Math189R SU17
Homework 4
Wednesday, May 31, 2017

Feel free to work with other students, but make sure you write up the homework and code on your own (no copying homework *or* code; no pair programming). Feel free to ask students or instructors for help debugging code or whatever else, though.

The starter files can be found under the Resource tab on course website. The graphs for problem 2 generated by the sample solution could be found in the corresponding zipfile. These graphs only serve as references to your implementation. You should generate your own graphs for submission. Please print out all the graphs generated by your own code and submit them together with the written part, and make sure you upload the code to your Github repository.

1 (Conditioning a Gaussian) Note that from Murphy page 113. "Equation 4.69 is of such importance in this book that we have put a box around it, so you can easily find it." That equation is important. Read through the proof of the result. Suppose we have a distribution over random variables $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$ that is jointly Gaussian with parameters

$$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \quad \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix},$$

where

$$\mu_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \ \mu_2 = 5, \ \Sigma_{11} = \begin{bmatrix} 6 & 8 \\ 8 & 13 \end{bmatrix}, \ \Sigma_{21}^\top = \Sigma_{12} = \begin{bmatrix} 5 \\ 11 \end{bmatrix}, \ \Sigma_{22} = \begin{bmatrix} 14 \end{bmatrix}.$$

Compute
(a) The marginal distribution $p(\mathbf{x}_1)$.
(b) The marginal distribution $p(\mathbf{x}_2)$.
(c) The conditional distribution $p(\mathbf{x}_1|\mathbf{x}_2)$
(d) The conditional distribution $p(\mathbf{x}_2|\mathbf{x}_1)$

a) $p(\vec{x}_1) = \mathcal{N}(\vec{\mu}_1, \vec{\Sigma}_{11}) = \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 6 & 8 \\ 8 & 13 \end{bmatrix}\right)$

b) $p(\vec{x}_2) = \mathcal{N}(\mu_2, \vec{\Sigma}_{22}) = \mathcal{N}(5, 14)$

c) $p(\vec{x}_1|\vec{x}_2) = \mathcal{N}(\vec{\mu}_{1|2}, \vec{\Sigma}_{11|22})$

$\vec{\mu}_{1|2} = \vec{\mu}_1 + \vec{\Sigma}_{12}\vec{\Sigma}_{22}^{-1}(\vec{x}_2 - \mu_2) = \frac{1}{14}\begin{bmatrix} 5 \\ 11 \end{bmatrix}(\vec{x}_2 - 5)$

$\vec{\Sigma}_{1|2} = \vec{\Sigma}_{11} + \vec{\Sigma}_{12}\vec{\Sigma}_{22}^{-1}\vec{\Sigma}_{21} = \begin{bmatrix} 6 & 8 \\ 8 & 13 \end{bmatrix} - \frac{1}{14}\begin{bmatrix} 5 \\ 11 \end{bmatrix}\begin{bmatrix} 5 & 11 \end{bmatrix} = \begin{bmatrix} 59/14 & 57/14 \\ 57/14 & 61/14 \end{bmatrix}$ ■

d) $p(\vec{x}_2 \mid \vec{x}_1) = \mathcal{N}(\vec{\mu}_{2|1}, \vec{\Sigma}_{2|1})$

↳ $\vec{\mu}_{2|1} = \vec{\mu}_2 + \vec{\Sigma}_{21}\vec{\Sigma}_{11}^{-1}(\vec{x}_1 - \mu_1) = 5 + \begin{bmatrix} 5 & 11 \end{bmatrix} \begin{bmatrix} 6 & 8 \\ 8 & 13 \end{bmatrix}^{-1} (\vec{x}_1 - \mu_1) = 5 + \begin{bmatrix} -\frac{23}{14} & \frac{13}{7} \end{bmatrix} \vec{x}_1$

↳ $\vec{\Sigma}_{2|1} = \vec{\Sigma}_{22} - \vec{\Sigma}_{21}\vec{\Sigma}_{11}^{-1}\vec{\Sigma}_{12} = 14 - \begin{bmatrix} 5 & 11 \end{bmatrix} \begin{bmatrix} 6 & 8 \\ 8 & 13 \end{bmatrix}^{-1} \begin{bmatrix} 5 \\ 11 \end{bmatrix} = \frac{25}{14}$

2 (**MNIST**) In this problem, we will use the MNIST dataset, a classic in the deep learning literature as a toy dataset to test algorithms on, to set up a model for logistic regression and softmax regression. In the starter code, we have already parsed the data for you. However, you might need internet connection to access the data and therefore successfully run the starter code.

