Machine Learning Homework 1

Ting-Yin Chen
Department of Applied Mathematics
National Chung Hsing University
Taichung, Taiwan
ti841130@gmail.com

I. Introduction

In this report, I will derive the forward and backward schemes for the regression problem to predict house dataset.

There are two parts: Frist, demonstrating the forward propagation and backward propagation which the 3-layer MLP has. Second, experimental results implemented by "PROOF" part.

II. Proof

A. Forward Propagation

Here, I'm talking about 3-layer MLP. The framework of 3-layer MLP is Fig.1.

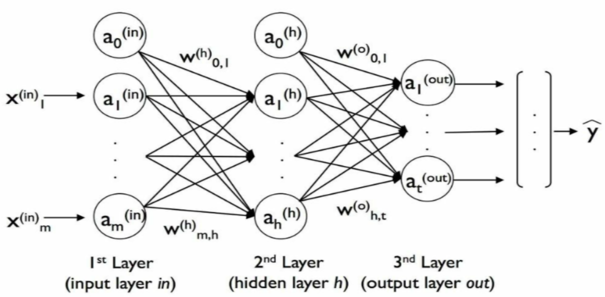

Fig. 1. framework of 3-layer MLP.

There are three layers in the framework. First layer is input layer. Second layer is hidden layer. Third layer is output layer.

Suppose $X_m^{(in)}$ is input data and it has m units. Input layer has m units. Hidden layer has h units. Output layer has t units. $\hat{y}$ is target.

Since each unit in the hidden layer is connected to all units in the input layers, first, calculate the activation unit of the hidden layer $a_j^{(h)}$ when $j = 1$ as follows:

$$z_1^{(h)} = [a_1^{(in)} a_2^{(in)} ... a_m^{(in)}]^T [w_{0,1}^{(h)} w_{1,1}^{(h)} ... w_{m,1}^{(h)}]$$

where $a_j^{(i)}$: output value/ vector/ tensor of j-th unit of the i-th layer; $w_{k,j}^{(i)}$: weights (the parameter to be learned) of the k-th unit to j-th of i-th layer; $z_j^{(i)}$: affine transformations of given inputs for j-th unit of i-th layer.

Thus, hidden layer value ($j = 1$):

$$a_1^{(h)} = \phi(z_1^{(h)})$$

where ϕ: activation function, I use $\phi = 1/(1 + e^{-z})$ in this report.

For purposes of code efficiency and readability, vectorizing implementation above(hidden layer has h units):

$$\begin{aligned}\overrightarrow{z}^{(h)} &:= [z_1^{(h)} z_2^{(h)} ... z_h^{(h)}]^T \\ &= [w_1^{(h)} w_2^{(h)} ... w_h^{(h)}]^T \overrightarrow{a}^{(in)} \\ &= (\mathbf{W}^{(h)})^T \overrightarrow{a}^{(in)}\end{aligned}$$

Thus, hidden layer vector:

$$\overrightarrow{a}^{(h)} = \phi(\overrightarrow{z}^{(h)})$$

And therefore,

$$\mathbf{Z}^{(h)} = (\mathbf{W}^{(h)})^T \mathbf{A}^{(in)}$$

$$\mathbf{A}^{(h)} = \phi(\mathbf{Z}^{(h)})$$

Similarly, for output layer:

$$\mathbf{Z}^{(out)} = (\mathbf{W}^{(out)})^T \mathbf{A}^{(h)}$$

$$\mathbf{A}^{(out)} = \phi(\mathbf{Z}^{(out)})$$

B. Backward Propagation

My cost function(MSE):

$$J(W) = \sum_i ||\hat{y}^{(i)} - y^{(i)}||^2$$

where i: i-th layer; $\hat{y}^{(i)}$: the predict output of i-th layer; $y^{(i)}$: the true value of i-th layer.

Here are the framework of backward for 3-layer MLP in Fig.2.

