Machine Learning: Problem Set 1

Justin D. Thomas

April 5, 2013

1. Newton's method for computing least squares

(a) Find the Hessian of the cost function $J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(\theta^\top x^{(i)} - y^{(i)})^2$.

Solution to 1-(a):

Note that $x^{(i)} \in \mathbb{R}^N$, where N is the number of features. For vocabulary practice m is the number of training samples. We have $\theta^\top x^{(i)} \in \mathbb{R}$ and $y^{(i)} \in \mathbb{R}$. The function J goes from $\mathbb{R}^N$ to $\mathbb{R}$. So the Hessian of J is an $N \times N$ symmetric matrix of functions $\mathbb{R}^N \to \mathbb{R}$. In fact, we will see that the Hessian of J is an $N \times N$ matrix of constant functions.

By the chain rule, we compute

$$\partial J/\partial\theta_j = \sum_i x_j^{(i)}(\theta^\top x^{(i)} - y^{(i)}). \tag{1}$$

Thus the Hessian of J is

$$H(J)_{jk} = \sum_{i=1}^{m} x_j^{(i)} x_k^{(i)}, \tag{2}$$

which does not depend on θ, so is a matrix of constant functions, as claimed.

(b) Show that the first iteration of Newton's method gives us

$$\theta^\star = (X^\top X)^{-1} X^\top \vec{y},$$

the solution to the least squares problem.

Solution to 1-(b):

By definition, X is the matrix whose i^{th} row vector is $x^{(i)}$. Thus $X_{ij} = x_j^{(i)}$ and

$$(X^\top X)_{jk} = \sum_i X_{ji}^\top X_{ik} = \sum_i x_j^{(i)} x_k^{(i)} = H(J)_{jk}, \tag{3}$$

where the last equality is (2). We apply Newton's method to find a zero of ∇J with initial guess $\theta_0 = 0$. Recall that since $J\colon \mathbb{R}^N \to \mathbb{R}$ we have $\nabla J\colon \mathbb{R}^N \to \mathbb{R}^N$. We write ∇J as an N-dimensional column vector of $\mathbb{R}$-valued functions of N variables. In particular, the j^{th} row of ∇J is $\nabla_j J := \partial J/\partial\theta_j$. By definition of Newton's method, we have

$$\theta_1 = \theta_0 - H(J)^{-1}(\nabla J)(\theta_0). \tag{4}$$

In this description θ_i is a column vector in $\mathbb{R}^N$. Since ∇J has dimension $N \times 1$ and $H(J)^{-1}$ has dimension $N \times N$, we see that the right hand side in the equation above is well-defined. Recall that ∇J is a function of θ. When $\theta = 0$ we see by equation (1) that

$$(\nabla J)(0) = -\sum_i x_j^{(i)} y^{(i)} = -\sum_i X_{ij} y^{(i)} = X^\top \vec{y}, \tag{5}$$

where, by definition, $\vec{y}$ is the $N \times 1$ vector whose i^{th} row is $y^{(i)}$, the i^{th} observed value in the training set. Putting $\theta_0 = 0$ into (4) and using equations (3) and (5), we have

$$\theta_1 = -(X^\top X)^{-1}(-X^\top \vec{y}) = (X^\top X)^{-1} X^\top \vec{y}, \tag{6}$$

as desired.

2. Locally-weighted logistic regression

Recall that logistic regression given by choosing $\theta \in \mathbb{R}^N$ to give the maximum likelihood of the sample set with the following prediction function

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^\top x}}. \tag{7}$$

In this model, $h_\theta(x)$ is predicting y, which takes values in $\{0, 1\}$. Given θ, then for each sample i the value

$$L_i(\theta) = h_\theta(x^{(i)})^{y^{(i)}} (1 - h_\theta(x^{(i)}))^{1-y^{(i)}}$$

is close to 1 if $h_\theta(x^{(i)})$ is close to $y^{(i)}$. Thus the product $L(\theta) = \prod_i L_i(\theta)$ is close to 1 if h_θ is a good predictor of $y^{(i)}$ given $x^{(i)}$ for all i. We think of this product as the likelihood of the sample $\{(x^{(i)}, y^{(i)}) \mid i = 1, \ldots, n\}$ when the parameter θ is given and h_θ is used to predict $y^{(i)}$ as a function of $x^{(i)}$. We want to maximize $L(\theta)$, but it is easier and equivalent to maximize $\ell(\theta) := \log L(\theta)$.

$$\ell(\theta) = \sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) = \sum_{i=1}^{m} \ell_i(\theta) \tag{8}$$

The log likelihood for logistic regression.

Now we localize this around $x \in \mathbb{R}^N$ using the weight function

$$w^{(i)}(x) = \exp(-\left|x - x^{(i)}\right|^2 / (2\tau^2)), \tag{9}$$

where τ is a constant parameter. This gives us

$$\ell(\theta, x) = \sum_{i=1}^{m} w^{(i)}(x)\ell_i(\theta) \tag{10}$$

Locally-weighted log likelihood linear regression.

