Cyclic Coordinate Descent for Logistic Regression with Lasso regularization

Patryk Prusak

supervisor

XYZ

Warsaw University of Technology

March 22, 2025

Advanced Machine Learning Course

Contents

1 Methodology

1.1 Selection and generation of datasets

1.2 Details about algorithm implementation and applied optimizations

Logistic Regression is a machine learning method capable of binary classification. It predicts the probability of an outcome by computing the linear combination of input features and weights (Formula 1), then passing it through the sigmoid function presented in Formula 2.

$$z = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n = \mathbf{w}^T \mathbf{x} + b \tag{1}$$

Where x denotes input feature vector, while $x_1, ..., x_n$ are the elements of that vector, w denotes model weights vector and b is the bias term.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2}$$

The output of the sigmoid function in range $[0, 1]$ denotes the probability that given feature vector x belongs to the positive class.What follows the prediction rule is based on the output of the sigmoid function, if it's larger than a set threshold, such as 0.5, we assign the sample to class 1, otherwise assign to class 0.

To fit the model to the training data one needs to minimize the loss function, in this case Binary Cross-Entropy defined in Formula 3.

$$\mathcal{L} = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log \hat{y}^{(i)} + \left(1 - y^{(i)}\right) \log \left(1 - \hat{y}^{(i)}\right) \right] \tag{3}$$

Where m denotes the number of training examples, $y^{(i)}$ is the class label and $\hat{y}^{(i)}$ is the predicted probability. The weights of the model need to be optimized to find the proper fit, this can be achieved by standard gradient descent algorithm. The weights are updated according to the following formulas (Formula 4, Formula 5):

$$w_j := w_j - \alpha \frac{\partial \mathcal{L}}{\partial w_j} \tag{4}$$

$$b := b - \alpha \frac{\partial \mathcal{L}}{\partial b} \tag{5}$$

Where α is the learning rate, the higher the value the more aggressive weight updates and $\frac{\partial \mathcal{L}}{\partial w_j}$ is a gradient with respect to weight w_j.

One of the methods to prevent overfitting of the model to the training data is Lasso Regulaization. Overfitting describes the situation when the trained model can predict samples from the training set very well but struggles on the test set. The loss function with Lasso regularization is defined in Formula 6.

$$\mathcal{L}_{\text{lasso}} = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log \hat{y}^{(i)} + \left(1 - y^{(i)}\right) \log \left(1 - \hat{y}^{(i)}\right) \right] + \lambda \sum_{j=1}^{n} |w_j| \tag{6}$$

Where m is the number of training samples, $y^{(i)}$ is the class label, $\hat{y}^{(i)}$ is the predicted probability, λ denotes regularization strength.

In essence during the training process, the model will also minimize the absolute sum of the coefficients in addition to the loss function. This will result in some of the weights being set to zero, effectively reducing the number of features the model is trained on. This can be useful in situations where the number of features is very large and some of them are irrelevant to the prediction task.

Now, to use the Cyclic Coordinate Descent instead of the standard Gradient Descent one needs to minimize the $\mathcal{L}_{\text{lasso}}$ using a different algorithm for updating model weights.

However the authors of the 2010 publication entitled *Regularization Paths for Generalized Linear Models via Coordinate Descent* [1] present a more sophisticated approach with certain optimizations. First, the Logistic Regression log-likelihood:

$$\max_{\beta} \sum_{i=1}^{N} [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)] - \lambda ||\beta||_1$$

is approximated with a quadratic approximation:

$$\ell_Q(\beta_0, \beta) = -\frac{1}{2N} \sum_{i=1}^{N} w_i (z_i - \beta_0 - x_i^T \beta)^2 + C$$

That converts the problem into a penalized weighted least squares.

The authors also use a regularization path that starts from largest λ where $\beta = 0$ and decreases λ gradually, using previous solutions as warm starts.

Instead of computing gradients from scratch with each iteration, the authors propose to use covariance updates:

$$\sum_{i=1}^{N} x_{ij} r_i = \langle x_j, y \rangle - \sum_{k:\beta_k \neq 0} \langle x_j, x_k \rangle \beta_k$$

For each feature j, the optimization problem simplifies to:

$$\min_{\beta_j} \left[\frac{1}{2} \sum_{i=1}^{N} w_i \left(z_i - \beta_0 - \sum_{k \neq j} x_{ik} \beta_k - x_{ij} \beta_j \right)^2 + \lambda |\beta_j| \right]$$

To update the weights using CCD one needs to (for weight j):

1. Compute partial residuals (excluding β_j)

$$r_i = z_i - (\beta_0 + \sum_{k \neq j} x_{ik} \beta_k)$$

2. Compute the gradient component ρ_j

$$\rho_j = \sum_{i=1}^{N} w_i x_{ij} r_i$$

3. Apply soft-thresholding for L1 regularization

$$\beta_j = \frac{S(\rho_j, \lambda)}{\sum_{i=1}^{N} w_i x_{ij}^2}$$

$$S(z, \lambda) = \text{sign}(z) \cdot \max(|z| - \lambda, 0)$$

4. Update β_0 that is not regularized

$$\beta_0 = \frac{\sum_{i=1}^{N} w_i (z_i - x_i^T \beta)}{\sum_{i=1}^{N} w_i}$$

From a high level overview the presented algorithm consists of:

1. Outer Loop: Decrease λ along a regularization path.
2. Middle Loop: Update the quadratic approximation using the current (β_0, β).
3. Inner Loop: Perform coordinate descent on the penalized weighted least squares problem.

2 Discussion about correctness of the LogRegCCD algorithm

2.1 Performance of the algorithm at lambda=0

2.2 Likelihood function values and coefficient values depending on iteration

2.3 Comparison with ready implementation of logistic regression with L1 penalty

3 Impact of dataset parameters: n.p,d,g on the performance of LogRegCCD algorithm

4 Benchmark of LogRegCCD with LogisticRegression algorithm

4.1 Performance of algorithms regarding different metrics

4.2 Values of coefficients obtained in these two methods

List of Figures

List of Tables

References

[1] Jerome H. Friedman, Trevor Hastie, and Robert Tibshirani. "Regularization Paths for Generalized Linear Models via Coordinate Descent". In: *Journal of Statistical Software* 33.1 (2010), pp. 1–22. DOI: 10.18637/jss.v033.i01. URL: https://doi.org/10.18637/jss.v033.i01.