Ch 10.1: Neural Nets

Lecture 21 - CMSE 381

Michigan State University
::
Dept of Computational Mathematics, Science & Engineering

Wed, April 10, 2024

Announcements

Last time:

- SVM

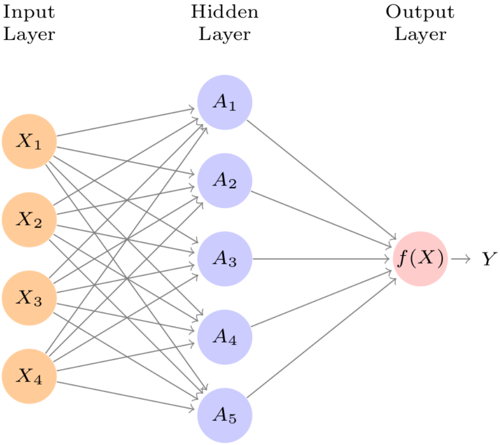
This lecture:

- Feed Forward Neural Nets

Section 1

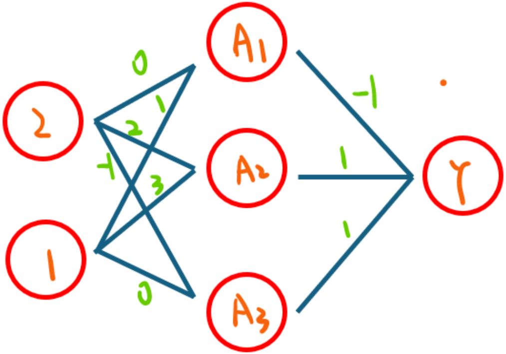
Neural Nets

Feed Forward Neural Network



- Input layer is starting data
- Each arrow means taking combo of those values with weights to get next value
- Here there is one hidden layer then the output
- Get to pick how many hidden units K, here we have $K = 5$

Starter: A linear network



Calculate the prediction for the new data $(2, 1)$

$A_1 = 2\cdot 0 + 1\cdot 1 = 1$

$A_2 = 2\cdot 2 + 1\cdot 3 = 5$

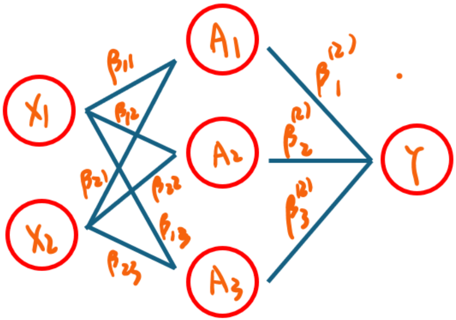
$A_3 = 2\cdot(-1) + 1\cdot 0 = -2$

$Y = -1\cdot A_1 + 1\cdot A_2 + 1\cdot A_3 = -1 + 5 - 2 = 2$

predicted Y is 2

$$\beta = \begin{pmatrix} 0 & 2 & -1 \\ 1 & 3 & 0 \end{pmatrix} \qquad \beta^{(2)} = \begin{pmatrix} -1 \\ 1 \\ 1 \end{pmatrix}$$

Training data $x^1 = (0,1), y^1 = 1,\ x^2 = (1,1), y^2 = -1.$



$$\hat{y}_1 = \beta_1^{(2)} A_1 + \beta_2^{(2)} A_2 + \beta_3^{(2)} A_3$$

$$= \beta_1^{(2)} (\beta_{11} X_1^1 + \beta_{21} X_2^1) + \beta_2^{(2)} (\beta_{12} X_1^1 + \beta_{22} X_2^1)$$

$$+ \beta_3^{(2)} (\beta_{31} X_1^1 + \beta_{32} X_2^1)$$

$$= \beta_1^{(2)} \beta_{21} + \beta_2^{(2)} \beta_{22} + \beta_3^{(2)} \beta_{32}$$

$$\hat{y}_2 = \beta_1^{(2)} (\beta_{11} + \beta_{21}) + \beta_2^{(2)} (\beta_{21} + \beta_{22}) + \beta_3^{(2)} (\beta_{31} + \beta_{32})$$

