Machine Learning Mid-term Assignment (Shimosaka)

18M30396 Cho Bunhin

August 3, 2018

Problems done: problem 1(all), problem 3(1,2,3), problem 6(all).
MATLAB codes can be found at
`https://github.com/chobunhin/ART.T458-Machine-Learning-Assignment`

Problem 1

Linear logistic regression with Batch steep descent

- For i-th sample, corresponding function value and gradient read

$$J_i(w) = \ln(1 + \exp(-y_i w^T x_i)), \tag{1}$$

$$\frac{\partial J_i}{\partial w}(w) = -\frac{\exp(-y_i w^T x_i)}{1 + \exp(-y_i w^T x_i)} y_i x_i, \tag{2}$$

- for t-th iteration, we update w by a batch of samples as well as the L^2 penalty:

$$w^{t+1} = w^t - \eta \cdot \left(\sum_{i \in I^t} \frac{\partial J_i}{\partial w} + \lambda w^t \right). \tag{3}$$

Linear logistic regression with AdaGrad method

- Diagonal Hessian

$$H_t = diag(G_t^{1/2}), \tag{4}$$

  where $\{G_t\}_i = \sum_{\tau=1}^{t} \{g^\tau\}_i^2$, and g is the gradient of full $J(w)$.

- Accumulated gradient

$$d_t = \sum_{\tau=1}^{t} g_\tau \tag{5}$$

- Updating rule:

$$w^{t+1} = w^t - \eta_0 H_t^{-1} d_t \tag{6}$$

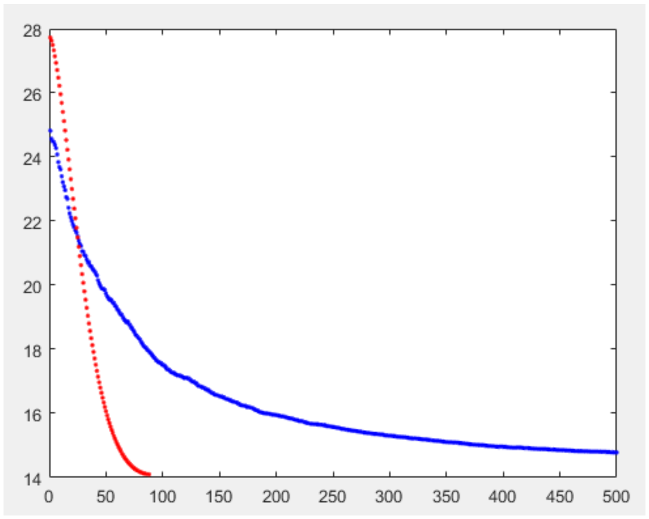

Figure 1: Function values comparison: blue-batch, red-adagrad

MATLAB code see `problem1.m`
Comparisons: Toy Dataset II; Running spec see `problem1.m`; We observed that AdaGrad method converges considerably faster than batch steepest descent method.

Problem 3

proof of 1. The original problem is equivalent to

$$\left\{\begin{array}{ll} \min_{w\in\mathbb{R}^d, z\in\mathbb{R}^n} & z^T\mathbf{1} + \lambda\|\mathbf{w}\|_\mathbf{2}^\mathbf{2} \\ \text{s.t.} & z \geq 0 \\ & z_i + y_i w^T x_i - 1 \geq 0 \end{array}\right. \tag{7}$$

with Lagrangian ($\mu \geq 0, \alpha \geq 0$):

$$L(w, z; \mu, \alpha) = \langle z, \mathbf{1}\rangle + \lambda\langle \mathbf{w}, \mathbf{w}\rangle - \langle \mu, \mathbf{z}\rangle - \langle \alpha, \mathbf{z}\rangle - \sum_{\mathbf{i=1}}^{\mathbf{n}}(\mathbf{y_i}\langle \mathbf{w}, \mathbf{x_i}\rangle - \mathbf{1}). \tag{8}$$