The problem is this: we have images of handwritten digits with 28×28 pixels in each image, as well as the label of which digit $0 \leq$ `label` ≤ 9 the written digit corresponds to. Given a new image of a handwritten digit, we want to be able to predict which digit it is. The format of the data is `label, pix-11, pix-12, pix-13, ...` where `pix-ij` is the pixel in the `ith` row and `jth` column.

(a) (**logistic**) Restrict the dataset to only the digits with a label of 0 or 1. Implement L2 regularized logistic regression as a model to compute $\mathbb{P}(y = 1|\mathbf{x})$ for a different value of the regularization parameter λ. Plot the learning curve (objective vs. iteration) when using Newton's Method *and* gradient descent. Plot the accuracy, precision ($p = \mathbb{P}(y = 1|\hat{y} = 1)$), recall ($r = \mathbb{P}(\hat{y} = 1|y = 1)$), and F1-score ($F1 = 2pr/(p + r)$) for different values of λ (try at least 10 different values including $\lambda = 0$) on the test set and report the value of λ which maximizes the accuracy on the test set. What is your accuracy on the test set for this model? Your accuracy should definitely be over 90%.

(b) (**softmax**) Now we will use the whole dataset and predict the label of each digit using L2 regularized softmax regression (multinomial logistic regression). Implement this using gradient descent, and plot the accuracy on the test set for different values of λ, the regularization parameter. Report the test accuracy for the optimal value of λ as well as it's learning curve. Your accuracy should be over 90%.

a) logistic model: $P(y=1 \mid \vec{x}; \vec{\theta}) = \sigma(\vec{\theta}^T \vec{x})$

$$nll(\vec{\theta}) = -\sum_i y_i \log \sigma(\vec{\theta}^T \vec{x}) + (1-y_i) \log(1 - \sigma(\vec{\theta}^T \vec{x})) + \frac{\lambda}{2} ||\vec{\theta}||_2^2$$

gradients:

$$\nabla_\theta \ell = \sum_i y_i (1 - \sigma(\vec{\theta}^T \vec{x})) \vec{x} - (1 - y_i) \sigma(\vec{\theta}^T \vec{x}) \vec{x} + \lambda \vec{\theta}$$

$$= X^T (\sigma(X\vec{\theta}) - \vec{y}) + \lambda \vec{\theta}$$

Hessian:

$$\nabla^2 \ell = \frac{d}{d\theta} \nabla \ell^T$$

$$= \sum_i \nabla_\theta \sigma(\theta^T \vec{x}) \vec{x}_i^T + \lambda I$$

$$= X^T \operatorname{diag}\left[\sigma(X\theta)(1 - \sigma(X\theta)_2)\right] X + \lambda I$$

■

Convergence plot in Github folder

- Newton's method is much faster

Test Metrics plot in Github folder

- mostly all are reasonable

b) softmax regression: $P(y=c|\vec{x}, W) = \frac{1}{Z}\exp(\vec{w}_c^T \vec{x}) = \frac{\exp(\vec{w}_c^T \vec{x})}{\sum_i \exp(\vec{w}_i^T \vec{x})}$

$$nll(W) = -\log \prod_i \prod_c \mu_{ic}^{y_{ic}} - \lambda \operatorname{tr}(W^T C)$$

$$= \sum_i \sum_c y_{ic} \log \mu_{ic} + \lambda \operatorname{tr}(W^T W)$$

$$\nabla_W nll = X^T(\vec{\mu} - \vec{y}) + \lambda W$$

$$y_{ic} = I_n$$

$$\mu_i = S(\vec{x}_i) = \frac{\exp(W^T \vec{x})}{\vec{1}^T \exp(W^T \vec{x})}$$

Accuracy Plot in Github folder

- max test accuracy: 0.9221

$\lambda = 0.01$

Convergence plot in Github folder.