ART.T458 Advanced Machine Learning Midterm Assignment (Final report assigned by Shimosaka)

Masamichi Shimosaka
Department of Computer Science
Tokyo Institute of Technology

Instruction

- Choose the problems from your preference and solve the chosen problems.
- Use Tex / MS word.
- Submit the A4 sized / letter sized report (pdf format file) by **8th 17:00 August 2020 via OCW-i**.
- The maximum points you can earn in each problem is shown in each section title.
- Let $s_i \geq 0$ be the score you earned in i-th problem, the final score used in the evaluation q is processed by
$$q = \min(30, \sum_i s_i).$$
- You could earn 2 pts (that is independent of q), if you show constructive criticism to my part of this course. Here word *constructive* does not indicate that 誤字脱字が多くてわかりにくかったです without any specifications, but x 回目の講義のスライド y ページ目の z の式に誤りがあり，w のように修正すべきです. Suggestions to improve the quality of the course, of course, are welcome.
- You do not need to include the source code of your implementation in your report. However, you are welcome to show the URL of your codes via github or some other publicly available repository services.
- IMPORTANT: In this report, you are not allowed to use high level machine learning libraries, such as SciPy, scikit-learn, (Py)-Torch, Tensor Flow, and Neural network toolbox in Matlab, but are allowed to use basic linear algebra libraries such as NumPy, and basic matlab language functionalities. It should be noted that the main objective of this report is to understand mathematical perspective of ML with implementations instead of how to use (black box) ML libraries.
- You can use **Japanese** as well as English.
- Some reference code in Jupyter notebook in `https://bit.ly/32gbpRt` might be helpful to promote this assignment and might also be updated frequently. You are allowed not to use this script but create your own code from the start. Of course, you could use Matlab as well as Python.

Problem 1 (10 pts)

We consider a binary classification with a linear logistic regression. Let $\boldsymbol{x} \in \mathbb{R}^d$, and $\boldsymbol{w} \in \mathbb{R}^d$ be an d-dimensional input vector, and a parameter of the model, respectively. The classifier is represented by $f(\boldsymbol{x}) = 2[\![\boldsymbol{w}^\top \boldsymbol{x} > 0]\!] - 1$, where $[\![c]\!]$ denotes an indicator function that returns 1 if c is true, otherwise returns 0. With the supervised dataset $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$, we consider an optimization problem for the logistic regression. The optimization problem can be written as

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmin}}\, J(\boldsymbol{w})$$

$$J(\boldsymbol{w}) := \sum_{i=1}^{n} \left(\ln\left(1 + \exp(-y_i \boldsymbol{w}^\top \boldsymbol{x}_i)\right)\right) + \lambda \boldsymbol{w}^\top \boldsymbol{w}.$$

Here we assume that $\boldsymbol{x}_i$ contains constant value 1 to make the classifier adaptable to offset in $d-1$ dimensional feature space. With some artificial dataset (see Toy Dataset section, Dataset IV), we consider to implement some optimization methods in the following way.

1. Implement batch steepest gradient method[1].
2. Implement Newton based method.
3. Compare the performance of the above two optimization methods by showing $|J(\boldsymbol{w}^{(t)}) - J(\hat{\boldsymbol{w}})|$ w.r.t. t, where $\boldsymbol{w}^{(t)}$ represents the parameter at t-th iteration, and $\hat{\boldsymbol{w}}$ represent optimal parameter that reaches minimum of J obtained by (either of) the two methods, in semi-log plot.
4. Implement Newton method and simple steepest gradient method for multiclass version of logistic regression (use Toy dataset V) and run the same experiment as binary logistic regression mentioned above.

Problem 2 (10 pts)

We consider *lasso*, where the square loss, and the L1 regularization are employed for linear regression. In this problem, we employ proximal gradient method (PG). So as to make the discussion simple, we use the following objective:

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \left((\boldsymbol{w} - \boldsymbol{\mu})^\top \boldsymbol{A} (\boldsymbol{w} - \boldsymbol{\mu}) + \lambda ||\boldsymbol{w}||_1 \right).$$

Implement PG for lasso and show the results in a couple of conditions. In this question, use the same learning rate $\eta_t = L^{-1}$, where L depicts the Lipsitz constant of the gradient of the objective, which is derived from the Hessian matrix $2\boldsymbol{A}$ (i.e. use the maximum eigen value of $2\boldsymbol{A}$ as the inverse of the learning rate: η_t^{-1}).