Following standard dual problem generating, we have

$$\frac{\partial L}{\partial w} = 0 \Rightarrow \hat{w} = \frac{1}{2\lambda}\sum_{i=1}^{n} \alpha_i y_i x_i, \tag{9}$$

$$\frac{\partial L}{\partial z} = 0 \Rightarrow \mathbf{1} - \mu - \alpha = \mathbf{0}, \tag{10}$$

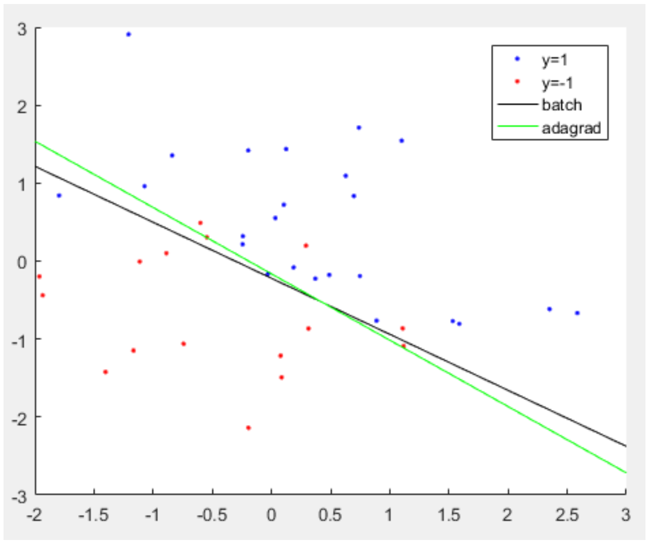

Figure 2: Separation results comparison

and the dual objective function now reads:

$$L(\hat{w}, \hat{z}; \mu, \alpha) = -\frac{1}{4\lambda}\langle \sum_{i=1}^{n} \alpha_i y_i x_i, \sum_{j=1}^{n} \alpha_j y_j x_j \rangle + \langle \alpha, \mathbf{1} \rangle, \tag{11}$$

which gives the conclusion that

$$K_{ij} = y_i y_j \langle x_i, x_j \rangle, \tag{12}$$

and the dual problem reads:

$$\begin{cases} \max_{\alpha \in \mathbb{R}^n} & -\frac{1}{4\lambda}\langle \alpha, K\alpha \rangle + \langle \alpha, \mathbf{1} \rangle \\ \text{s.t.} & \mathbf{0} \leq \alpha \leq \mathbf{1} \end{cases} \tag{13}$$

$\square$

2. KKT condition $\Rightarrow \frac{\partial L}{\partial w} = 0 \Rightarrow \hat{w} = \frac{1}{2\lambda} \sum_{i=1}^{n} \alpha_i y_i x_i$ according to (9). $\square$

3. MATLAB code see `problem3.m`
Settings: Toy Dataset II; number of samples: 60; learning rate $\eta_t = \eta = 0.005$; iterate 1000 times. The primal-dual objective values as well as final result are shown in figure 3 and figure 4.

