Logistic regression and artificial neural networks

Carmine-Emanuele Cella

October 16, 2015

1 Context

Logistic regression and artificial neural networks (ANN) are statistical learning models that provide a functional form f and parameter vector θ to express class membership probability (in a binary classification context)[1] of x as:

$$f_\theta(x) = \log\left(\frac{\mathcal{P}(1|x)}{\mathcal{P}(0|x)}\right). \tag{1}$$

The parameters are determined based on the data by maximum-likelihood estimation.

1.1 Logistic regression

The functional form for logistic regression is given by the dot product between data and parameters, plus some constants (biases): $f_\theta(x) = \theta \cdot x + b$. By adding and additional component 1 to all data vectors it is possible to avoid the constant, thus having $f_\theta(x) = \theta \cdot x$.

A binary logistic regression calculates the class membership probability for one of the two categories in a data set:

$$\mathcal{P}(1|x,\theta) = \frac{1}{1+e^{-f_\theta(x)}} \tag{2}$$

and $\mathcal{P}(0|x,\theta) = 1 - \mathcal{P}(1|x,\theta)$. The hyperplane of all points x satisfying $\theta \cdot x = 0$ forms the decision boundary between the two classes.

Computing the maximum likelihood estimation of the optimal parameters means maximizing $\prod_{i=1}^{N} \mathcal{P}(y_i|x_i,\theta)$.

1.2 Artificial neural networks

The functional form for ANN depends on the nonlinearity used and on the general architecture. For a typical ANN with 1 input layer, 1 hidden layer and 1 output layer it can be defined as $f_{\theta_h,\theta_o}(x) = b + \theta_o \cdot \sigma(c + \theta_h \cdot x)$ where b, c are biases, θ_o, θ_h are the parameters for output and hidden layer respectively and σ

[1] This approach can be nonetheless generalized to any number of classes.

is a nonlinearity. Also in this case it is possible to remove the biases by adding a component to all data points in any layer, thus having $f_{\theta_h,\theta_o}(x) = \theta_o \cdot \sigma(\theta_h \cdot x)$. Therefore, for a feedforward multi-layer network, the final output is given by:

$$\mathcal{P}(1|x, \theta_h, \theta_o) = \frac{1}{1 + e^{-\theta_o \cdot f_{\theta_h}(x)}}. \tag{3}$$

It is important to remark that in this context the output has a probabilistic form because of the constraints imposed in equation 1; namely, this is achived in ANN by embedding the output neurons into a function called *softmax*:

$$softmax(a) = \frac{e_i^a}{\sum_j e_j^a}. \tag{4}$$

Also in this case, the parameters are computed by maximum-likelihood estimation. While the functional forms for logistic regression and ANN models differ, **in this context, a network without hidden layers is actually identical to a logistic regression model**. The effect of the nonlinearity in the hidden layers is that the output of ANN can be a nonlinear function of the inputs; for this reason, in a classification context, the decision boundary can be nonlinear as well, making the model more flexible compared to logistic regression.

2 Parameter estimation

As stated above, for both logistic regression and ANN, the parameters are determined by maximizing the likelihood $\prod_{i=1}^{N} \mathcal{P}(y_i|x_i, \theta)$. From a computational standpoint, it is usually easier to minimize the negative log probability (also called loss-function):

$$L = -\sum_{i=1}^{N} log(\mathcal{P}(y_i|x_i, \theta)). \tag{5}$$

This is called *negative log likelihood criterion* and can be done with several optimization algorithms, from simple gradient descent to second-order methods.

It is possible to apply a regularization criterion to the parameters in the form $\|\theta\|_2^2 = \sum_i \theta_j^2$ (Frobœnius norm).

2.1 Backprogragation and gradient descent

A typical way used both in logistic regression and ANN for minimizing the negative log probability is by updating the parameters θ (backpropagation) by means of the applycation of a gradient descent in a stochastic way, thus only selecting random samples t during the calculation:

$$\theta \leftarrow \theta - \epsilon(2\lambda\theta + \nabla_\theta L(f_\theta(x^t), y^t)) \tag{6}$$

where L is the loss functin as described above, ϵ is a constant called *learning rate*, and λ is a regularization factor (for logistic regression, normally $\epsilon = 1, \lambda = 0$).