Neural Networks

Prof. Sharon McNicholas

STATS 780/CSE 780

Introduction

- As with many other topics covered in an hour or two during this course, there are whole courses and entire books devoted to neural networks.

- Like PPR, a neural network is a non-linear statistical model.

- Originally developed as a model of the brian.

- A neural network is a two-stage model that can be used for regression or classification; again, we want to use $\mathbf{X}$ to predict Y.

The Idea

- Similar to Hastie et al. (2009)[a], we will consider classification using a neural network with one hidden layer (aka a single-layer perceptron).
- Suppose Y has K classes and take $\mathbf{X} = (X_1, \ldots, X_p)'$.
- For our purposes, consider binary variables $Y_1, \ldots, Y_K$.
- Introduce hidden units $Z_1, \ldots, Z_M$, where each Z_m is a linear combination of $X_1, \ldots, X_p$.
- Here is a visual representation of a neural network from Hastie et al. (2009); note that bias can be added to each unit in the hidden and output layers.

[a] Hastie, T., Tibshirani, R. and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Second Edition. Springer: NY.

Example of Neural Network

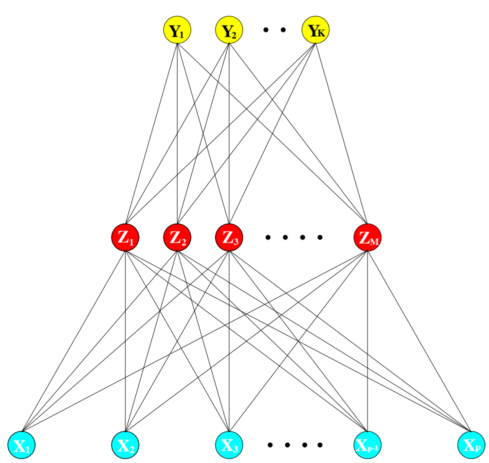

FIGURE 11.2. *Schematic of a single hidden layer, feed-forward neural network.*

The Idea contd.

- Similar to Hastie et al. (2009), we can describe the neural network as follows:
$$Z_m = \sigma(\alpha_{0m} + \boldsymbol{\alpha}_m'\mathbf{X}),$$
$$T_k = \beta_{0k} + \boldsymbol{\beta}_k'\mathbf{Z},$$
$$f_k(\mathbf{X}) = g_k(\mathbf{T}),$$
for $m = 1, \ldots, M$ and $k = 1, \ldots, K$, where $\mathbf{Z} = (Z_1, \ldots, Z_M)'$ and $\mathbf{T} = (T_1, \ldots, T_K)'$.
- The α_{0m}, $\boldsymbol{\alpha}_m$, β_{0k}, and $\boldsymbol{\beta}_k$ are unknown parameters called weights.
- $\sigma()$ is a (non-linear) transformation known as the activation function.
- $g_k(\mathbf{T})$ allows a transformation of $\mathbf{T}$ and is called the output function.

Activation Function

- Suppose $\sigma()$ is the identity function, then we have
$$Z_m = \alpha_{0m} + \boldsymbol{\alpha}_m'\mathbf{X}$$
and so a linear model.
- For $\sigma()$ non-linear, a neural network can be viewed as a non-linear generalization of a linear model.
- The sigmoid function is a popular choice for $\sigma()$, i.e.,
$$\sigma(v) = \frac{1}{1 - e^{-v}}.$$

Sigmoid Function (from Hastie et al., 2009)

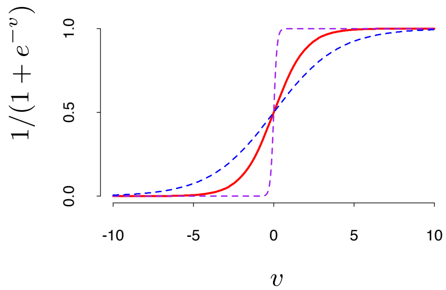

FIGURE 11.3. *Plot of the sigmoid function $\sigma(v) = 1/(1+\exp(-v))$ (red curve), commonly used in the hidden layer of a neural network. Included are $\sigma(sv)$ for $s = \frac{1}{2}$ (blue curve) and $s = 10$ (purple curve). The scale parameter s controls the activation rate, and we can see that large s amounts to a hard activation at $v = 0$. Note that $\sigma(s(v - v_0))$ shifts the activation threshold from 0 to v_0.*

Output Function

- For regression, $g_k(\mathbf{T})$ is usually chosen to be the identity function, i.e.,

$$g_k(\mathbf{T}) = T_k.$$

- For classification, the softmax function is common:

$$g_k(\mathbf{T}) = \frac{e^{T_k}}{\sum_{h=1}^{K} e^{T_k}}. \tag{1}$$

- For any T_k, (1) guarantees that $g_k(\mathbf{T}) > 0$, for all $k = 1, \ldots, K$, and

$$\sum_{k=1}^{K} g_k(\mathbf{T}) = 1.$$

Relationship with PPR

- Consider a neural network with one hidden layer — aka a single-layer perceptron.

- Looking at it from the viewpoint of a PPR, we can write

$$\begin{aligned} g_m(\boldsymbol{\omega}'_m\mathbf{X}) &= \beta_m\sigma(\alpha_{0m} + \boldsymbol{\alpha}'_m\mathbf{X}) \\ &= \beta_m\sigma(\alpha_{0m} + \|\boldsymbol{\alpha}_m\|\boldsymbol{\omega}'_m\mathbf{X}). \end{aligned} \tag{2}$$

- Here, $\boldsymbol{\omega}_m = \boldsymbol{\alpha}_m/\|\boldsymbol{\alpha}_m\|$.

- Another way to look at the relationship in (2) is

$$g_m(V_m) = \beta_m\sigma(\alpha_{0m} + \|\boldsymbol{\alpha}_m\|V_m).$$

Comments

- If there is no hidden layer, a (classification) neural network would be equivalent to multinomial logistic regression.

- There can be more than one hidden layer, which can be thought of as allowing a sort of hierarchy — aka a multi-layer perceptron.

- As Hastie et al. (2009, Sec. 11.5.3) explain, it is generally best to scale (standardize) the input (predictors).

- There is model fitting, and a lot of subtleties, to consider; extensive details are given (or referenced) in Hastie et al. (2009).

Comments contd.

- Like PPR, neural networks can work very well when prediction — and not modelling — is the goal.

- As Hastie et al. (2009) point out, neural networks (and PPR) generally work well in situations where there is high signal-to-noise ratio.

- In "competitions", neural networks are often amongst the best learning methods.

- Let's look at some neural network examples in R.