Solution to logistic regression

- Given data $(x_1, y_1), \cdots, (x_n, y_n)$, y_i are binary response variables. The maximum likehihood estimator for logistic regression is the solution to the problem:

$$\min_{\beta} f(\beta) = \sum_{i=1}^{n} log(1 + e^{-y\beta^T x_i}).$$

- Gradient of $f(\beta)$:

$$\nabla f(\beta) = \sum_{i=1}^{n} \frac{-y_i x_i^T e^{-y\beta^T x_i}}{1 + e^{-y\beta^T x_i}} = \sum_{i=1}^{n} \frac{-y_i x_i^T}{1 + e^{y\beta^T x_i}}$$

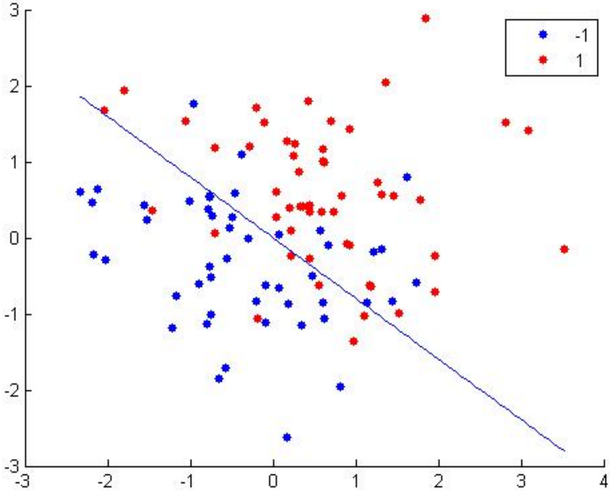
Gradient descent method for logistic regression

- Initialize β_k is a starting point at 0, k=0, α is small steplength, calculate $\nabla f(\beta_k)$,$d_k = -\nabla f(\beta_k)$
- $\beta_{k+1} = \beta_{k+1} + \alpha d_k$., evaluate $\nabla f(\beta_{k+1})$,d_{k+1}.
- Repeat until $\|\nabla f(\beta_k)\|_2^2$ is small enough or until a maximum number of iterations.

Matlab example

- Function simulate_logistic generates a data set for logistic regression. Inputs n: number of rows of predictor matrix X (number of observations), p: number of predictors, β is a predetermined parameter.
- Function gradient_logistic evaluates the function value of likelihood at a point β with data X, y, and also provide the gradient at that point.
- Function gradient_descent_logistic performs gradient descent method on this particular data. The output is the minimizer of the likelihood function β.

Matlab example 1



Matlab example 2

- Use logistic regression to predict whether a patient has heart disease or not. (This example is borrowed from CS490, Prof. Howbert, UW)
- Heart disease patient data set contains data related to the patients' age, gender, blood pressure, maximum heart rate...
- Label: 1 for no heart disease, 2 for heart disease.
- Matlab function for logistic regression: mrnfit(X,y), X is the matrix of predictors, y contains labels.

Regularization methods for logistic regression

- Similar to linear regression, when $p \gg n$, the MLE solution in logistic regression is very unstable.
- The solution has very high variance so one must be very careful when using this estimator to make prediction.
- Regularization methods can be a remedy to this situation.

Logistic regression with ridge penalty

- Logistic regression with ridge penalty

$$\min_{\beta} f(\beta) = \sum_{i=1}^{n} log(1 + e^{-y\beta^T x_i}) + \frac{\lambda}{2}\|\beta\|_2^2.$$

- Gradient of $f(\beta)$:

$$\nabla f(\beta) = \sum_{i=1}^{n} \frac{-y_i x_i^T e^{-y\beta^T x_i}}{1 + e^{-y\beta^T x_i}} + \lambda\beta = \sum_{i=1}^{n} \frac{-y_i x_i^T}{1 + e^{y\beta^T x_i}} + \lambda\beta.$$

- With little modification at the step of evaluating the gradient, we can use gradient descent method to find the solution to ridge logistic regression.

Logistic regression with Lasso penalty

Gradient descent-ish method for Lasso Logistic regression

$$\min_{\beta} \sum_{i=1}^{n} log(1 + e^{-y\beta^T x_i}) + \lambda\|\beta\|_1, \lambda > 0$$

Also denote $f(\beta) = \sum_{i=1}^{n} log(1 + e^{-y\beta^T x_i})$, $g(\beta) = \lambda\|\beta\|_1$

- Initialization: A point β^0, iteration count k, α steplength:
- Repeat: Calculate $\nabla f(\beta^k)$. Solve the sub-problem and set the solution to β^{k+1}

$$\beta^{k+1} = \min f(\beta^k) + \langle \nabla f(\beta^k), \beta - \beta^k \rangle + g(\beta) + \frac{1}{2\alpha}\|\beta - \beta^k\|_2^2.$$

Gradient descent-ish method to find Lasso Logistic regression

- This process is repeat until no improvement is made.
- In the k iteration the sub-problem is equivalent to:

$$\min_{\beta}\langle \nabla f(\beta^k), \beta - \beta^k\rangle + \lambda\|\beta\|_1 + \frac{1}{2\alpha}\|\beta - \beta^k\|_2^2$$

$$\min_{\beta}\langle \nabla f(\beta^k), \beta\rangle + \lambda\|\beta\|_1 + \frac{1}{2\alpha}(\langle \beta, \beta\rangle - 2\langle \beta, \beta^k\rangle)$$

$$\min_{\beta} \lambda\|\beta\|_1 + \frac{1}{2\alpha}[\langle \beta, \beta\rangle - 2\langle \beta, \beta^k - \alpha\nabla f(\beta^k)\rangle]$$

$$\min_{\beta} \lambda\|\beta\|_1 + \frac{1}{2\alpha}\|\beta - (\beta^k - \alpha\nabla f(\beta^k))\|_2^2.$$

Gradient descent-ish method to find Lasso Logistic regression

- Let $w = \beta^k - \alpha \nabla f(\beta^k)$, then the solution β^{k+1} is :

$$\beta_i^{k+1} = \begin{cases} w_i - \lambda\alpha & \text{if } w_i > \lambda\alpha \\ w_i + \lambda\alpha & \text{if } w_i < -\lambda\alpha \\ 0 & \text{if } \textit{otherwise} \end{cases}$$

Gradient descent-ish method to find Lasso Logistic regression

- Initialization: A point β^0, iteration count k, α steplength:
- Repeat: Calculate $\nabla f(\beta^k)$, $w = \beta^k - \alpha \nabla f(\beta^k)$ then the solution β^{k+1} is :

$$\beta_i^{k+1} = \begin{cases} w_i - \lambda\alpha & \text{if } w_i > \lambda\alpha \\ w_i + \lambda\alpha & \text{if } w_i < -\lambda\alpha \\ 0 & \text{if } \textit{otherwise} \end{cases}$$

- Until no improvement can be made.

Examples using Lasso Logistic regression

- Simulate a data set for logistic regression with $\beta = [1, 2, 3]$
- Calculate the Lasso logistic solution with different values of tuning parameters λ.
- Make prediction on the test data set.
- Plot solution

Hand written digit data set

- Handwritten digit data contains images of digits 1 and 2.
- The data contains 1732 images. Use 25% of the data as training, the rest 75% as testing.
- Perform Lasso Logistic regression with Cross validation to find the best classifier.