Homework 6 - MATH 4322

Cathy Poliak

Spring 2023

Instructions

1. Due date: April 25, 2023
2. Answer the questions fully for full credit.
3. Scan or Type your answers and submit only one file. (If you submit several files only the recent one uploaded will be graded).
4. Preferably save your file as PDF before uploading.
5. Submit in Blackboard under Contents → Homework → Homework 6.
6. These questions are based on the Neural Networks lectures.
7. The information in the gray boxes are R code that you can use to answer the questions.

Problem 1

You are given:

- n data samples $\mathbf{x}_i = (x_{1,i}, \cdots, x_{p,i}), i = 1, \ldots, n$
- n corresponding to true responses (or labels) $y_i, i = 1, \ldots, n$.

and asked to train a single linear neuron "network" to approximate function $f(.)$ such that $f(\mathbf{x}_i) = y_i, i = 1, \ldots, n$. Provided the train steps for your "network" by answering the following questions.

a) What is the formula to calculate an output $\hat{y}_i$ from an input $\mathbf{x}_i$? What are the model parameters in that formula?

b) What criteria do we need to optimize in order to estimate the model parameters?

c) What is the name of the method used to optimize this criteria in case you do not have access to an analytical solution?

Answers

a) $\hat{y}_i = \sum_{i=1}^{p} w_i x_i + b$, the model parameters are:

- Weights of inputs: $w_1, w_2, \ldots, w_p$.
- bias: b.

b) We need to optimize the sum of the squared residuals of predicted values $\hat{y}_i$, when compared to the true values y_i:

$$\frac{1}{2}\sum_{i=1}^{p}(\hat{y}_i - y_i)^2$$

Or if your data samples are fed one at a time, then when the i^{th} sample is fed, we need to optimize:

$$\frac{1}{2}(\hat{y}_i - y_i)^2$$

c) Gradient decent.

Problem 2

Presume that for a single linear neuron model with input variables $x_1, \ldots, x_5$, you are given the following parameter values:

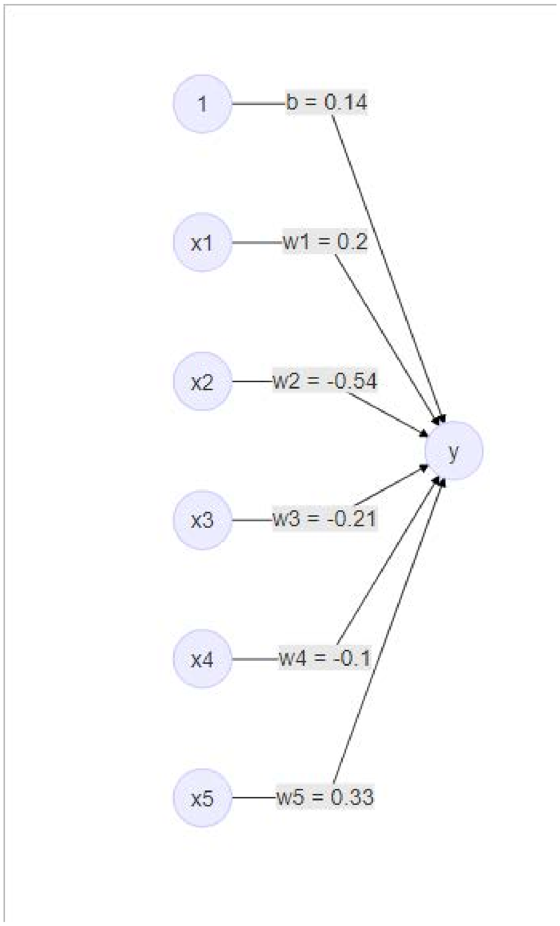
- weights: $w_1 = 0.2, w_2 = -0.54, w_3 = -0.21, w_4 = -0.1, w_5 = 0.33$,
- bias: $b = 0.14$.

a) Draw a mathematical model of this linear neuron that takes an arbitrary input vector $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5)$.

b) Calculate the linear neuron output for the case of $x_1 = 4, x_2 = -3, x_3 = 7, x_4 = 5, x_5 = -1$. Show your work.

Answer

a) Diagram



b) $\hat{y} = 0.14 + 0.2(4) - 0.54(-3) - 0.21(7) - 0.1(5) + 0.33(-1) = 0.26$

Problem 3

You are given an artificial neural network (ANN) of linear neurons with

- Input layer of two neurons: x_1, x_2
- Fully-connected hidden layer of three neurons: h_1, h_2, h_3
- One output neuron, y.

