Logistic Regression with L_2 Regularization

Sandy Wiraatmadja
swiraatm@eng.ucsd.edu

Qiheng Wang
qiw018@cs.ucsd.edu

Abstract

In this paper, we explore logistic regression with L_2 regularization as a binary classification learning model. Given a set of data with different feature values, we want to be able to closely predict what the binary label is for each example. This can be done by training the model on a training set to find the parameters that maximize its log conditional likelihood (LCL). Two optimization methods are analyzed in this paper: Stochastic Gradient Descent (SGD) and Limited-memory BFGS (L-BFGS). The two methods are applied to the Gender Recognition [DCT] dataset from MLcomp. Logistic regression with L-BFGS produces higher test accuracy compared to SGD.

1 Introduction

Machine learning, which is a branch of artificial intelligence, has been growing significantly due to the availability of massive data that can be used in developing and training models. One common problem in machine learning is to train a model that can be used for binary statistical classification. For example, given a data of email messages, some classified as spam and some as non-spam, a model can be trained to learn some distinguishing features between spam and non-spam emails. After the learning process is done and the parameters are chosen, the model can be used on a new data of email messages to label each email as either spam or non-spam. Several classifier learning algorithms have been developed throughout the years, such as k-nearest neighbors, linear regression, or support vector machines [1]. One algorithm that we will focus on in this paper is logistic regression.

Logistic regression model is widely used in probabilistic classification by fitting the training data to the model and find the parameters that maximizes the log joint conditional likelihood (LCL) of the training set. In this paper, we analyze how to train a logistic regression model for our binary (Bernoulli) label classification problem with two different gradient-based optimization methods for maximizing the LCL. These two algorithms are Stochastic Gradient Descent (SGD) which uses a modified version of a gradient descent method, and Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) which uses the quasi-Newton methods [2]. We use our own implementation of the SGD algorithm in Matlab, while on the other hand, we used Mark Schmidt's Matlab `minFunc` function which implemented L-BFGS. We compare the two models to see how well they perform on the Gender Recognition [DCT] test set from MLcomp.

2 Design and Analysis of Algorithms

Logistic regression uses the principle of maximum log conditional likelihood, where we choose a parameter estimate $\hat{\theta}$ that maximizes the log joint conditional likelihood [3]. This is the sum of the log conditional likelihood for each training example:

$$
\begin{aligned}
LCL &= \sum_{i=1}^{n} \log L(\theta; y_i|x_i) = \sum_{i=1}^{n} \log f(y_i|x_i;\theta) \\
&= \sum_{i:y_i=1} \log p_i + \sum_{i:y_i=0} \log(1-p_i) \\
&= \sum_{i=1}^{n} \log(p_i y_i + (1-p_i)y_i). \qquad (1)
\end{aligned}
$$

Here we assume the conditional model

$$
p_i = p(Y=1|x;\alpha,\beta) = \sigma(\alpha + \sum_{j=1}^{d} \beta_j x_j) = \frac{1}{1+\exp -[\alpha + \sum_{j=1}^{d} \beta_j x_j]} \qquad (2)
$$

where α is the intercept. For simplification, we assume that $\alpha = \beta_0$ and we add $x_0 = 1$ for all examples, such that (2) becomes

$$
p_i = \frac{1}{1+\exp -[\sum_{j=0}^{d} \beta_j x_j]}. \qquad (3)
$$

The objective function then becomes

$$
\hat{\beta} = \underset{\beta}{\operatorname{argmax}}\, LCL. \qquad (4)
$$

Since logistic regression tries to fit the training data into the model to maximize LCL, the problem of overfitting tend to arise. The standard method to solve this problem is regularization, which imposes a penalty on the magnitude of the parameter values [3]. However, there is a tradeoff between minimizing the regularization penalty and maximizing the regularized log joint conditional likelihood (RLCL). The objective function for the optimization problem is

$$
\hat{\beta} = \underset{\beta}{\operatorname{argmax}}\, RLCL = \underset{\beta}{\operatorname{argmax}}(LCL - \mu||\beta||_2^2) \qquad (5)
$$

where $||\beta||_2^2 = \sum_{j=1}^{d} \beta_j^2$ is the squared L_2 norm of the parameter vector β of length d. The constant μ is the strength of the regularization which quantifies the trade-off between maximizing likelihood and minimizing parameter values, making them close to zero. This is called the quadratic or Tikhonov regularization. Note that since each training data provides information about the intercept of the model β_0, there is enough information to avoid overfitting of this parameter. Therefore, β_0 does not need to be regularized.

