Machine Learning, Lecture 7: Logistic regression: Model Fitting

S. Nõmm

[1]Department of Computer Science, Tallinn University of Technology

19.03.2015

MLE

- Let us remind that logistic regression corresponds to the following binary classification model

$$p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) = \text{Ber}(y \mid \text{sigm}(\boldsymbol{\theta}^T \boldsymbol{x}))$$

- Negative log-likelihood for logistic regression

$$\begin{aligned}
\mathcal{NLL}(\boldsymbol{\theta}) &= -\sum_{i=1}^{N} \log\left[\mu_i^{\mathbf{1}(y_i=1)} \times (1-\mu_i)^{\mathbf{1}(y_i=0)}\right] \\
&= -\sum_{i=1}^{N} \left[y_i \log \mu_i + (1-y_i) \log(1-\mu_i)\right]
\end{aligned}$$

- Suppose $\tilde{y}_i \in \{-1, 1\}$ (instead of $y_i \in \{0, 1\}$), then

$$p(y=1) = \frac{1}{1+e^{-\boldsymbol{\theta}^T \boldsymbol{x}}}; \quad p(y=-1) = \frac{1}{1+e^{\boldsymbol{\theta}^T \boldsymbol{x}}}$$

  leads

$$\mathcal{NLL}(\boldsymbol{\theta}) = \sum_{i=1}^{N} + \log(1 + e^{-\tilde{y}\boldsymbol{\theta}^T x_i})$$

MLE

$$\mathcal{NLL}(\boldsymbol{\theta}) = \sum_{i=1}^{N} + \log(1 + e^{-\tilde{y}\boldsymbol{\theta}^T x_i})$$

Gradient and Hessian are given by

$$\begin{aligned} g &= \frac{d}{d\boldsymbol{\theta}} f(\boldsymbol{\theta}) = \sum_i (\mu_i - y_i) x_i = \boldsymbol{X}^T(\boldsymbol{\mu} - y) \\ \boldsymbol{H} &= \frac{d}{d\boldsymbol{\theta}} g(\boldsymbol{\theta})^T = \sum_i \mu_i (1 - \mu_i) x_i x_i^T = \boldsymbol{X}^T \boldsymbol{S} \boldsymbol{X} \end{aligned}$$

where $S = \text{diag}(\mu_i)(1 - \mu_i)$.
$\boldsymbol{H}$ is positive define $\Rightarrow \mathcal{NLL}$ is convex and therefore has a unique minimum.

Gradient descent / Steepest descend

- Simplest algorithm for unconstrained optimization

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta_k g_k$$

where η_k is referred as the *step size* or *learning rate*. Main question is how to set the value of η_k such, that the method will converge to a local optimum irrespective from the initial point. Such property is called *Global convergence*

- According to Taylor's theorem:

$$f(\boldsymbol{\theta} + \eta \mathrm{d}) \approx f(\boldsymbol{\theta} + \eta g^T \mathrm{d})$$

where d is the descend direction. If η is too small condition

- If η is too small execution may become to slow and/or minimum may not be necessarily reached.
- *Line minimization* or *Line search*, Let us choose η such that it would minimize

$$\phi(\eta) = f(\boldsymbol{\theta}_k + \eta \mathrm{d}_k)$$

Gradient descent / Steepest descend

- *Zig-zaging effect*: Exact line search satisfies

$$\eta_k = \text{arg min}_{\eta > 0} \phi(\eta)$$

Necessary condition for the optimum is $\phi'(\eta) = 0$.
$\phi'(\eta) = \mathrm{d}^T g$ where $g = f'(\boldsymbol{\theta} + \eta \mathrm{d})$. Therefore one either have $g = 0$ or $g \perp \mathrm{d}$.

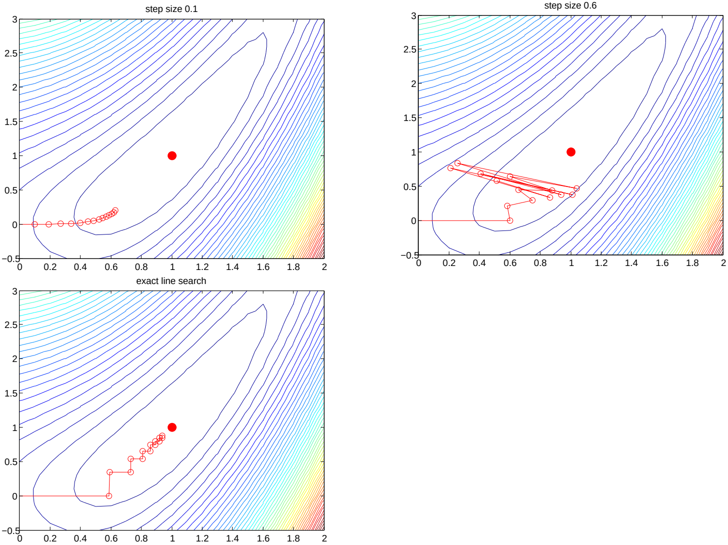
- To reduce zig-zaging add a *momentum* term, $(\theta_k - \theta_{k-1})$:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta_k \mathrm{g}_k + \mu_k(\theta_k - \theta_{k-1})$$

where $0 \leq \mu_k \leq 1$. This method is frequently referred as *heavy ball method*