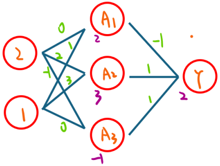
$$\hat{\beta}_{ij}, \hat{\beta}_K^{(2)} = \arg\min_{\beta_{ij}, \beta_K^{(2)}} (y_1 - \hat{y}_1)^2 + (y_2 - \hat{y}_2)^2 = (1 - \hat{y}_1)^2 + (1 + \hat{y}_2)^2$$

$i = 1, 2.\ j = 1, 2, 3.\ K = 1, 2, 3$

Adding bias and activation

Computing Y for $(0,1)$

$$A_k = g(\beta_{k0} + \sum_{j=1}^{p} \beta_{kj} X_j), \quad y = f(X) = \beta_0^{(2)} + \sum_{k=1}^{K} \beta_k^{(2)} A_k$$



$$\begin{cases} \beta_{10} = 2 \\ \beta_{20} = 3 \\ \beta_{30} = -1 \end{cases} : \text{bias}, \quad \beta_0^{(2)} = 2$$

$$g(z) = (z)_+ = \begin{cases} z, & \text{if } z \geq 0 \\ 0, & \text{else} \end{cases}$$

activation function

A different example

- Draw the diagram for a neural net with input data points with $p = 3$ (i.e., (X_1, X_2, X_3)) and two units in the hidden layer.
- Using the β and $\beta^{(2)}$ matrices, what is the output predicted Y for the point $(2, 0, 1)$?

$$\beta = \begin{pmatrix} 1 & 0 & -2 & 2 \\ -3 & 1 & 0 & -1 \end{pmatrix} \qquad \beta^{(2)} = \begin{pmatrix} -1 \\ -2 \\ 1 \end{pmatrix}$$

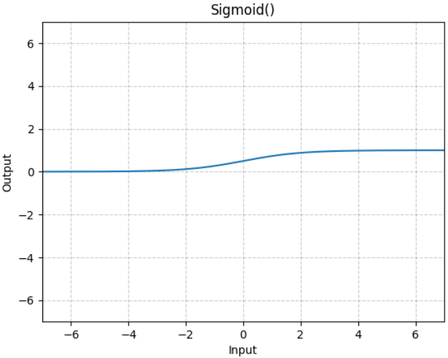
- Use the activation function

$$g(z) = (z)_+ = \begin{cases} 0 & \text{if } z < 0 \\ z & \text{else.} \end{cases}$$

Choices for activation function

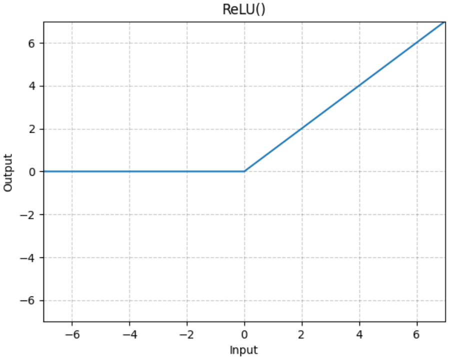
Sigmoid:

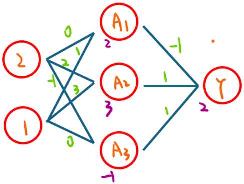
$$g(z) = \frac{e^z}{1+e^z} = \frac{1}{1+e^{-z}}$$



ReLU: Rectified linear unit

$$g(z) = (z)_+ = \begin{cases} 0 & \text{if } z < 0 \\ z & \text{else.} \end{cases}$$





$$\begin{cases} \beta_{10} = 2 \\ \beta_{20} = 3 \\ \beta_{30} = -1 \end{cases} \text{ : bias }, \quad \beta_0^{(2)} = 2$$

$$g(z) = (z)_+ = \begin{cases} z, & \text{if } z \geq 0 \\ 0, & \text{else} \end{cases}$$

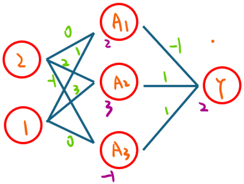
activation function

$$A_k = g(\beta_{k0} + \sum_{j=1}^{p} \beta_{kj} Xj),$$

$$A = g(\mathbf{W} \cdot \mathbf{X}) \qquad \mathbf{X}^T = (1 \; X_1 \; X_2 \; \cdots \; X_p)$$

Caclulate out the matrix multiplication using the matrices at left to show that the equations are the same.