The following weight matrices are provided:

1) Between input & hidden layer:

		Hidden		
		h_1	h_2	h_3
	1 (bias)	-0.3	0.5	0.5
Input	x_1	0.6	-0.4	0.5
	x_2	-0.7	-0.3	0.2

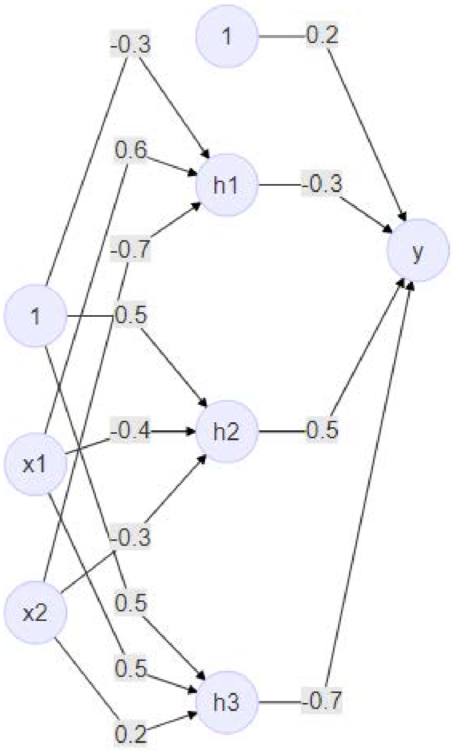
2) Between hidden & output layer:

		Output
		y
	1 (bias)	0.2
Hidden	h_1	-0.3
	h_2	0.5
	h_3	-0.7

a) Draw this ANN as was done in lecture slides.
b) Calculate the output of this ANN for the case of $x_1 = 10, x_2 - -5$. Show work.

Answers

a) Diagram



b) Calculations:

$$\hat{h}_1 = -0.3 + 0.6 * 10 - 0.7 * (-5) = 9.2$$
$$\hat{h}_2 = 0.5 - 0.4 * 10 - 0.3 * (-5) = -2$$
$$\hat{h}_3 = 0.5 + 0.5 * 10 + 0.2 * -5 = 4.5$$
$$\hat{y} = 0.2 - 0.3 * 9.2 + 0.5 * -2 - 0.7 * 4.5 = -6.71$$

Problem 4

- We want to predict the 'medv' value based on the input of the other thirteen variables.
- We will run a regression neural network for the `Boston` data set.
- We will split the data into training/testing by a 70/30 split.

a) Type and run the following in `R`.

```
library(neuralnet)
library(MASS)

data = Boston #renaming the Boston data set to "data"
summary(data)
```

```
##       crim                zn             indus            chas        
##  Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000  
##  1st Qu.: 0.08205   1st Qu.:  0.00   1st Qu.: 5.19   1st Qu.:0.00000  
##  Median : 0.25651   Median :  0.00   Median : 9.69   Median :0.00000  
##  Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917  
##  3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000  
##  Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000  
##       nox               rm             age              dis        
##  Min.   :0.3850   Min.   :3.561   Min.   :  2.90   Min.   : 1.130  
##  1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100  
##  Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207  
##  Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795  
##  3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188  
##  Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127  
##       rad              tax           ptratio          black       
##  Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  0.32  
##  1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38  
##  Median : 5.000   Median :330.0   Median :19.05   Median :391.44  
##  Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67  
##  3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23  
##  Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90  
##      lstat            medv      
##  Min.   : 1.73   Min.   : 5.00  
##  1st Qu.: 6.95   1st Qu.:17.02  
##  Median :11.36   Median :21.20  
##  Mean   :12.65   Mean   :22.53  
##  3rd Qu.:16.95   3rd Qu.:25.00  
##  Max.   :37.97   Max.   :50.00
```

What is the mean of `age`? What is the mean of `ptratio`?

Answer

mean(age) = 68.57, mean(ptratio) = 19.05

b) Normalizing data

- It is recommended to **normalize** (or scale, or standardize, either works) features in order for all the variables to be on the same scale.

- With normalization, data units are eliminated, allowing you to easily compare data from different locations.
- This avoids unnecessary results or difficult training processes resulting in algorithm convergence problems.
- There are different methods for scaling the data.
- The **z-normalization**
$$x_{scale} = \frac{x - \bar{x}}{s}$$
- The **min-max scale**
$$x_{scale} = \frac{x - x_{min}}{x_{max} - x_{min}}$$
- And so forth
- The function in `R` is scale(x,center = ,scale =)
- For this example we will use the min-max method to get all the scaled data in the range [0, 1].
- In order to scale we need to find the minimum and maximum value for each of the columns in the data set. To do this we use the apply function.
- The `apply` function returns a vector or an array or a list of values obtained by **applying** a function to margins of an array or matrix.