1. Show the result of PG in terms of $||\boldsymbol{w}^{(t)} - \hat{\boldsymbol{w}}||$ w.r.t. the number of iteration. Use semi log plot. Use the following condition:

   $$\boldsymbol{A} = \begin{pmatrix} 3 & 0.5 \\ 0.5 & 1 \end{pmatrix}, \boldsymbol{\mu}^\top = \begin{pmatrix} 1 & 2 \end{pmatrix}.$$

   To verify the property of L1 regularization, run the experiment with $\lambda = 2, 4, 6$. Recall that the numerical result can be found in the slide used in the lecture. You can also verify the result by using cvx (matlab) / cvxopt (python).
2. Investigate concept of regularization path and plot regularization path of this lasso. Specifically, we assume that report includes figure that shows trajectories of optimal parameter w_1, w_2 w.r.t. hyper parameter λ. You do not need to implement efficient regularization path generator but are asked to just iterate lasso with multiple λ then store the optimal parameters then show them in the figure. Discuss the property of this regularization path.
3. Investigate group lasso and implement it with proximal gradient method, then compare the optimization result with cvx (or cvxopt / cvxpy). You should clarify proximal operation for group regularization. (see Toy Dataset VI and its sample code if necessary).

Problem 3 (10 pts + optional 10 pts)

We consider the dual of the support vector machine (L2-regularized hinge loss based binary classifier). The original optimization problem of this classification can be represented as

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w} \in \mathbb{R}^d}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \max(0, 1 - y_i \boldsymbol{w}^\top \boldsymbol{x}_i) + \lambda ||\boldsymbol{w}||_2^2 \right), \tag{1}$$

[1] We assume here the learning rate is constant for simplicity. Consider the upper bound of Lipsitz constant of the gradient of this objective.

where $\boldsymbol{x}_i \in \mathbb{R}^d$, $\boldsymbol{w} \in \mathbb{R}^d$, $y_i \in \{\pm 1\}$, and $\lambda > 0$ denotes i-th input variable, the parameter vector, and the label for i-th input data, and a coefficient of the regularization term, respectively.

1. Verify that the dual Lagrange function of this optimization can be written as

$$\begin{array}{rl} \text{maximize}_{\boldsymbol{\alpha} \in \mathbb{R}^n} & -\frac{1}{4\lambda}\boldsymbol{\alpha}^\top \boldsymbol{K}\boldsymbol{\alpha} + \boldsymbol{\alpha}^\top \mathbf{1} \\ \text{subject to} & \mathbf{0} \leq \boldsymbol{\alpha} \leq \mathbf{1} \end{array}, \tag{2}$$

   where $\mathbf{1}$ and $\mathbf{0}$ denote a n dimensional vector whose elements are all 1 and 0, respectively, and $\boldsymbol{K} \in \mathbb{R}^{n \times n}$ denotes a symmetric square matrix, and its i-th row and j-th column element can be represented by $y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j$.

2. From the KKT condition, verify the optimal weight parameter $\boldsymbol{w}$ given by $\boldsymbol{\alpha}$ can be written as

$$\hat{\boldsymbol{w}} = \frac{1}{2\lambda}\sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \tag{3}$$

3. Implement minimization for the negative dual Lagrange function using projected gradient (for simplicity). For the validity, show the score of the dual Lagrange function (use (2)), and sum of hinge loss function and the regularization, w.r.t. number of iteration (use (1) and (3)), respectively. Confirm that the duality gap reaches 0 for the convergence. Here we assume projected gradient just computes

$$\boldsymbol{\alpha}^{(t)} = P_{[0,1]^n}\left(\boldsymbol{\alpha}^{(t-1)} - \eta_t\left(\frac{1}{2\lambda}\boldsymbol{K}\boldsymbol{\alpha}^{(t-1)} - \mathbf{1}\right)\right),$$

   where η_t represents learning rate at t-th iteration, and $P_{[0,1]^n}$ depicts a projection operator that each of the input cast into $[0, 1]$.