Matrix version: Output



$$\begin{cases} \beta_{10} = 2 \\ \beta_{20} = 3 \\ \beta_{30} = -1 \end{cases} : \text{bias}, \quad \beta_0^{(2)} = 2$$

$$g(z) = (z)_+ = \begin{cases} z, & \text{if } z \geq 0 \\ 0, & \text{else} \end{cases}$$

activation function

$$f(X) = \beta_0^{(2)} + \sum_{k=1}^{K} \beta_k^{(2)} A_k$$

$$Y = \beta^{(2)} \cdot \mathbf{A} \qquad \mathbf{A}^T = (1 \; A_1 \; A_2 \; \cdots \; A_K)$$

Now what?

Choose parameters by minimizing RSS, $\sum_{i=1}^{n}(y_i - f(x_i))^2$

Chosen in advance:

- Number of layers (more on that next lecture)
- Number of hidden units
- Activation function $g(z)$

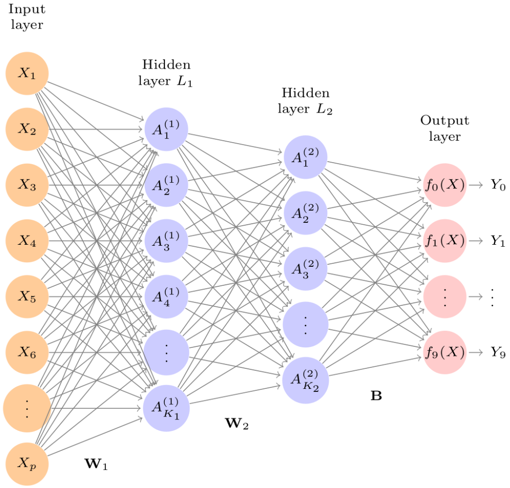
Found by fitting the data:

- $\mathbf{W}$
- $\beta^{(2)}$

Section 2

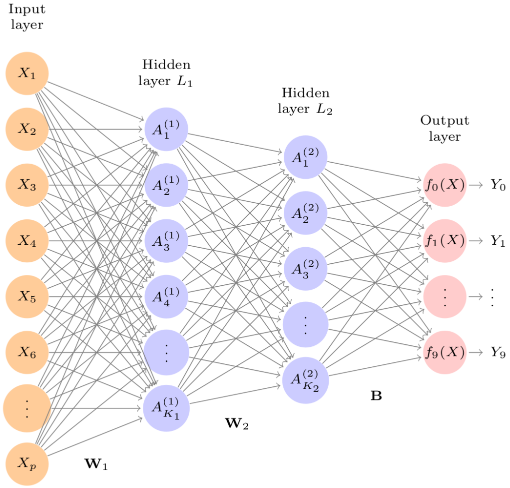
Multilayer Neural Networks

Multiple layers



- Include more layers
- Can pick number of units per layer
- Each layer is linear combinations of previous

Hidden layers

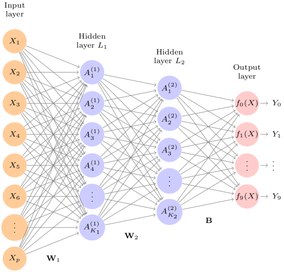


$$
\begin{aligned}
A_k^{(1)} &= h_k^{(1)}(X) \\
&= g(w_{k0}^{(1)} + \textstyle\sum_{j=1}^{p} w_{kj}^{(1)} X_j)
\end{aligned}
$$

$$
\begin{aligned}
A_\ell^{(2)} &= h_\ell^{(2)}(X) \\
&= g(w_{\ell 0}^{(2)} + \textstyle\sum_{k=1}^{K_1} w_{\ell k}^{(2)} A_k^{(1)})
\end{aligned}
$$

- Two hidden layers. L_1 has 256 units; L_2 has 128
- 10 output variables due to class labeling

More on that architecture

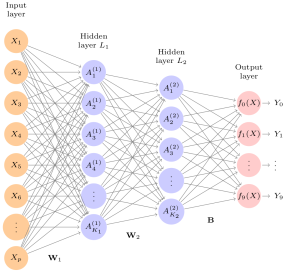


$$
\begin{aligned}
A_k^{(1)} &= h_k^{(1)}(X) \\
&= g(w_{k0}^{(1)} + \textstyle\sum_{j=1}^{p} w_{kj}^{(1)} X_j)
\end{aligned}
$$

$$
\begin{aligned}
A_\ell^{(2)} &= h_\ell^{(2)}(X) \\
&= g(w_{\ell 0}^{(2)} + \textstyle\sum_{k=1}^{K_1} w_{\ell k}^{(2)} A_k^{(1)})
\end{aligned}
$$