- Type and run the following:

```
max_data = apply(data,2,max)
#2=columns, we are getting the maximum value from each column
min_data = apply(data,2,min)
data_scaled = scale(data, center = min_data, scale = max_data - min_data)
head(data_scaled)
```

```
##           crim   zn      indus chas       nox        rm       age       dis
## 1 0.0000000000 0.18 0.06781525    0 0.3148148 0.5775053 0.6416066 0.2692031
## 2 0.0002359225 0.00 0.24230205    0 0.1728395 0.5479977 0.7826982 0.3489620
## 3 0.0002356977 0.00 0.24230205    0 0.1728395 0.6943859 0.5993821 0.3489620
## 4 0.0002927957 0.00 0.06304985    0 0.1502058 0.6585553 0.4418126 0.4485446
## 5 0.0007050701 0.00 0.06304985    0 0.1502058 0.6871048 0.5283213 0.4485446
## 6 0.0002644715 0.00 0.06304985    0 0.1502058 0.5497222 0.5746653 0.4485446
##          rad        tax   ptratio     black      lstat      medv
## 1 0.00000000 0.20801527 0.2872340 1.0000000 0.08967991 0.4222222
## 2 0.04347826 0.10496183 0.5531915 1.0000000 0.20447020 0.3688889
## 3 0.04347826 0.10496183 0.5531915 0.9897373 0.06346578 0.6600000
## 4 0.08695652 0.06679389 0.6489362 0.9942761 0.03338852 0.6311111
## 5 0.08695652 0.06679389 0.6489362 1.0000000 0.09933775 0.6933333
## 6 0.08695652 0.06679389 0.6489362 0.9929901 0.09602649 0.5266667
```

What is the scaled value of the first observation for `medv`?

Answer 0.42222

c) Now we can split the data into training and testing data sets. We will use the 70/30 split

```
set.seed(10)
index = sample(1:nrow(data), round(0.7*nrow(data)))
train_data = as.data.frame(data_scaled[index,])
test_data = as.data.frame(data_scaled[-index,])
dim(train_data)
```

```
## [1] 354  14
```

How many observations do we have in the training data set?

Answer 354

d) Type and run the following

```r
set.seed(1)
net_data = neuralnet(medv ~ ., data = train_data,
                     hidden = 10, linear.output = TRUE)

plot(net_data)
```

Apply the test data set to determine the MSE

```r
predict_net = predict(net_data,test_data)
predict_net_start = predict_net*(max(data$medv) - min(data$medv)) + min(data$medv)
test_data_start = test_data$medv*(max(data$medv) - min(data$medv)) + min(data$medv)
sum((predict_net_start - test_data_start)^2)/nrow(test_data)
```

```
## [1] 15.37819
```

What is the test MSE for this model?

Answer I got 15.37819, this is $3921.5.

e) Let us compare this test MSE to the linear regression model. Type and run the following:

```r
lm.boston = lm(medv ~ ., data = data, subset = index )
summary(lm.boston)
```

```
##
## Call:
## lm(formula = medv ~ ., data = data, subset = index)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.981  -3.029  -0.529   1.882  25.349
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.073e+01  6.167e+00   6.606 1.53e-10 ***
## crim        -9.253e-02  5.445e-02  -1.700 0.090140 .
## zn           4.810e-02  1.737e-02   2.769 0.005936 **
## indus       -6.504e-04  8.068e-02  -0.008 0.993573
## chas         2.910e+00  1.124e+00   2.589 0.010045 *
## nox         -1.948e+01  4.636e+00  -4.202 3.39e-05 ***
## rm           3.203e+00  5.057e-01   6.333 7.60e-10 ***
## age          1.022e-02  1.627e-02   0.628 0.530150
## dis         -1.481e+00  2.454e-01  -6.033 4.20e-09 ***
## rad          2.948e-01  8.303e-02   3.551 0.000439 ***
## tax         -1.227e-02  4.786e-03  -2.564 0.010775 *
## ptratio     -9.461e-01  1.678e-01  -5.638 3.61e-08 ***
## black        9.767e-03  3.464e-03   2.819 0.005094 **
## lstat       -5.385e-01  6.229e-02  -8.646  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 4.989 on 340 degrees of freedom
## Multiple R-squared:  0.7062, Adjusted R-squared:  0.6949
## F-statistic: 62.85 on 13 and 340 DF,  p-value: < 2.2e-16
```

```
test = data[-index,]
predict_lm = predict(lm.boston,test)
sum((predict_lm - test$medv)^2)/nrow(test)
```

```
## [1] 17.77379
```

What is the training MSE for the linear model?

Answer I got 17.77379 which is higher than the neural network.