Example Gradient descent

Let us consider convex function $f(\boldsymbol{\theta}) = 0.5(\theta_1^2 - \theta_2)^2 + 0.5(\theta_1 - 1)^2$
Stat from the point $(0, 0)$



Newton's method

Algorithm:

1. Initialize $\boldsymbol{\theta}_0$;
2. k=0;
3. Until converge do
4. k=k+1;
5. Evaluate $g_k = \nabla f(\boldsymbol{\theta}_k)$;
6. Evaluate $\boldsymbol{H}_k = \nabla^2 f(\boldsymbol{\theta}_k)$;
7. Solve $\boldsymbol{H}_k \mathrm{d}_k = -g_k$ for d_k;
8. Use line search to find step size η_k along d_k
9. $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \eta_k \mathrm{d}_k$
10. end until

Newton's method based techniques

- Iteratively reweighted least squares (IRLS). Applies Newton's algorithm to find MLE for binary logistic regression.
- Quasi- Newton (variable metric) methods. Replaces $\boldsymbol{H}$ by its approximation which is updated on each iteration.

ℓ_2 regularization

- Let us suppose that the data is linearly separable.
- MLE solution is obtained when $\|\boldsymbol{\theta}\| \to \infty$
- Logistic sigmoid function approach Heaviside step function and each point will be classified as 0 or 1 with probability 1. Such solution will not generalize well.
- ℓ_2 regularization: Objective, gradient and Hessian are given by:-

$$
\begin{aligned}
f'(\boldsymbol{\theta}) &= \mathcal{NLL}(\boldsymbol{\theta}) + \lambda \boldsymbol{\theta}^T \boldsymbol{\theta} \\
\mathrm{g}'(\boldsymbol{\theta}) &= \mathrm{g}(\boldsymbol{\theta}) + \lambda \boldsymbol{\theta} \\
\boldsymbol{H}'(\theta) &= \boldsymbol{H}(\boldsymbol{\theta}) + \lambda \boldsymbol{I}
\end{aligned}
$$

Online learning

- Estimates are updated as new observation point(s) arrives (becomes available). On each step the learner must respond with a parameter estimate.
- Regret minimization : The objective used in online learning is the *regret*, which is the averaged loss incurred.
- Stochastic optimization and risk minimization: The objective is to minimize expected loss

Regret minimization

- The objective used in online learning is the *regret*, which is the averaged loss incurred.

$$\text{regret}_k = \frac{1}{k}\sum_t = 1^k f(\boldsymbol{\theta}_t, \boldsymbol{z}_t) - \min_{\boldsymbol{\theta}^*} \in \Theta \frac{1}{k}\sum_{t=1}^{k} f(\boldsymbol{\theta}_*, \boldsymbol{z}_t)$$

- Online gradient descend

$$\boldsymbol{\theta}_{k+1} = \text{proj}_{\Theta}(\boldsymbol{\theta}_k - \eta_k \text{g}_k)$$

where $\text{proj}_{\nu}(v) = \arg\ \min_{\boldsymbol{\theta}\in\Theta} \|\boldsymbol{\theta} - v\|_2$

Stochastic optimization and risk minimization:

- The objective is to minimize expected loss

$$f(\boldsymbol{\theta}) = \mathbb{E}[f(\boldsymbol{\theta}, z)]$$

  where the expectation is taken over future data.

- Stochastic gradient descent (SGD). Running average:

$$\bar{\boldsymbol{\theta}}_k = \frac{1}{k}\sum_{t=1}^{k}\boldsymbol{\theta}_t$$

  which may be implemented recursively as follows:

$$\bar{\boldsymbol{\theta}}_k = \bar{\boldsymbol{\theta}}_{k-1} - \frac{1}{k}(\bar{\boldsymbol{\theta}}_{k-1} - \bar{\boldsymbol{\theta}}_k)$$

- Step size
- Pre -parameter step size

The LMS algorithm

- Compute MLE for linear regression is an online manner
- The online gradient at iteration k is given by

$$\mathrm{g}_k = x_i(\boldsymbol{\theta}_k^T x_i - y_i)$$

  where $i = i(k)$ is the training example used at iteration k
- $\boldsymbol{\theta}$ update

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta_k(\hat{y}_k - y_k)x_k$$

The perceptron algorithm

The goal is to fit a binary logistic regression model in an online manner

1. Input: Linearly separable data set $x_i \in \mathbb{R}^D, \quad y_i \in \{-1, 1\}$;
2. Initialize $\boldsymbol{\theta}_0$;
3. $k = 0$;
4. repeat
5. $\quad k = k + 1$;
6. $\quad i = k|N \ (k \mod N)$;
7. $\quad$ if $\hat{y}_y \neq y_i$ then
8. $\quad\quad \boldsymbol{\theta}_k + 1 = \boldsymbol{\theta}_k + y_i x_i$;
9. $\quad$ else
10. $\quad\quad$ do nothing
11. $\quad$ end
12. until converged

The perceptron algorithm

- Will converge provided the data is linearly separable.
- First machine learning algorithm ever derived.