Machine Learning in Econometrics: Lecture 11

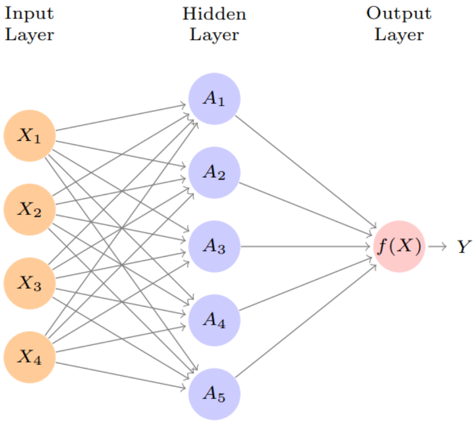
Instructor: Chaoyi Chen
NJE & MNB

November 14, 2023

- Neural networks became popular in the 1980s
- Along came SVMs, Random Forests and Boosting in the 1990s and Neural Networks took a back seat.
- Re-emerged around 2010 as Deep Learning.
- By 2020s very dominant and successful.
- Part of success due to vast improvements in computing power, larger training sets, and software

Topic 5: Single Layer Neural Network

$$
\begin{aligned}
f(X) &= \beta_0 + \sum_{k=1}^{K} \beta_k h_k(X) \\
&= \beta_0 + \sum_{k=1}^{K} \beta_k g(w_{k0} + \sum_{j=1}^{p} w_{kj} X_j).
\end{aligned}
$$



Topic 5: Terminology

- $f(X) = \beta_0 + \sum_{k=1}^{K} \beta_k h_k(X) = \beta_0 + \sum_{k=1}^{K} \beta_k g(w_{k_0} + \sum_{j=1}^{p} w_{k_j} X_j)$
- $A_k = h_k(X) = g(w_{k_0} + \sum_{j=1}^{p} w_{k_j} X_j)$ are called the activations in the hidden layer.
- $g(.)$ is called the activation function. Popular are the sigmoid and rectified linear
- Activation functions in hidden layers are typically nonlinear, otherwise the model collapses to a linear model.
- So the activations are like derived features — nonlinear transformations of linear combinations of the features.
- The model is typically fit by minimizing $\sum_{i=1}^{n} (y_i - f(x_i))^2$ (e.g. minimize SSR)

Topic 5: Sigmoid and Rectified Linear Function

- *Sigmoid* activation function

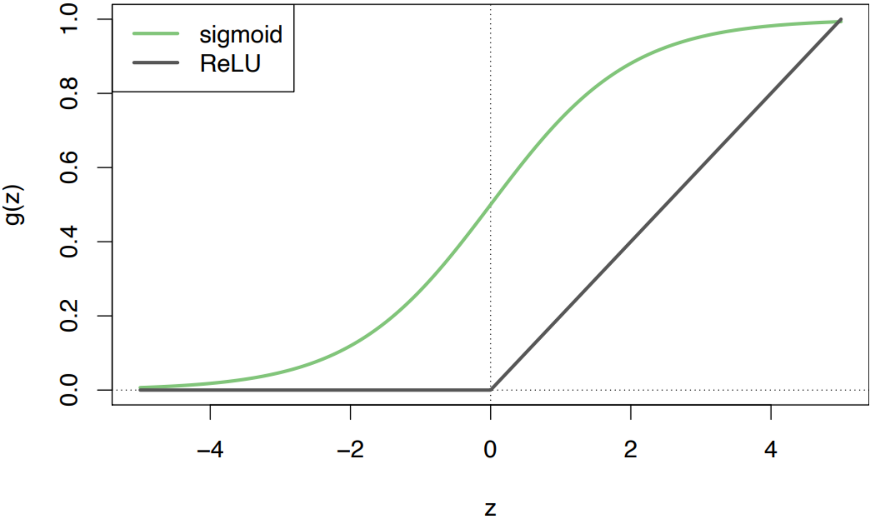
$$g(z) = \frac{e^z}{1+e^z} = \frac{1}{1+e^{-z}}$$

- Therefore, it is the same function used in logistic regression to convert a linear function into probabilities between zero and one
- *Rectifed linear unit* (ReLU) activation function

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

- ReLU activation can be computed and stored more efficiently than a sigmoid activation.