4. (Option: You can earn additional 10 pts) Investigate optimization algorithm known as dual coordinate descent technique used in LibLinear and sequential minimal optimization (SMO) also used in LibSVM, and summarize the basic concept of the two algorithms. Then run the three algorithms including projected gradient, SMO, and dual coordinate descent. Finally compare the performance in terms of the convergence speed.

Problem 4 (10 pts + optional 10 pts)

We consider a binary classification problem, where the hinge loss function, and L1 regularization are leveraged. Let $\boldsymbol{x} \in \mathbb{R}^d$ be an input, and $\boldsymbol{w} \in \mathbb{R}^d$ be a parameter of the model, respectively. We consider a binary discriminant function with linear regression as $f(\boldsymbol{x}) = 2[\![\boldsymbol{w}^\top \boldsymbol{x} \geq 0]\!] - 1$, where $[\![c]\!]$ returns 1 when c is true, otherwise it returns 0, and With a supervised dataset $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$, the learning problem can be formalized as

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w} \in \mathbb{R}^d}{\text{argmin}}\left(\sum_{i=1}^{n} \max(0, 1 - y_i \boldsymbol{w}^\top \boldsymbol{x}_i) + \lambda ||\boldsymbol{w}||_1\right), \tag{4}$$

where $y_i \in \{\pm 1\}$ denotes a binary label for i-th training example.

1. Derive a linear program from (4) (with auxiliary variable $\xi_i \geq \max(0, 1 - y_i \boldsymbol{w}^\top \boldsymbol{x}_i) \geq 0$, and $e_i \geq |w_i| \geq 0$). Recall that linear program can be written as

$$\begin{array}{rl} \text{minimize}_{\boldsymbol{z}} & \boldsymbol{c}^\top \boldsymbol{z} \\ \text{subject to} & \boldsymbol{A}\boldsymbol{z} \leq \boldsymbol{b} \end{array}$$

   (See *LpBoost* that deals with LP from L1-regularized hinge loss model)

2. By using some artificial dataset (see Toy Dataset section, Dataset IV), implement this problem via cvx (in matlab) / cvxopt (in python) (just for reference) and a (batch) proximal subgradient method. Then confirm that the parameter properly converges. The specification of the dataset should be described in the report.

3. (Option: You can earn additional 10 pts) Derive the dual Lagrange function of this linear program, then investigate *LpBoost* for automated featurization by using the dual problem, and the process how to find relevant features in a boosting way. Implement this dual program by a column generation technique. For simplicity, use Dataset II described in this document and use decision stump as single weak learner, a candidate of the feature. Show the classification accuracy as well as the number of features used in the model and the number of support vectors.

Problem 5 (10 pts)

We consider a binary nonlinear classification problem by using some nonlinear transformation and linear classifier. Let $\boldsymbol{x}_i \in \mathbb{R}^d, y_i \in \{\pm 1\}$ be i-th training example in n sized dataset, i.e. we have a supervised dataset $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$, and $\boldsymbol{\phi}(\cdot) : \mathbb{R}^d \to \mathbb{R}^{\tilde{d}}$ be a feature map from the original d dimensional feature space to $\tilde{d}$ dimensional feature space. For example, use Gaussian kernel as the feature map such as

$$\{\boldsymbol{\phi}(\boldsymbol{x})\}_k = \exp\left(-\alpha ||\boldsymbol{x} - \boldsymbol{x}_{z_k}||_2^2\right),$$

where $\{\boldsymbol{\phi}(\boldsymbol{x})\}_k$ represents a k-th feature map among $\tilde{d}$ dimensional feature maps, z_k represents an index of the training and $\alpha > 0$ is a hyper parameter of the kernel. It should be noted that z_k is chosen by the randomly permuted sequence of indices. Solve the following problems using Dataset I (see Toy Dataset section).

1. Declare machine learning problem within linear binary classification. Use Gaussian kernel for handling nonlinear separation. Specify the loss function, regularization, respectively.
2. Implement, and run the ml problem as you declare. Compare the performance of the model by modifying the size of n, $\tilde{d} < n$, and α.

