78bc4225 **Report for Homework 5** CS1420 – Spring 2018 April 21, 2018

1 Problem 1

```
—– LINEAR REGRESSION —–
Average Training Loss: 0.5403729394828254
Average Testing Loss: 0.6598453517957842
—– 1-Layer NN —–
Average Training Loss: 0.5449476651961953
Average Testing Loss: 0.663365600887563
—– 2-Layer NN —–
('Average Training Loss:', 0.4709101923698204)
('Average Testing Loss:', 0.5752831256573131)
```

There is no big difference between linear regression and 1-layer NN because 1-layer NN is just apply neuron weight on each attribute, which is indeed just fitting a linear function to the dataset, the same as linear regression. However, 2-layer NN has better result than both linear regression and 1-Layer NN. This is because 2-layer NN includes the activation function, which can capture the non-linearity in the data.

2 Problem 2

The parameter I used for 2NN is learning rate 0.01, hidden size:10 and epochs 25. Adding several hidden size will increase the performance but keep increasing hidden neurons will eventually stop to increase accuracy. Adding epoch helps the model to converge to minimum poin. However, once the model is converged, adding more epoch will also not increase accuracy. Decreasing learning rate can help the model to find the minimum point but it slows down the convergence speed.

3 Problem 3

```
Linear:
('Average Training Loss:', 0.72125077414546612)
('Average Testing Loss:', 0.87022854763276591)
Step:
('Average Training Loss:', 0.70530064269216253)
('Average Testing Loss:', 0.82767301361321388)
Relu
('Average Training Loss:', 0.68213413915303522)
('Average Testing Loss:', 0.84750598511254027)
```

From the result, we can see linear activation has the worst performance and Relu activation has the best performance. This is because, linear activation doesn't have any effect on the first layer output so the network is still linear. Relu works the best because it's a non-linear function and different from step function, relu preserves the value of first layer output if it's greater than 1.