To minimize the cost function $J(W)$, we need to calculate the partial derivative of the parameters W with respect to each weight of each layer of the network:

$$\frac{\partial}{\partial W_{k,j}^{i}} J(W)$$

For output layer:

$$\begin{aligned}J(W) &= \sum_i ||\hat{y}^{(i)} - y^{(i)}||^2 \\ &= Tr((\hat{Y} - Y)^T (\hat{Y}^{-} Y)) \\ &= Tr((A^{(out)} - Y)^T (A^{(out)} - Y))\end{aligned}$$

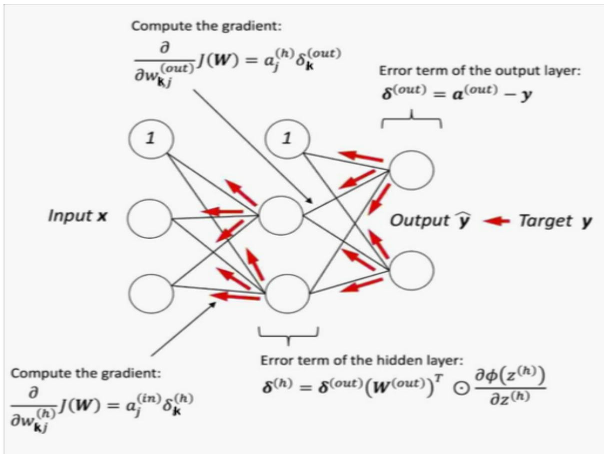

Fig. 2. framework of 3-layer MLP.

Take partial derivative of the parameters W:

$$\frac{\partial}{\partial W^i_{k,j}} J(W) = \frac{\partial}{\partial W^i_{k,j}} Tr((A^{(out)} - Y)^T (A^{(out)} - Y))$$
$$= \frac{\partial Tr((A^{(out)} - Y)^T (A^{(out)} - Y))}{(A^{(out)} - Y)} \frac{(A^{(out)} - Y)}{\partial W^i_{k,j}}$$
$$= 2(A^{(out)} - Y)) \frac{(A^{(out)} - Y)}{\partial A^{(out)}} \frac{\partial A^{(out)}}{\partial W^i_{k,j}}$$
$$= 2(A^{(out)} - Y)) \frac{\partial Z^{(out)}}{\partial W^i_{k,j}}$$
$$= 2(A^{(out)} - Y)) \frac{\partial (W^{(out)}_{k,j} A^{(h)})}{\partial W^i_{k,j}}$$
$$= 2A^{(h)}_j \delta^{(out)}_k, \delta^{(out)}_k = A^{(out)} - Y$$

For hidden layer:

$$\frac{\partial J(W)}{\partial W^{(h)}_{k,j}} = \frac{\partial Tr((A^{(out)} - Y)^T (A^{(out)} - Y))}{\partial (A^{(out)} - Y)} \frac{\partial (A^{(out)} - Y)}{\partial W^{(h)}_{k,j}}$$
$$= 2(A^{(out)} - Y) \frac{\partial (A^{(out)} - Y)}{\partial A^{(out)}} \frac{\partial A^{(out)}}{\partial (W^{(h)}_{k,j})}$$
$$= 2(A^{(out)} - Y) \frac{\partial ((W^{(out)}_{k,j})^T A^{(h)})}{\partial A^{(h)}} \frac{\partial A^{(h)}}{\partial (W^{(h)}_{k,j})}$$
$$= 2(A^{(out)} - Y) \frac{\partial ((W^{(out)}_{k,j})^T A^{(h)})}{\partial A^{(h)}} \frac{\partial \phi(Z^{(h)})}{\partial Z^{(h)}} \frac{\partial Z^{(h)}}{\partial W^{(h)}_{k,j}}$$
$$= 2(A^{(out)} - Y) \frac{\partial ((W^{(out)}_{k,j})^T A^{(h)})}{\partial A^{(h)}} \cdot$$
$$[\phi(Z^{(h)}) \odot (C - \phi(Z^{(h)}))] \frac{\partial ((W^{(h)}_{k,j})^T A^{(in)})}{\partial W^{(h)}_{k,j}}$$
$$= 2A^{(in)}_j \delta^{(h)}_k, \delta^{(h)}_k = W^{(out)}_{k,j} \delta^{(out)}_k \odot \frac{\partial \phi(Z^{(h)})}{\partial Z^{(h)})}$$

