LOGISTIC REGRESSION

MAHESH VEMULA[1]

CONTENTS

1 Introduction 2
1.1 Regression 2
1.2 Classification 2
2 Logit Function 3
3 Estimation 3
3.1 Estimation Methods 4
4 Regularization 4

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

We will derive the logistic regression model. I will be closely following the book "Elements of Statistical Learning" regards the notation and derivation. Let us a take a small diversion into Statistical Decision Theory.

Suppose we have a training input $X \in \mathbb{R}^p$ which are random variables and the corresponding output Y.

1.1 *Regression*

Consider a regression problem, in that scenario we are interested in finding a regression function that minimizes a certain loss function. Let us take the following $L-2$ norm, mean square error to be our loss function. Since the inputs and outputs are random variables we are interested in an expectation prediction error (**EPE**)

$$\begin{aligned} \mathbf{EPE} &= E\left(Y - f(X)\right)^2 && (1)\\ &= \int (Y - f(X))^2 P(dx, dy) && (2) \end{aligned}$$

Minimization of the **EPE**leads to a regression function

$$f(x) = E\left(Y|X = x\right) \qquad (3)$$

So a regression function which approximate the expected conditional distribution is the best in terms of mean square error loss function.

1.2 *Classification*

Like in the regression function we consider a zero-one loss function **L** that penalizes a unit for misclassification. The expectation prediction error (**EPE**)

$$\begin{aligned} \mathbf{EPE} &= E\left[L\left(G, \hat{G}(X)\right)\right] && (4)\\ &= E_X \sum L\left(G_k, \hat{G}(X)\right) P(G_k(X)) && (5) \end{aligned}$$

Minimization of the **EPE**is then equivalent to

$$\hat{G}(X) = \text{argmin} \sum L\left(G_k, \hat{G}(X)\right) P(G_k(X)) \qquad (6)$$

For a zero -one loss function this turns out that the best classfication function is the following

$$\begin{aligned} \hat{G}(X) &= \text{argmin}\,[1 - P(g|X = x)] && (7)\\ &= \max P(g|X = x) && (8) \end{aligned}$$

i.e. the assignment that has the maximum posterior probability.

2 LOGIT FUNCTION

Thus for classification problems we are more interested in approximating and modelling the $P(G = k|X = x)$. A simple stratighforward way of doing this would be to model this probablity as a linear function i.e. for a two class problem.

$$P(G = 1|X = x) = \beta_0 + \beta^T x \tag{9}$$

As can be seen , the left side of the equation is a probability bounded between 0 and 1 whereas the right side of the equation is unbounded. This inconsistency indicates a poor model fit and we choose an alternative which retains some of the linearity and also regular, i.e the logit function

$$\log\left(\frac{P(G = 1|X = x)}{1 - P(G = 1|X = x)}\right) = \beta_0 + \beta^T x \tag{10}$$

$$P(G = k|X = x) = \frac{\exp(\beta_0 + \beta^T x)}{1 + \exp(\beta_0 + \beta^T x)} \tag{11}$$

3 ESTIMATION

Estimation of the weights proceeds via maximization of the the log likelihood. Here the outputs are 1's and 0's with probabilities $p(x; \beta)$ and $1 - p(x; \beta)$. (Note for a binomial disttibution , we have $L(p) = p^y(1 - p)^{1-y}$. The log likelihood is therefore written as follows

$$\begin{aligned} l(\beta) &= \sum_i y_i \log p(x_i; \beta) + (1 - y_i)\log(1 - p(x_i; \beta)) && (12) \\ &= \sum_i y_i \log\frac{p(x_i; \beta)}{(1 - p(x_i; \beta))} - \log(1 - p(x_i; \beta)) && (13) \\ &= \sum_i y_i \beta^T x - \log\left(1 + \exp^{\beta x}\right) && (14) \end{aligned}$$

Note we have used $\log\left(\frac{p}{1-p}\right) = \beta^T x$ and $\frac{1}{1-p} = 1 + \beta^T x$.
Maximization of 14 by taking the derivatives we have

$$\begin{aligned} \frac{\partial l(\beta)}{\partial \beta} &= \sum_i y_i x_i - \frac{\exp^{\beta x}}{1 + \exp^{\beta x} x} && (15) \\ &= \sum_i y_i x_i - p(x_i)(x_i) && (16) \\ &= \sum_i x_i \left(y_i - p(x_i)\right) && (17) \end{aligned}$$

3.1 *Estimation Methods*

1. Newton Raphson Method: The Newton Raphson method for approximating a given objective function $l(\beta)$ with parameter β is as follows

$$\beta^{new} = \beta^{old} - \left(\frac{\partial^2 l(\beta)}{\partial\beta\partial\beta^T}\right)^{-1} \frac{\partial l(\beta)}{\partial\beta^T} \tag{18}$$

   Computing the Hessian of the matrix and solving for the inverse for big data problems is not a feasible solution.

2. Stochastic Gradient Method: A gradient based update is as follow:

$$\begin{aligned} \beta^{new} &= \beta^{old} + \lambda \frac{\partial l(\beta)}{\partial\beta^T} && (19) \\ &= \beta^{old} + \lambda \sum_i x_i \left(y_i - p(x_i)\right) && (20) \end{aligned}$$

   Computing the derivative by summing for all observations in 20 will be an computationaly intensive iterative task. For stochastic graidient , the general idea is that we obtain a random approximation to the gradient by taking one sample (or possibly in bactches). For a one sample approach this turns out to be as follows:

$$\beta^{new} = \beta^{old} + \lambda x_i \left(y_i - p(x_i)\right) \tag{21}$$

4 REGULARIZATION

In trying to learn from the data sometimes the algorithm overfits the data and this gets reflected in the form of large weights. The idea behind regularization is to impose a penalty on the magnitude of the weights. The objective function is therefore transformed as follows

$$\mathbf{J}(\beta) = l(\beta) - \mu\beta\beta^T \tag{22}$$

$$\frac{\partial \mathbf{J}(\beta)}{\partial\beta} = \frac{\partial l(\beta)}{\partial\beta} - 2\mu\beta \tag{23}$$

$$\beta^{new} = \beta^{old} + \lambda x_i \left(y_i - p(x_i)\right) - 2\mu\beta \tag{24}$$

Therefore the update equations for the stochastic gradient method now become as follows