The partial derivative of LCL with respect to parameter β_j is

$$
\frac{\partial}{\partial \beta_j} LCL = \sum_i (y_i - p_i) x_{ij} \qquad (6)
$$

whereas the partial derivative of RLCL with respect to parameter β_j is

$$
\frac{\partial}{\partial \beta_j} RLCL = \sum_i (y_i - p_i) x_{ij} - 2\mu\beta_j. \qquad (7)
$$

The following subsections discuss the different algorithm of the two optimization methods.

2.1 Stochastic Gradient Descent

Since our objective function is a maximization problem, this method should be more appropriately called the stochastic gradient ascent. The parameter values β is changed one step at a time following the gradient until it finally converges to the local maxima point.

The regular gradient descent update rule of the parameter β_j is

$$\beta_j := \beta_j + \lambda \frac{\partial}{\partial \beta_j} RLCL \tag{8}$$

where λ is the learning rate, denoting how big of a step each update should take.

However, calculating the partial derivatives per iteration can be time consuming as each requires $O(nd)$ time where n is the training example size and d is the number of features. To minimize computation, we use stochastic gradient descent method, where we get a random approximation to the partial derivatives by just looking at one randomly chosen example at a time to update β. Thus, we can drop n and do it in much less time, around $O(d)$ time, independent of the size of the training data. This is extremely useful for very large training set.

By using SGD, the parameter update rule then becomes

$$\beta_j := \beta_j + \lambda[(y_i - p_i)x_j - 2\mu\beta_j] \tag{9}$$

for any randomly chosen i^{th} example.

And the update rule for the whole parameter vector $\bar{\beta}$ is

$$\bar{\beta} := \bar{\beta} + \lambda(\bar{y} - \bar{p})^T X \tag{10}$$

which should be done at least once after all epochs of stochastic gradient ascent. An epoch is a complete update for every example in the dataset.

Convergence is reached when the change in the objective value, RLCL, is within a certain threshold which we hold constant at 10^{-3}. Typically, this check is done in the middle of an epoch, which means that the iteration can stop if it reaches convergence before even going through the entire examples. This can save a lot of time when dealing with massive data. However, we decided to do the convergence check after every epoch. The learning rate λ is made smaller by 10% after every epoch. This is done so that convergence is reached faster, and also bigger λ means bigger step and this might cause unstability where the objective function cannot reach its true local maxima.

2.2 L-BFGS

L-BFGS is an optimization algorithm of the quasi-Newton method using a limited amount of computer memory[2]. This method is often used for parameter estimation. As mentioned above, for the purpose of this paper, we use Mark Schmidt's `minFunc` function, written in Matlab, that implements the L-BFGS algorithm. This function is a minimizing function. Therefore, we need to modify the objective function (5) to be

$$\begin{aligned} \hat{\beta} &= \underset{\beta}{\operatorname{argmin}}(-RLCL) \\ &= \underset{\beta}{\operatorname{argmin}}(-LCL + \mu||\beta||_2^2) \end{aligned} \tag{11}$$

3 Design of Experiments

4 Results of Experiments

5 Findings and Lessons Learned

References

[1] Wikipedia article *Binary classification*. Available at `http://en.wikipedia.org/wiki/Binary_classification`.

[2] Wikipedia article *Limited-memory BFGS*. Available at `http://en.wikipedia.org/wiki/Limited-memory_BFGS`.

[3] Elkan, C. (2014). *Maximum Likelihood, Logistic Regression, and Stochastic Gradient Training*. Available at `http://cseweb.ucsd.edu/~elkan/250B/logreg.pdf`.