Problem 6 (10 pts + optional 5 pts)

We consider a matrix optimization problem with the following objective:

$$\underset{\boldsymbol{Z} \in \mathbb{R}^{m \times n}}{\operatorname{argmin}} \left(\sum_{i,j \notin Q} |A_{i,j} - Z_{i,j}|^2 + \lambda ||\boldsymbol{Z}||_* \right)$$

where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ represents data matrix (user vs. movie rating data in recommendation systems), and also contains null value on Q denotes. $|| \cdot ||_*$ represents a nuclear norm of the matrix, $\lambda > 0$ denotes a hyper parameter for the regularization. In this optimization problem, $\boldsymbol{Z}$ corresponds to the recovered data from the incomplete data matrix $\boldsymbol{A}$. In the scenario of the recommendation systems, the inferred $\boldsymbol{Z}$ at location Q corresponds to the inferred score of movie rating.

1. Describe the definition of the nuclear norm of a matrix by investigating it from www.
2. Define the proximal operation with the nuclear norm. (Hint: Use singular value decomposition.)
3. With some dataset (see Toy Dataset section), implement proximal gradient method for this machine learning problem and shows the recovered data $\boldsymbol{Z}$ by using surface plotting.
4. (Option: You can earn additional 5 pts) Implement non-negative matrix factorization as alternative approach to recover $\boldsymbol{A}$ and compare the performance by choosing the hyper parameters. Discuss the advantages and disadvantages of the two methods you implement here.

Problem 7 (10 pts)

In this problem, we consider recent advances in adaptive gradient based techniques.

1. Pick up at least 3 algorithms on adaptive gradient methods other than AdaGrad, Adam methods, which are explained in the lecture. It should be noted that you need reference (including authors, titles, and the conference information). Since the derived procedure depends on the form of regularization, you need to declare the types of regularization.
2. Implement the three algorithms mentioned above instead of using high level machine learning library.
3. Compare the performance of the three algorithms with Adam, AdaGrad.

Problem 8 (10 pts + optional 30 pts)

We consider an extension of regularization design in terms of structured sparsity.

1. Investigate structured sparsity and summarize the application with structured sparsity.
2. Investigate and summarize the definition of machine learning with structured sparsity such as overlapped group lasso. Recall that group lasso is the simplest machine learning problem on structured sparsity.
3. (Option: You can earn additional 30 pts) Investigate and implement alternating direction multiplier methods (ADMM) for solving the overlapped group lasso with some toy dataset.

Toy Datasets

Use the following datasets for some problems, if necessary. It might be better to give seed to a random number generator in the initialization step. Though the following codes are described in matlab, MS hopes students do not have difficulty in generating the similar dataset in python or other programming language implementation.

Dataset I

```
n = 100;
x = 3 * (rand(n, 2) - 0.5);
radius = [x(:, 1).^2 + x(:, 2).^2];
y = (radius > 0.7 + 0.1 * randn(n, 1)) & (radius < 2.2 + 0.1 * randn(n, 1));
y = 2 * y -1;
```

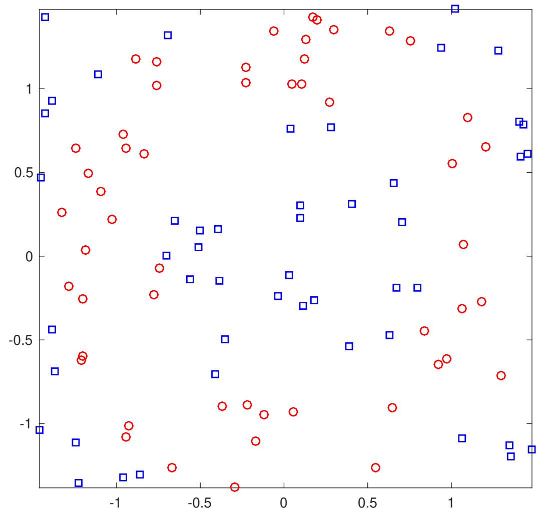

Figure 1: Dataset I

Dataset II

```
n = 40;
omega = randn(1, 1);
noise = 0.8 * randn(n, 1);

x = randn(n, 2);
y = 2 * (omega * x(:, 1) + x(:, 2) + noise > 0) - 1;
```