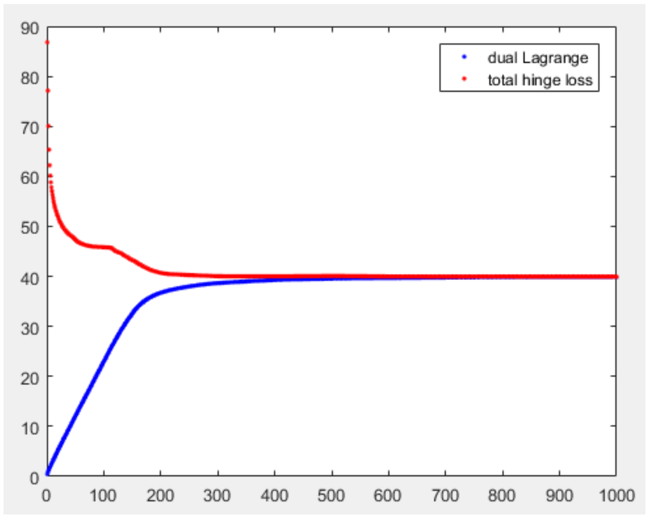

Figure 3: primal dual objective value wrt. iterate

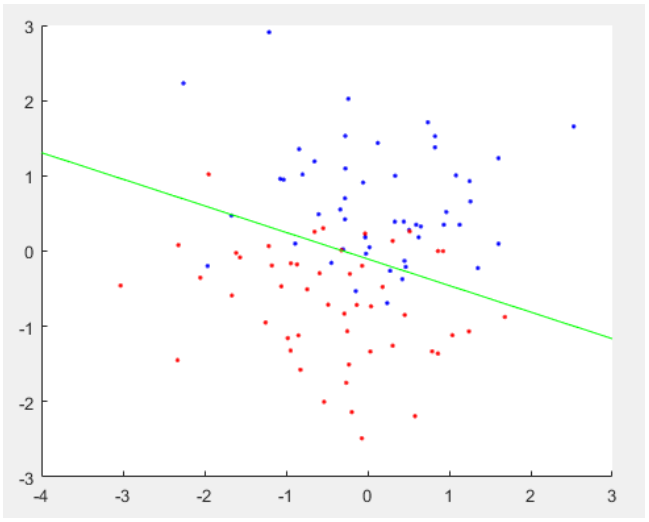

Figure 4: result given by SVM

Problem 6

1. nuclear norm For matrix $Z \in \mathbb{R}^{m\times n}$, let the singular values of Z be

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min(m,n)}, \tag{14}$$

then the nuclear norm of Z reads

$$\|Z\|_* = \sum_{j=1}^{\min(m,n)} \sigma_j. \tag{15}$$

□

2. proximal operator of nuclear norm Let $Z = U\Sigma V^T$ be the SVD of Z, where $\Sigma = diag\{\sigma_1, \sigma_2, \ldots, \sigma_{\min(m,n)}\}$. Then proximal operator

$$\begin{aligned} Prox_{\lambda\|\cdot\|_*}(Z) &= \arg\min_X \frac{1}{2}\|X - Z\|_F^2 + \lambda\|X\|_* && (16)\\ &= U \cdot diag\{S_\lambda(\sigma_1), \ldots, S_\lambda(\sigma_{\min(m,n)})\} \cdot V^T, && (17)\end{aligned}$$

where soft shrinkage $S_\lambda(\sigma) = \max(\sigma - \lambda, 0)$. □

3. Proximal gradient method implementation:

- Objective function:

$$f(Z) = \sum_{i,j\in Q} |A_{i,j} - Z_{i,j}|^2 + \lambda\|Z\|_* \equiv g(Z) + h(Z), \tag{18}$$

- Gradient of smooth part:

$$\big(\nabla g(Z)\big)_{i,j} = \begin{cases} 0 & (i,j) \notin Q \\ 2(Z_{i,j} - A_{i,j}) & (i,j) \in Q, \end{cases} \tag{19}$$

  which infers that $\|\nabla g(Z_1) - \nabla g(Z_2)\|_F \leq \gamma\|Z_1 - Z_2\|_F$ and Lipschitz constant $\gamma = 2$.

- Updating rule:

$$\begin{aligned} Z^{(t+1)} &= \arg\min_Z \left\{\langle \nabla g(Z^{(t)}), Z - Z^{(t)}\rangle + \frac{\gamma}{2}\|Z - Z^{(t)}\|_F^2 + h(Z)\right\} && (20)\\ &= \arg\min_Z \left\{\langle \nabla g(Z^{(t)}), Z - Z^{(t)}\rangle + \|Z - Z^{(t)}\|_F^2 + \lambda\|Z\|_*\right\} && (21)\\ &= \text{Prox}_{\frac{\lambda}{2}\|\cdot\|_*}\left(Z^{(t)} - \frac{1}{2}\nabla g(Z^{(t)})\right) && (22)\end{aligned}$$

- Parameter tunning: $\lambda = 0.05$; 200 iterations; all-zero initial guess.

- For recovered low-rank matrix see figure 5.