Finally, having updating rule:

$$\frac{\partial}{\partial W^{(out)}_{k,j}} J(W) = a^{(h)}_j \delta^{(out)}_k$$
$$\frac{\partial}{\partial W^{(h)}_{k,j}} J(W) = a^{(in)}_j \delta^{(h)}_k$$

III. Experiment

A. Dataset

In this report, I'm going to use house dataset. It has 506 datas, for 506 rows and 14 columns. The following describes the dataset columns:

* CRIM - per capita crime rate by town
* ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
* INDUS - proportion of non-retail business acres per town.
* CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
* NOX - nitric oxides concentration (parts per 10 million)
* RM - average number of rooms per dwelling
* AGE - proportion of owner-occupied units built prior to 1940
* DIS - weighted distances to five Boston employment centres
* RAD - index of accessibility to radial highways
* TAX - full-value property-tax rate per $10,000
* PTRATIO - pupil-teacher ratio by town
* B - 1000(Bk - 0.63)$\hat{2}$ where Bk is the proportion of blacks by town
* LSTAT - % lower status of the population
* MEDV - Median value of owner-occupied homes in $1000's

I will choose the top thirteen columns for my X and the last one for my y which I want to predict.

B. Results

* data are split into two parts: 70% for training data, 30% for testing data.
* do normalize for X data
* data are shuffle for each epoch
* epoch = 1500
* eta=0.001

1) experiment 1: For test1(Fig.3.) and test2 (Fig.4.), the difference is I normalize the y data or not. The hidden units are same for 30 and minibatch are also same for 10.

2) experiment 2: For test3(Fig.5.) and test4 (Fig.6.), the difference is the numbers of hidden units. Both y are normalized and minibatch are also same for 10.

3) experiment 3: For test5(Fig.7.) and test6 (Fig.8.), the difference is the sizes of minibatch. Both y are normalized and the hidden units are same for 30.

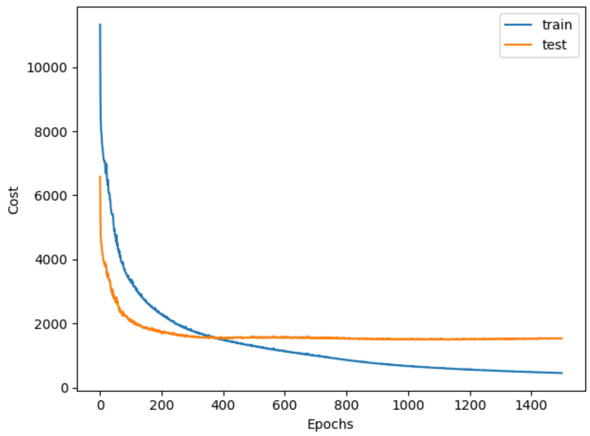

Fig. 3. without normalize y.

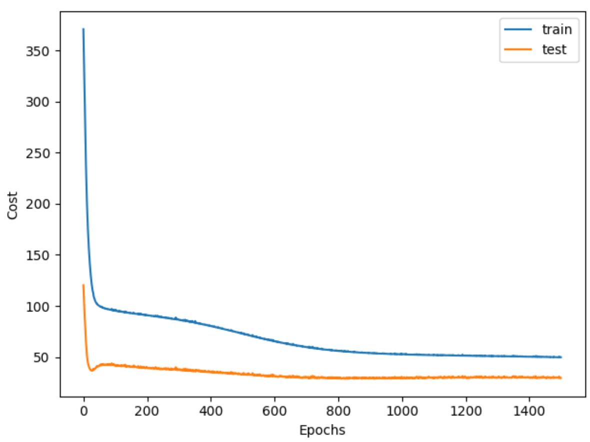

Fig. 4. with normalize y.