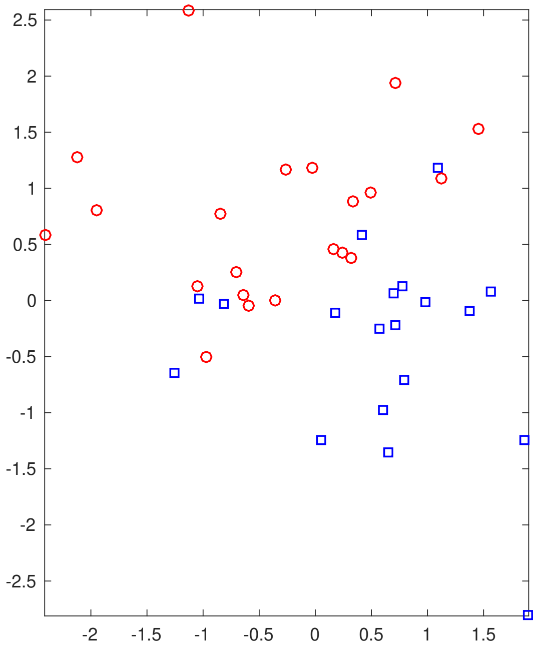

Figure 2: Dataset II

Dataset III

```
m = 20;
n = 40;

r = 2;

A = rand(m, r) * rand(r, n);

ninc = 100;

Q = randperm(m * n, ninc);

A(Q) = NaN;
```

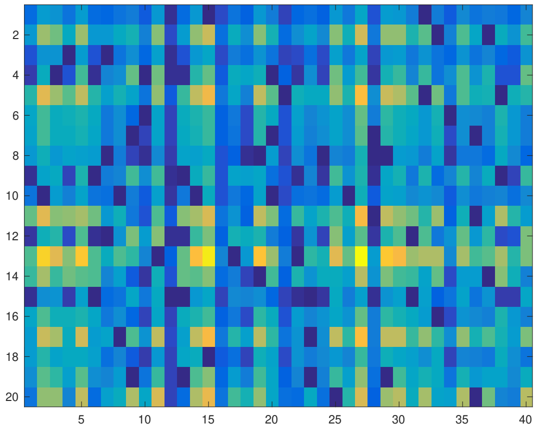

Figure 3: Dataset III

Dataset IV

```
n = 200;
x = 3 * (rand(n, 4) - 0.5);
```

```
    y = (2 * x(:, 1) - 1 * x(:,2) + 0.5 + 0.5 * randn(n, 1)) > 0;
    y = 2 * y -1;
```

Dataset V

```
    n = 200;
    x = 3 * (rand(n, 4) - 0.5);
    W = [2, -1, 0.5;
        -3, 2, 1;
         1, 2, 3];

    [maxlogit, y] =max( [x(:, 1:2), ones(n, 1)] * W' + 0.5 * randn(n, 3), [], 2);
```

Dataset VI and sample script for group lasso

```
clear all;

d = 200;
n = 180;


% we consider 5 groups where each group has 40 attributes
g = cell(5, 1);
for i = 1:length(g)
    g{i} = (i-1)*40+1:i*40;
end

x = randn(n, d);
noise = 0.5;

% we consider feature in group 1 and group 2 is activated.
w = [20 * randn(80, 1);
    zeros(120, 1);
    5 * rand];

x_tilde = [x, ones(n, 1)];

y = x_tilde * w + noise * randn(n, 1);

lambda = 1.0;

wridge = (x_tilde'*x_tilde + lambda * eye(d+1))\(x_tilde' * y);


cvx_begin
variable west(d+1,1)
minimize( 0.5 / n * (x_tilde * west - y)' * (x_tilde * west - y) + ...
    lambda * (norm(west(g{1}), 2.0) + ...
    norm(west(g{2}), 2.0) + ...
    norm(west(g{3}), 2.0) + ...
    norm(west(g{4}), 2.0) + ...
    norm(west(g{5}), 2.0) ))
cvx_end

x_test = randn(n, d);
x_test_tilde = [x_test, ones(n, 1)];

y_test = x_test_tilde * w + noise * randn(n, 1);

y_pred = x_test_tilde * west;


mean((y_pred - y_test) .^2)

figure(1);
clf;
plot(west(1:d), 'r-o')
```

```
hold on
plot(w, 'b-*');
plot(wridge, 'g-+');

legend('group lasso', 'ground truth', 'ridge regression')

figure(2);
clf;
plot(y_test, y_pred, 'bs');
xlabel('ground truth')
ylabel('prediction')

fprintf('carinality of w hat: %d\n', length(find(abs(west) < 0.01)))
fprintf('carinality of w ground truth: %d\n', length(find(abs(w) < 0.01)))
```