Implementation of linear regression for a classification problem

Steven Luu

Abstract

We implement linear regression from first principles and apply it to classify sets of labeled data. For simplicity, we work with a data set which contains only two features and belongs to one of two classifications.

1 Introduction

The data we work with contains m examples, which we denote as $x^{(i)}$, where i refers to the i^{th} example of the dataset. In general, the datum, $x^{(i)}$, may contain n features and hence we have that $x^{(i)} \in \mathbb{R}^n$. For this problem, we aim to build a model which can classify each of the datum from their features. In general, the dataset may have k types of classifications. We denote the label/classification of $x^{(i)}$ as being $y^{(i)} \in \{1, 2, \ldots, k\}$.

For this specific dataset, the datum contains 100 examples with only two features and belongs to one of two classifications. This means that we have that $m = 100$, $n = 2$ and $k = 2$; this is therefore a binary classification problem.

For binary classification problems, it is typical to use the sigmoid function, $g(x) = 1/(1 + e^{-x})$. The sigmoid function takes values between 0 and 1 and hence the output of the sigmoid function can be interpreted as a probability. For example, suppose that the two classification values are 1 and -1 (true or false, cat or dog, etc) and that $g(x) = 0.8$. This model can then be interpreted as the belief that x has a 80 percent chance of being classified as 1 and 20 percent as being classified as -1 (or vice versa). This can be described mathematically as

$$p(y = 1|x) = g(x), \tag{1}$$
$$p(y = -1|x) = 1 - g(x), \tag{2}$$

where $p(y = 1|x)$ is the conditional probability that the data is classified as a 1 given its features.

From this, we may determine a critical curve, which we will call the 'separatrix'. The location of this curve describes the chances of the datum being classified as either 1 or -1 to be 50 percent.

2 The model

We implement the linear regression model by using the sigmoid function. More specifically, our hypothesis of the model, $h_\theta(x)$, is given by

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}, \tag{3}$$

where $\theta \in \mathbb{R}^{n+1}$ are the parameters of the model (which contains a bias term). As described before, the output of the model will therefore have the interpretation of

$$p(y = 1|x) = h_\theta(x), \tag{4}$$
$$p(y = -1|x) = 1 - h_\theta(x). \tag{5}$$

To find the optimal parameters of this model, we will calculate the averaged empirical loss and use this as our cost function, $l(\theta)$.

$$l(\theta) = -\frac{1}{m}\sum_{i=1}^{m} \log(h_\theta(y^{(i)}x^{(i)})). \tag{6}$$

This cost function measures hows accurately the model can correctly classify $x^{(i)}$. The closer the value of $h_\theta(y^{(i)}x^{(i)})$ is to one, the more accurate the model. Consequently, the value of $\log(h_\theta(y^{(i)}x^{(i)}))$ takes values closer to zero if the model is to be increasing accurate. Therefore, the aim is to find parameters values which minimize the cost function $l(\theta)$.

A typical algorithm which can be used to find these optimal parameters is known as Newton's method. For a given cost function, Newton's method gives us the update rule as

$$\theta \mapsto \theta - H^{-1}\nabla_\theta l(\theta), \tag{7}$$

where H is the Hessian of $l(\theta)$ and ∇_θ is the Jacobian operator (with respect to the parameters). We note that this is also commonly known as gradient descent.

3 Results

We first obtain an initial visualization of the data. Visualization of the data is very straightforward as the dataset only contains two feature variables.

The code used for this classification problem is contained in the file named 'LR_Code'. The training algorithm uses batch gradient descent to find the optimal parameters and continues until the difference between the two latest parameter values is no greater than a specified tolerance value, indicating convergence.

We note that this model has been trained on the complete dataset. While this is not useful for predictive purposes, the aim of this task was to illustrate how linear regression can be used for a supervised classification problem.

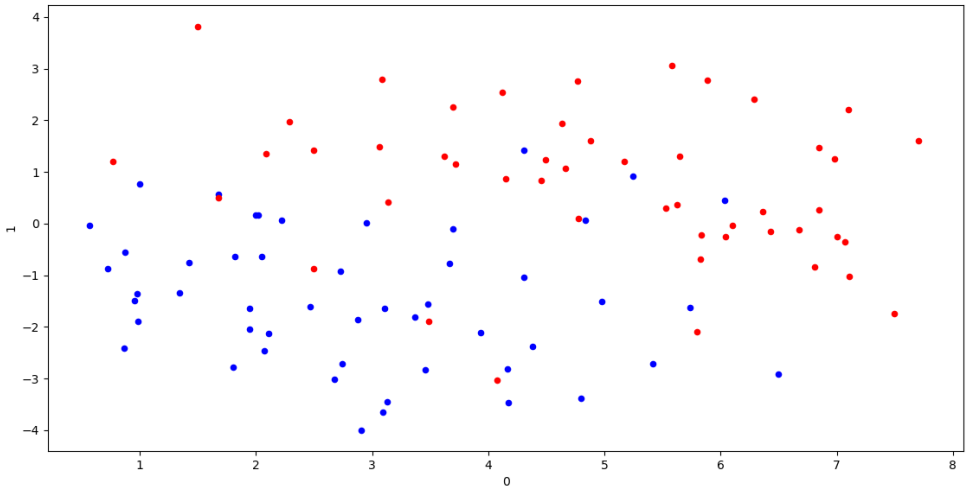

Figure 3.1: In this figure the dataset is plotted against its two features, which are denoted as 0 and 1. The red dots illustrate the data examples which are classified as '1' while those in blue are those classified as '-1'.

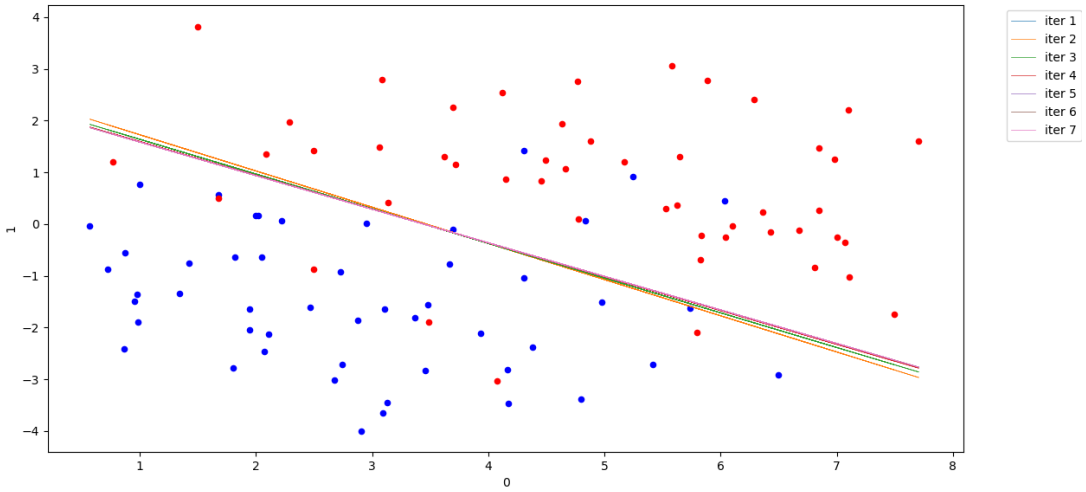

Figure 3.2: This figure illustrates the separatrix curve determined at each iteration of the gradient descent algorithm; the training algorithm converges after 7 steps. We see that the separatrix curve aims to separate the dataset into its two respective classifications. Points which are further away from the separatrix have a higher probability of being correctly classified while those closer to the separatrix are less probable of being correctly classified.