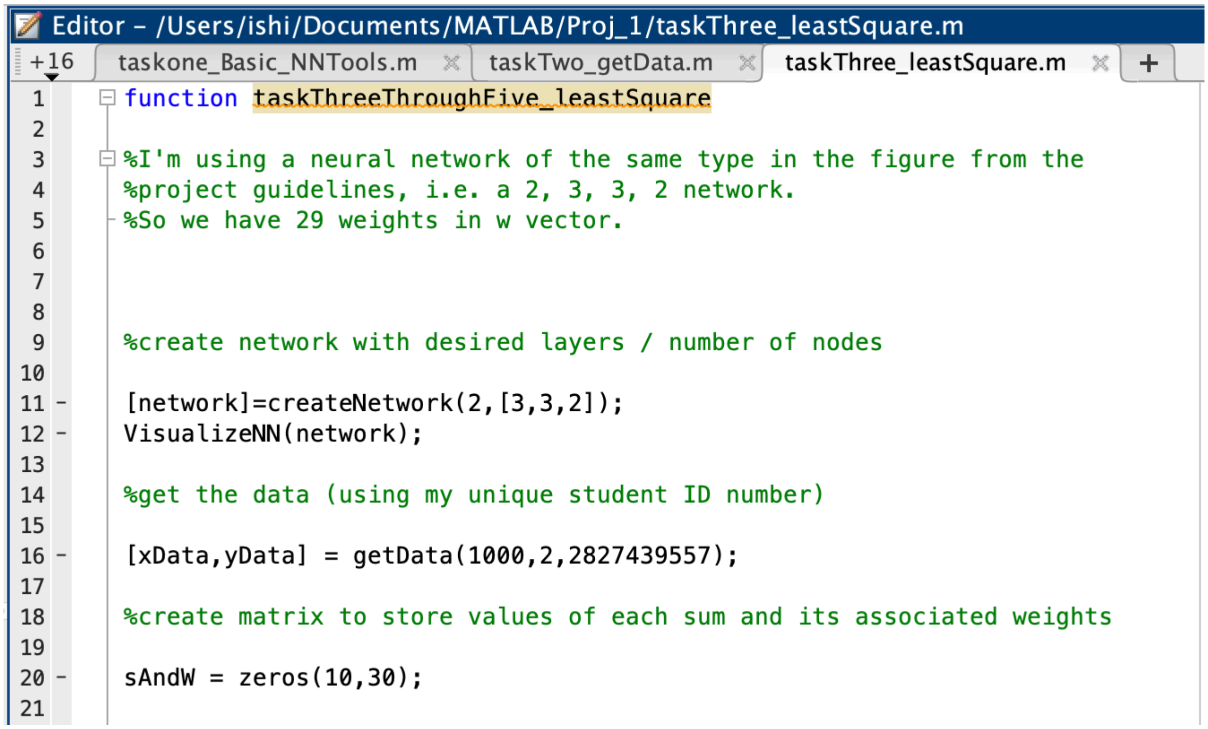




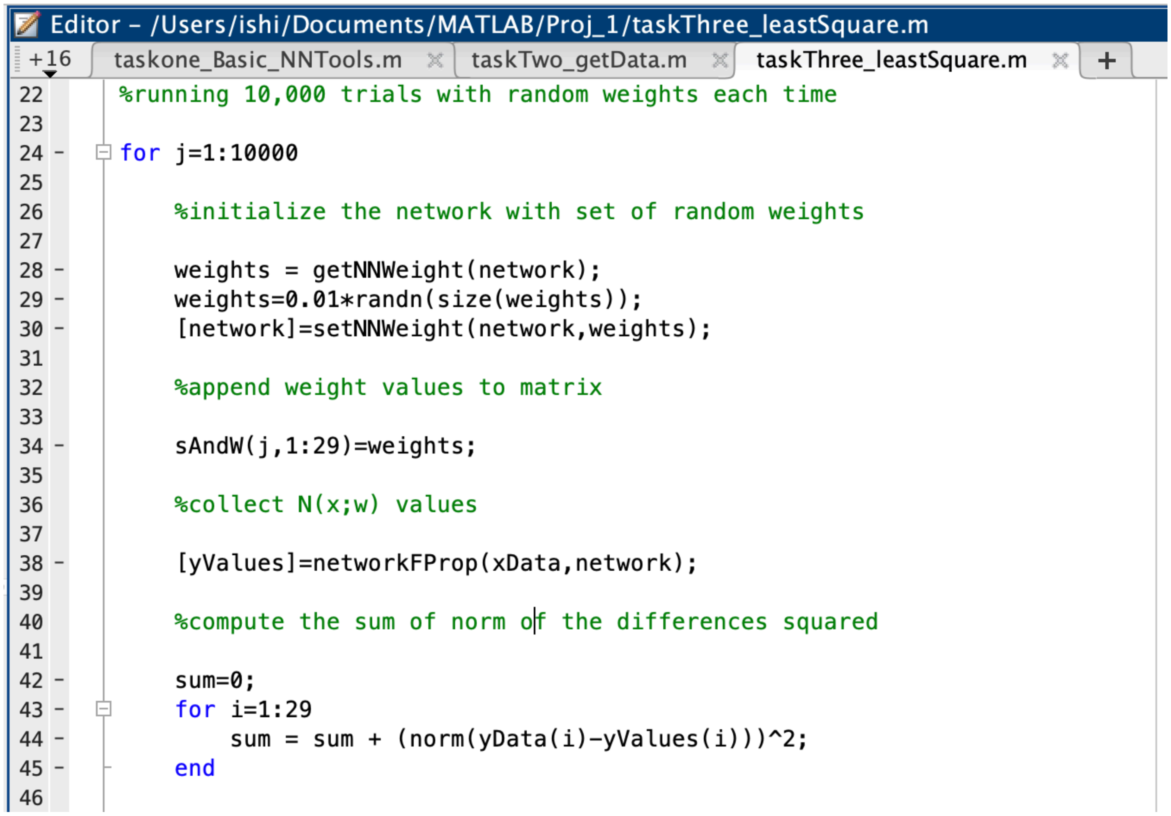
2. Similar to the first task, I copied the test code but replaced the last variable with my student ID number. The resulting plot:



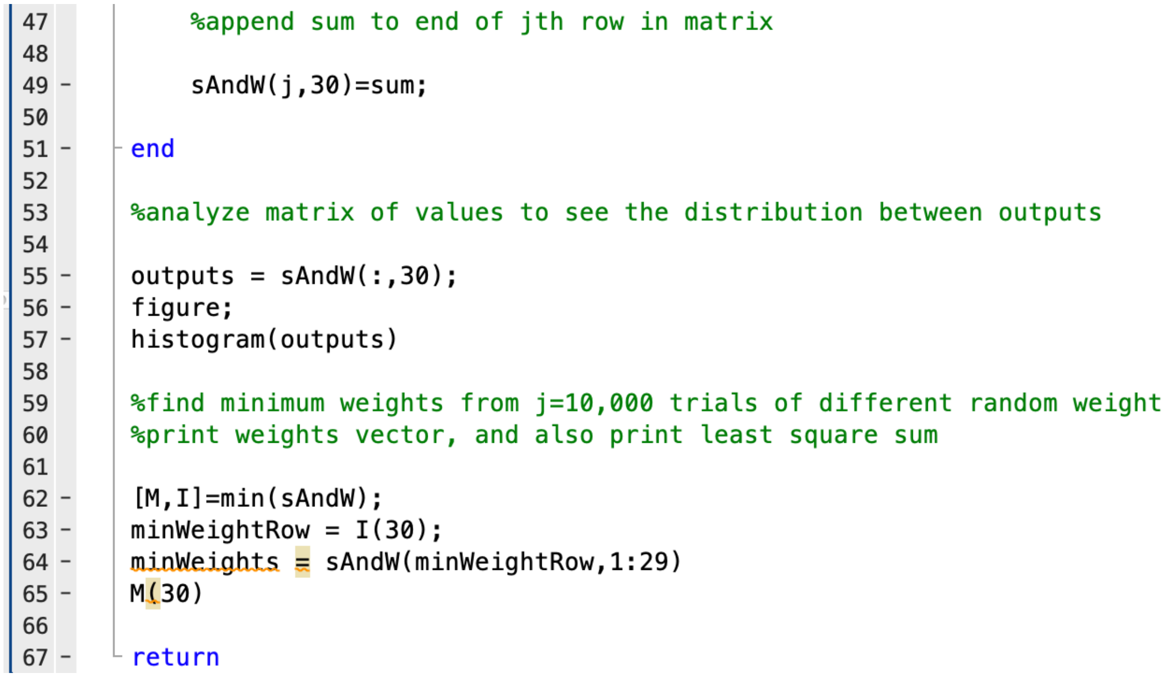
3. Since tasks 3 through 5 are cumulative, I combined them into one coded routine. Here is all of my code, with provided explanations:


Editor – /Users/ishi/Documents/MATLAB/Proj_1/taskThree_leastSquare.m
+16
taskone_Basic_NNTools.m
taskTwo_getData.m
taskThree_leastSquare.m
+

```
function taskThreeThroughFive_leastSquare

%I'm using a neural network of the same type in the figure from the
%project guidelines, i.e. a 2, 3, 3, 2 network.
%So we have 29 weights in w vector.



%create network with desired layers / number of nodes

[network]=createNetwork(2,[3,3,2]);
VisualizeNN(network);

%get the data (using my unique student ID number)

[xData,yData] = getData(1000,2,2827439557);

%create matrix to store values of each sum and its associated weights

sAndW = zeros(10,30);

```

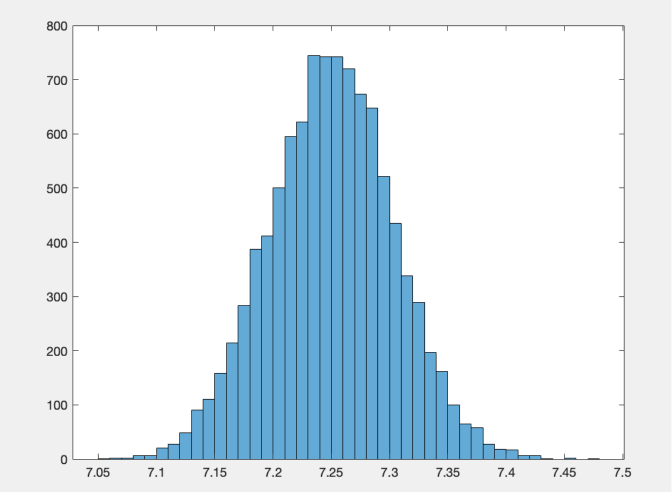

Editor - /Users/ishi/Documents/MATLAB/Proj_1/taskThree_leastSquare.m
+16
taskone_Basic_NNTools.m
taskTwo_getData.m
taskThree_leastSquare.m
+

```
%running 10,000 trials with random weights each time

for j=1:10000

    %initialize the network with set of random weights

    weights = getNNWeight(network);
    weights=0.01*randn(size(weights));
    [network]=setNNWeight(network,weights);

    %append weight values to matrix

    sAndW(j,1:29)=weights;

    %collect N(x;w) values

    [yValues]=networkFProp(xData,network);

    %compute the sum of norm of the differences squared

    sum=0;
    for i=1:29
        sum = sum + (norm(yData(i)-yValues(i)))^2;
    end