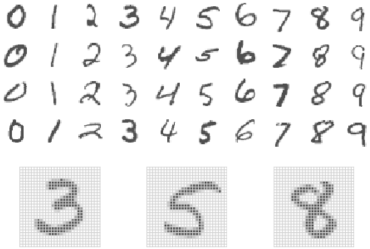
Topic 5: Activation Functions Figure



Topic 5: Multilayer Neural Networks

- Modern neural networks typically have more than one hidden layer, and often many units per layer
- In theory a single hidden layer with a large number of units has the ability to approximate most functions.
- However, the learning task of discovering a good solution is made much easier with multiple layers each of modest size.
- We will illustrate a large dense network on the `MNIST` handwritten digit dataset.

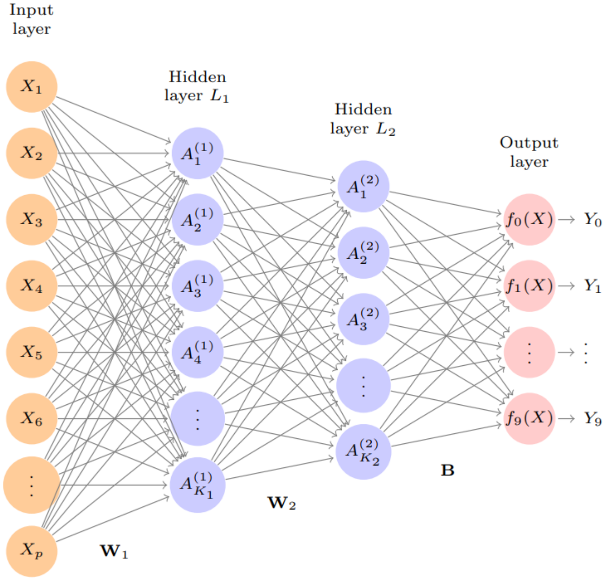
Topic 5: Example of MNIST



Handwritten digits
$28 \times\ 28$ grayscale images
$60K$ train, $10K$ test images
Features are the 784 pixel grayscale values $\in (0, 255)$
Labels are the digit class 0–9

- Goal: build a classifier to predict the image class.
- We build a two-layer network with 256 units at first layer, 128 units at second layer, and 10 units at output layer.
- Along with intercepts (called *biases*) there are 235,146 parameters (referred to as *weights*)

Topic 5: Neural Network Diagram with Two Hidden Layers



Topic 5: Remarks

- Figure in the preceding slide shows a multilayer network architecture that works well for solving the digit classification task.
- It has two hidden layers L_1 (256 units) and L_2 (128 units).
- It has ten output variables, rather than one. In this case, the ten variables really represent a single qualitative variable and so are quite dependent.
- The loss function used for training the network is tailored for the multiclass classification task.
- The first hidden layer is

$$A_k^{(1)} = h_k^1(X) = g\left(w_{k0}^{(1)} + \sum_{j=1}^{p} w_{kj}^{(1)} X_j\right), \text{ for } k = 1, \ldots, K_1$$

- The second hidden layer is

$$A_l^{(2)} = h_l^2(X) = g\left(w_{l0}^{(2)} + \sum_{k=1}^{K_1} w_{lk}^{(2)} A_k^{(1)}\right), \text{ for } l = 1, \ldots, K_2.$$

Topic 5: Details of Output Layer

- Let $Z_m = \beta_{m0} + \sum_{l=1}^{K_2} \beta_{ml} A_l^{(2)}$, $m = 0, 1, \ldots, 9$ be 10 linear combinations of activations at second layer.
- Output activation function encodes the softmax function

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

- We fit the model by minimizing the negative multinomial log-likelihood (or cross-entropy):

$$-\sum_{i=1}^{n} \sum_{m=0}^{9} y_{im} log(f_m(x_i)).$$

- $y_{im} = 1$ if true class for observation i is m. Otherwise, $y_{im} = 0$ - In the machine learning community, this is known as one-hot encoding.

Topic 5: Results

Method	Test Error
Neural Network + Ridge Regularization	2.3%
Neural Network + Dropout Regularization	1.8%
Multinomial Logistic Regression	7.2%
Linear Discriminant Analysis	12.7%

- Early success for neural networks in the 1990s.
- With so many parameters, regularization is essential.
- Some details of regularization and fitting will come later.
- Very overworked problem — best-reported rates are $< 0.5\%$!
- Human error rate is reported to be around 0.2%, or 20 of the 10K test images