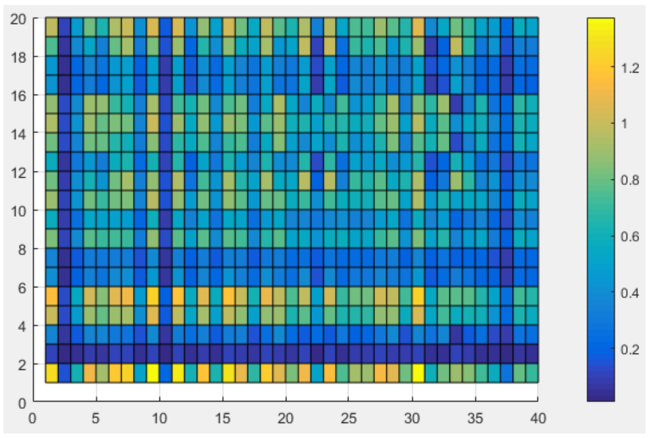

Figure 5: Surf plot of recovered matrix given by nuclear norm regularization with proximal gradient method

4. Non-negative matrix factorization and comparison. MATLAB code see `problem6.m`

- Hierarchical alternating least squares (HALS) for NMF solves

$$\min \|A - \sum_{k=1}^{r} u_k v_k^T\|_Q^2 \tag{23}$$

alternatively w.r.t non-negative u, v and successively w.r.t subscript k. Each update can be solved in close form as shown below. We only give formula on u_k.

$$f(u_k) = \|R_k - u_k v_k^T\|_Q^2 \tag{24}$$
$$= \sum_{i,j\in Q} \Big[(R_k)_{i,j} - (u_k)_i (v_k)_j\Big]^2, \tag{25}$$

$\partial f / \partial (u_k)_i = 0$ yields (also considering non-negativity)

$$(u_k)_i = \max\Big(0, \frac{\sum_{i,(i,j)\in Q} (v_k)_j \cdot (R_k)_{i,j}}{\sum_{i,(i,j)\in Q} (v_k)_j^2}\Big), \tag{26}$$

where residual matrix

$$R_k = A - \sum_{l\neq k} u_l v_l^T. \tag{27}$$

- Parameter setting: rank for factorization $r = 3$ (rank of A is 2); 200 iterations; random initial guess with uniform dist on $[0, 1]$; Dataset III.
- Relative error comparison with PG is recorded in figure 6.

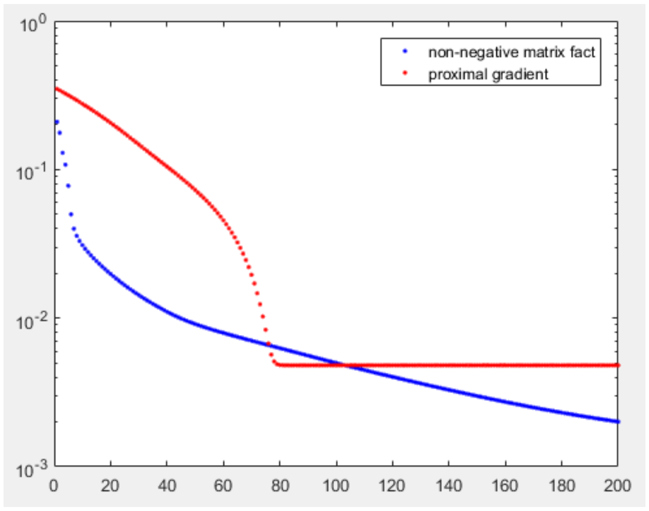

Figure 6: relative error comparison of HALS and PG

- Analysis for HALS: simple updating rule (fixed-point iteration); linear convergence to true matrix; tricky initial guess since the iteration fails when u_k or v_k becomes zero.
- Analysis for PG: solving convex problem meaning convergence will not be a problem; not necessarily gives exact recovery; keep low-rank property well.

Suggestion

The pdf slides of the lectures are a little bit too large, which makes it difficult for viewing the slides in computer. Also it would be thankful if searching function is available in the slides. Many thanks to Prof. Shimosaka for giving this excellent lecture.