- Superscript denotes layer
- W_1 denotes entire matrix of weight
- In this setting, size is $785 \times 256 = 200,960$ values
- 785 instead of 784 to involve intercept term (called bias in this literature)

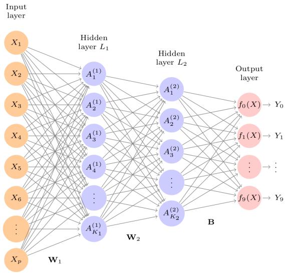
Matrix version: First layer



$$\begin{aligned} A_k^{(1)} &= h_k^{(1)}(X) \\ &= g(w_{k0}^{(1)} + \textstyle\sum_{j=1}^{p} w_{kj}^{(1)} X_j) \end{aligned}$$

$$A^{(1)} = g(\mathbf{W}^{(1)} \cdot \mathbf{X}) \qquad \mathbf{X}^T = (1\ X_1\ X_2\ \cdots\ X_p)$$

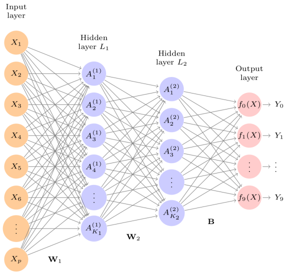
Matrix version: Second layer



$$\begin{aligned} A_\ell^{(2)} &= h_\ell^{(2)}(X) \\ &= g(w_{\ell 0}^{(2)} + \textstyle\sum_{k=1}^{K_1} w_{\ell k}^{(2)} A_k^{(1)}) \end{aligned}$$

$$A^{(2)} = g(\mathbf{W}^{(2)} \cdot \mathbf{A}) \qquad (\mathbf{A}^{(1)})^T = (1\ A_1^{(1)}\ A_2^{(1)}\ \cdots\ A_{K_1}^{(1)})$$

Matrix version: Last layer, first step

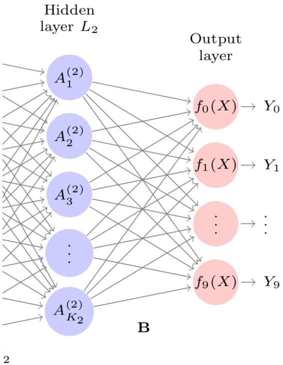


$$
\begin{aligned}
Z_m &= \beta_{m0} + \textstyle\sum_{\ell=1}^{K_2} \beta_{m\ell} h_\ell^{(2)}(X) \\
&= \beta_{m0} + \textstyle\sum_{\ell=1}^{K_2} \beta_{m\ell} A_\ell^{(2)},
\end{aligned}
$$

$$\mathbf{Z} = \beta \cdot \mathbf{A}$$

β is $M \times (K_2 + 1)$ matrix $\quad (\mathbf{A}^{(2)})^T = (1\ A_1^{(2)}\ A_2^{(2)}\ \cdots\ A_{K_2}^{(2)})$

The last column for classification: Softmax



$$f_m(X) = \Pr(Y = m|X) = \frac{e^{Z_m}}{\sum_{\ell=0}^{9} e^{Z_\ell}},$$

- Want this to act like a probability.
- Answer is to use the softmax activation function $f_m(X)$
- Values are non-negative
- Values sum to 1
- Return class with highest probability

An example

$$Z = \begin{pmatrix} 1 & 3 & -1 & 2 & 5 \end{pmatrix}$$

- $e^1 \ + \ e^3 \ + \ e^{-1} + e^2 + e^5 = 178.97$
- $f =$ $\begin{pmatrix} 0.01518 & 0.1122 & 0.00205 & 0.04128 & 0.82924 \end{pmatrix}$

MNIST

- Goal: Build a model to classify images into their correct digit class
- Each image has $p = 28 \cdot 28 = 784$ pixels
- Each pixel is grayscale value in [0,255]
- Data converted into column order
- Output represented by one-hot vector $Y = (Y_0, Y_1, \cdots, Y_9)$
- 60K training images, 10K test images