Linear Data Analysis
Classification - Logistic Regression

Cain Susko

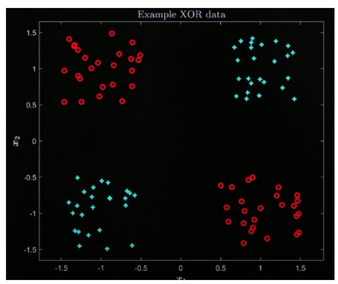
Queen's University
School of Computing

March 29, 2022

a Shortcomings of Perceptrons

Perceptrons as a singular or as a single 'layer' are unable to solve certain hyperplanes. A common pattern in which this occurs is Exclusive Or. This is caused by the fact that we use a binary activation function. To solve this, multi-layered networks would be needed but the method to train them was unknown.

Example if we had data which modeled an exclusive or truth table we would get:



Intuitively, it can be seen that there is no 2-dimensional line that separates the blue and red points.

Error Propogation

In order to resolve this issue, in 1985 Rumelhart et al, created a method to propagate labels through a multi-layered using a 'logistic activation function'. However this was a rediscovery of Paul Werbos' work from 1974.

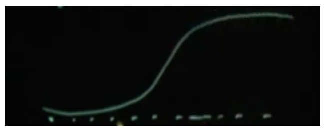
b Scores From Logistic Activation

Consider a single neuron with linear activation. This would be a weighted linear sum. Similarly a layers of linear activation neurons is weighted linear sum. This means many layers can only do as much as one neuron (with a linear activation function).
Thus, we must introduce non-linearity somewhere.

Semilinear Activation

instead of using a linear activation function, we will use a Sigmoid function in order to introduce more complexity.

Logistic Activation Consider the plot below:



It shows the sigmoid function plotted along an x axis of score. Recall score is defined as:

$$z(\hat{x}_i) = \frac{1}{1 + e^{-\hat{x}_i^\top \hat{w}}}$$

We know this equation is differentiable because:

$$\frac{d}{d\hat{w}} \to 0 \text{ as } z \to \pm\infty$$

The derivative of the function can be seen to represent a point's closeness to the hyperplane, as the farther away from the hyperplane a point goes, the bigger it's score will be, which will put it on a flat part of the sigmoid function, which means the derivative is 0.
Unfortunately, to date there is no known closed form solution for the equation.

Residual Error

The residual error of a classification is the difference between it's label and the actual output. There are many ways of representing the error, for example: Squared Error.

c Residual Error of Scores

This section will explore 2 ways of formulating error for a single artificial neuron. The exact formula for Residual Error r_i is:

$$r_i = y_1 - z(\hat{w}; \hat{x}_i)$$

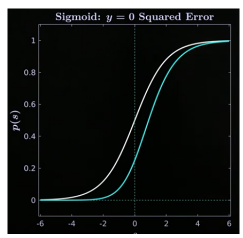
There are 2 formulations which are used for calculating the error of a neuron:

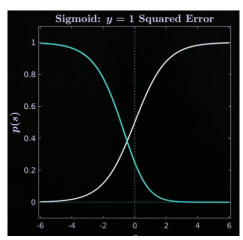
$$e_i = (r_i)^2 \tag{1}$$

$$e_i = \begin{cases} -\ln(1 - z(\hat{w}; \hat{x}_i)) & \text{if } y_i = 0 \\ -\ln(z(\hat{w}; \hat{x}_i)) & \text{if } y_i = 1 \end{cases} \tag{2}$$

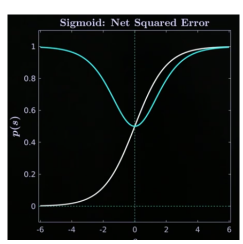
Example 1

If we were to plot the error, using equation (1), of 2 points with labels 1 and 0 with the Logistic Function, we would get something like these:



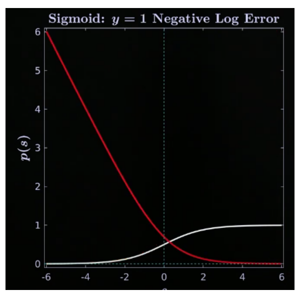


Then, if one were to add the errors together, one would get the net squared error (with a local minimum!)

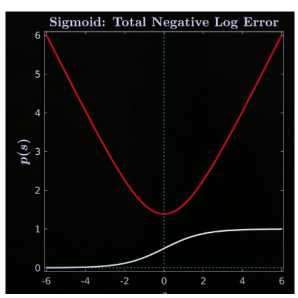


Example 2

If we then did the same thing with equation (2) we would get:



Such that the summation looks like this. Note that this equation more heavily penalizes misclassification compared to the above formula.



Sigmoid Summary

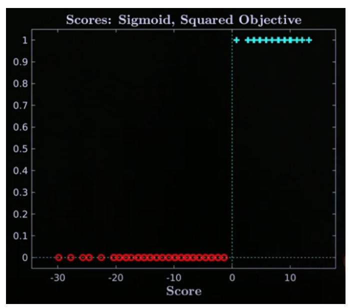
- errors are added for all m observations
- The net error is non-linear as a function of the weights $\hat{w}$

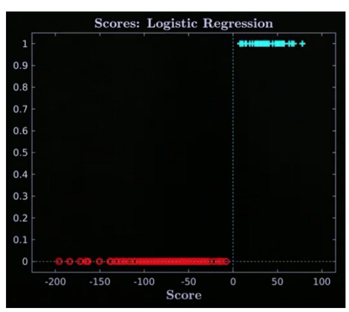
- the numerical methods for finding the error are:
  - squared error - optimised by following the steepest descent
  - logistic regression - optimized by maximum likelihood. (can be done using `glmfit` in MatLab

Note: using logistic regression on linearly separable data will cause the weights to go to infinity.

d Logistic Regression for Iris Data

If one were to take the score of data using both equation 1 and 2, and then plot them one would see that 0 is the separating score and that the scores from equation (2) are nearly a factor of 10 larger than those from (1).





Semilinear Activation Summary

- Is needed for non-linear classification (ie. Xor)

- requires a numerical solution
- there are many conceptual frameworks for use; it just depends on the data and eventual uses.

Learning Outcomes

Students should now be able to:

- determine whether data may be linearly seperable by plotting the data for 2D or plotting the scores for higher dimensions.
- Use library functions to compute scores (ie. `glmfit`)
- assess the results, using accuracy or otherwise.