For reasons I don't understand, Newton's method does not work well for any (or some?) value(s) of x when finding maxima of the function $\theta \mapsto \ell(\theta, x)$. We can fix this by adding a linear term to $\partial_\theta \ell(\theta, x)$, or a θ-quadratic term to $\ell(\theta, x)$. We take a quadratic term of the form $(-\lambda/2)\left|\theta\right|^2$, or $\lambda\theta^\top\theta$, where λ is very small, say $\lambda = 0.0001$. The addition of a linear term to a function will adjust the critical points, but when the linear term is small, they are not too far off. Our adjusted likelihood function for locally weighted linear regression is

$$\ell'(\theta, x) = -\frac{\lambda}{2}\theta^\top\theta + \ell(\theta, x) \tag{11}$$

Adjusted log likelihood for locally-weighted linear regression.

Now we compute $\partial_\theta \ell$ (really the gradient). Clearly $\partial_\theta(-\lambda/2)\left|\theta\right|^2 = -\lambda\theta$. Also $\partial_\theta \ell(\theta, x) = \sum_i w^{(i)}(x)\partial_\theta \ell_i(\theta)$. So we need to compute $\partial_\theta \ell_i(\theta)$. We use

$$\begin{aligned} \partial_{\theta_j} h_\theta(x) &= \partial_{\theta_j} \frac{1}{1 + e^{-\theta^\top x}} \\ &= \frac{x_j e^{-\theta^\top x}}{(1 + e^{-\theta^\top x})^2} \\ &= x_j h_\theta(x)(1 - h_\theta(x)) \end{aligned} \tag{12}$$

Thus

$$\begin{aligned} \partial_{\theta_j} \log h_\theta(x^{(i)}) &= x_j^{(i)}(1 - h_\theta(x^{(i)})), \\ \partial_{\theta_j} \log(1 - h_\theta(x^{(i)})) &= -x_j^{(i)} h_\theta(x^{(i)}). \end{aligned}$$

Now if we set $z \in \mathbb{R}^m$ to be the column vector with i^{th} entry, z_i, given by

$w^{(i)}(y^{(i)} - h_\theta(x^{(i)}))$ we have

$$\begin{aligned}
\partial_{\theta_j}\ell'(\theta, x) &= -\lambda\theta_j + \sum_{i=1}^{m} w^{(i)}(x)x_j^{(i)}(y^{(i)}(1 - h_\theta(x^{(i)})) - (1 - y^{(i)})h_\theta(x^{(i)})) \\
&= -\lambda\theta_j + \sum_{i=1}^{m} w^{(i)}(x)x_j^{(i)}(y^{(i)} - h_\theta(x^{(i)})) \\
&= -\lambda\theta_j + \sum_{i=1}^{m} X_{ji}^\top z_i \\
&= -\lambda\theta_j + (X^\top z)_j
\end{aligned}$$

Dropping j from the notation we get an equality of functions $\mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}^N$,

$$\nabla_\theta \ell'(\theta, x) = -\lambda\theta + X^\top z \tag{13}$$

Going further, we get $\partial_{jk}\ell' = \partial_j(X^\top z)_k - \lambda\delta_{jk}$. Note that $\partial_j z_i = -\partial_j h_\theta(x^{(i)}))$. By (12), $\partial_j h_\theta(x^{(i)}) = x_j^{(i)} h_\theta(x^{(i)})(1 - h_\theta(x^{(i)}))$. Together, these equations give

$$\begin{aligned}
H\ell'(\theta, x)_{jk} &= \partial_{jk}\ell'(\theta, x) \\
&= -\lambda\delta_{jk} + \sum_i X_{ki}^\top \partial_j z_i \\
&= (-\lambda I)_{jk} + \sum_i X_{ki}^\top x_j^{(i)} h_\theta(x^{(i)})(1 - h_\theta(x^{(i)})) \\
&= (-\lambda I)_{jk} + \sum_{i,l} X_{ki}^\top h_\theta(x^{(l)})(1 - h_\theta(x^{(l)}))\delta_{il} X_{lj} \\
&= (-\lambda I + X^\top D X)_{jk},
\end{aligned} \tag{14}$$

where $D_{il} = h_\theta(x^{(l)})(1 - h_\theta(x^{(l)}))\delta_{il}$ is a diagonal $m \times m$ matrix.

(a) Implement the Newton-Raphson algorithm for optimizing $\ell(\theta)$ for a new query point x, and use this to predict the class of x.

Solution to 2-(a):

Recall that the Newton-Raphson algorithm is just another name for the update rule

$$\theta := \theta - H^{-1}\nabla_\theta \ell(\theta),$$

used for finding the critical points of $\ell(\theta)$.