```



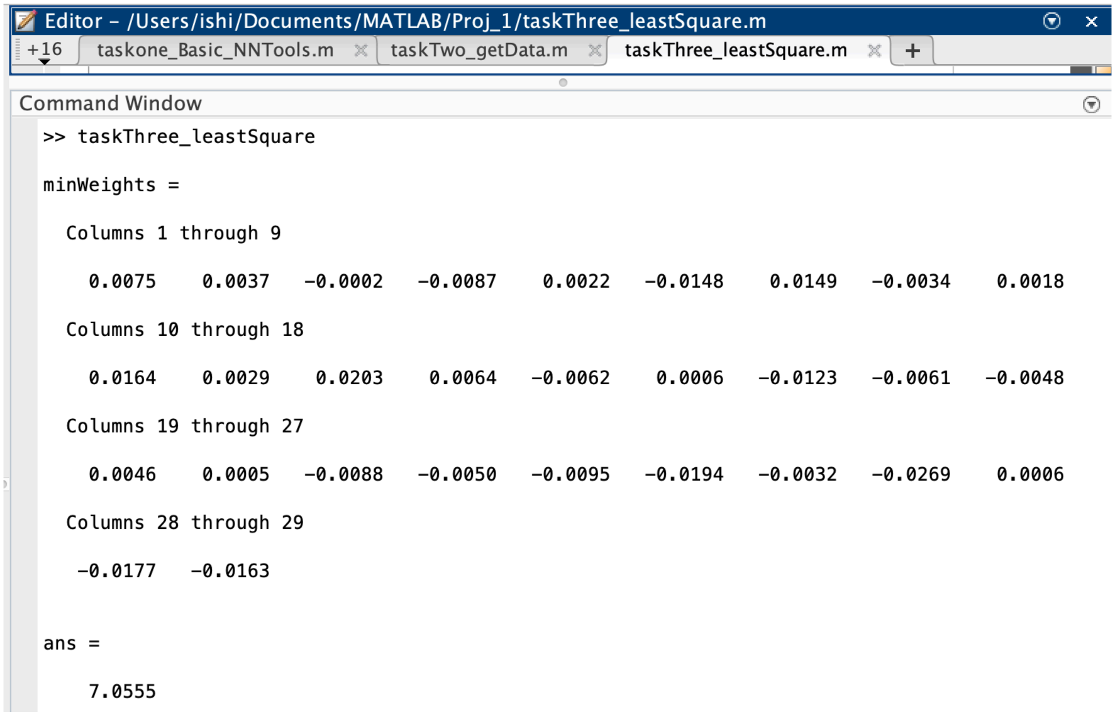
```
    %append sum to end of jth row in matrix

    sAndW(j,30)=sum;

end

%analyze matrix of values to see the distribution between outputs

outputs = sAndW(:,30);
figure;
histogram(outputs)

%find minimum weights from j=10,000 trials of different random weight
%print weights vector, and also print least square sum

[M,I]=min(sAndW);
minWeightRow = I(30);
minWeights = sAndW(minWeightRow,1:29)
M(30)

return
```

Here is the outputted histogram showing the distribution of the performance function (least square sums) over 10,000 trials of randomly generate weight vectors:



As can be seen, the distribution is close to a normal distribution. Most values tend to lie in the range of 7.2-7.3, with minimum close to 7.

Finally, here is the output of the minimum sum value, 7.0555, along with its associated weight vector.



```
>> taskThree_leastSquare

minWeights =

  Columns 1 through 9

    0.0075    0.0037   -0.0002   -0.0087    0.0022   -0.0148    0.0149   -0.0034    0.0018

  Columns 10 through 18

    0.0164    0.0029    0.0203    0.0064   -0.0062    0.0006   -0.0123   -0.0061   -0.0048

  Columns 19 through 27

    0.0046    0.0005   -0.0088   -0.0050   -0.0095   -0.0194   -0.0032   -0.0269    0.0006

  Columns 28 through 29

   -0.0177   -0.0163


ans =

    7.0555
```