WSU, CS

Applying ADMM for multi-class classification problem.

Artem Komarichev

August 18, 2015

Goal: using logistic regression recognize handwritten digits (from 0 to 9). There is ten classes ($K = 10$)

Dataset: 5000 training examples (let's name it m) and each sample contains 400 features (pixels in a 20x20 grid).

1 Problem formulation

In the case of L2 regularized logistic regression the problem becomes:

$$minimize \quad \frac{1}{m}\sum_{i=1}^{m}(-y\log h_\theta(x) - (1-y)\log(1-h_\theta(x)) + \frac{\lambda}{2m}\sum \theta_j^2 \tag{1.1}$$

where $h_\theta(x) = g(\theta^T x)$ and $g(z) = \frac{1}{1+e^{-z}}$ - sigmoid function.

The ADMM algorithm for the above problem is [1]:

$$\begin{aligned} x^{k+1} &= \arg\min_x \frac{1}{m}\sum_{i=1}^{m} l(x) + \frac{\lambda}{2m}\sum ||x - z^k + u^k||_2^2 \\ z^{k+1} &= S_{\frac{m\lambda}{\rho}}(x^{k+1} + u^k) \\ u^{k+1} &= u^k + x^{k+1} - z^{k+1} \end{aligned} \tag{1.2}$$

where $l(x) = -y \log h_\theta(x) - (1-y)\log(1-h_\theta(x)$.

The x-update is a proximal operator evaluation. And this was solved by using quasi-Newton method (L-BFGS). Correspondingly, the partial derivatives of regularized logistic regression cost for x defined as in [2]:

$$\frac{1}{m}\sum_{i=1}^{m}[x^T(h_\theta(x)-y)+\lambda(x-z^k+u^k)] \tag{1.3}$$

Drawback: As you can notice second step in (1.2) is the soft thresholding operator. We can use shrinkage only in the case when our regulizer is l_1! But I was using the l_2 regularization in the cost function (1.1). So I have to either replace z-update with the proximity operator or replace l_2 on l_1 in the objective function.

2 One-vs-all Classification

I've implemented one-vs-all classification by training multiple regularized logistic regression classifiers, one for each of the K classes in dataset (in our case $K = 10$).

While training the classifire for class $k \in \{1, ..., K\}$, I was creating a $m-$dimensional vector of labels y, where $y_j \in 0, 1$ indicates whether the j^{th} sample belongs to class k ($y_j = 1$), or if it belongs to a different class ($y_j = 0$).

3 One-vs-all Prediction

After the previous step (training one-vs-all classifier), I was able to predict the digit for each sample. Simply by calculating the "probability" that it belongs to each class using already trained classifier. And we should assign an appropriate digit to it based on the highest probability.

4 Conclusion

As I did already mention in one of my previous email. The highest accuracy I got is about 97.36%. Authors of the assignment got the accuracy about 94.9% [2].

5 References

[1] Stephen Boyd. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers, 2011.

[2] Andrew Ng. Online Machine Learning class on Coursera. Programming exercise 3: Multi-class Classification and Neural Networks, 2011.