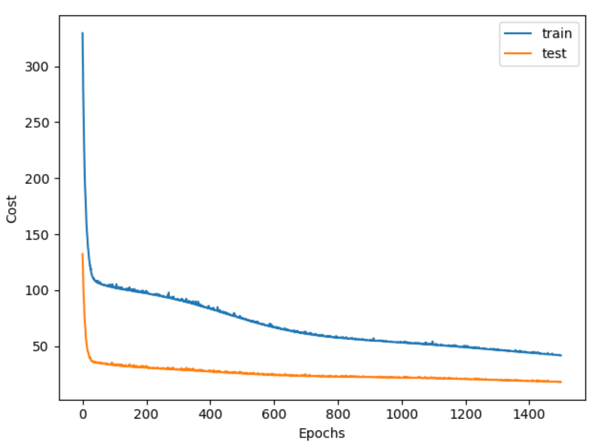

Fig. 5. 50 hidden units.

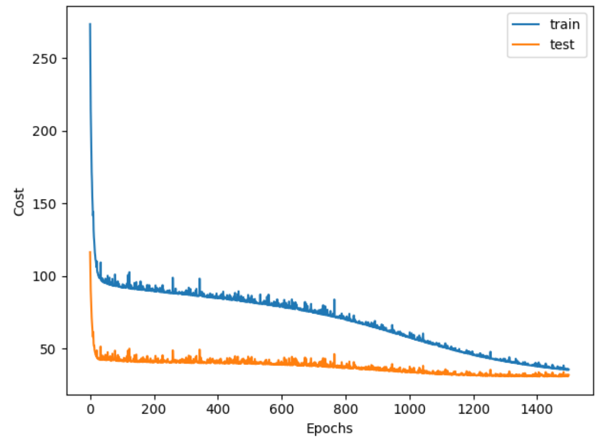

Fig. 6. 100 hidden units.

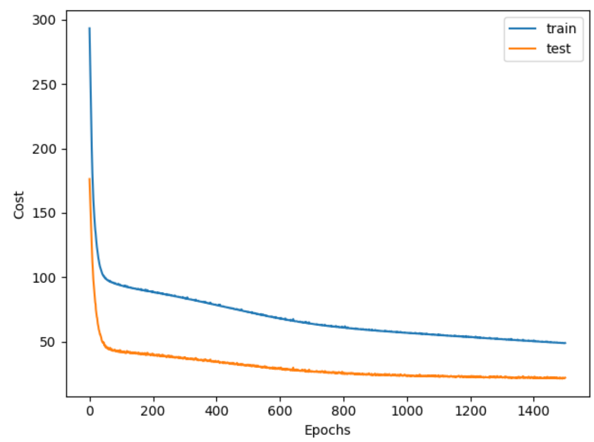

Fig. 7. 5 minibatch.

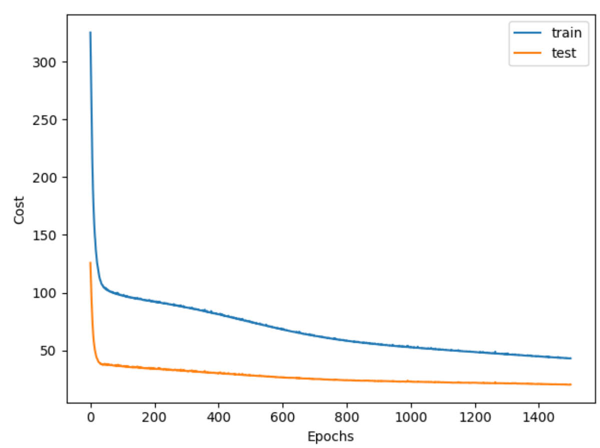

Fig. 8. 20 minibatch.