CSE512 Fall 2018 Machine Learning - Homework 4

Your Name: Caitao Zhan

Solar ID: 111634527

NetID email address: caitao.zhan@stonybrook.edu

Names of people whom you discussed the homework with: Ting Jin

1 Support Vector Machines

1.1 Linear case

Assume we haved learned the α's and b in the original input space

$$\mathbf{w}^T\mathbf{x} + b = \sum_{i=1}^{n} \alpha_i y^i \left\langle \mathbf{x}^i, \mathbf{x} \right\rangle + b \tag{1}$$

The prediction function of linear SVM:

$$f(\mathbf{x}; \mathbf{w}, b) = \begin{cases} 1 & \text{iff } (1) > 0 \\ -1 & \text{otherwise} \end{cases}$$

LOOCV error:

$$\frac{1}{n} \sum_{i=1}^{n} \delta(y^i, f(\mathbf{x}; \mathbf{w}^{-i}, b^{-i}))$$

where the superscript $-i$ denotes the parameters we found by removing the ith training example, and δ is an indicator function. Consider two cases:

1. Removing a support vector data point. The ith data point lies on the margin, and might be classified wrong. Because for such points, $\alpha_i > 0$, and might affect equation (1).

2. Removing a non-support vector data point. The ith data point lies outside the margin, and will be classified correctly for sure. Because for such points, $\alpha_i = 0$, and will not affect equation (1)

For case NO. 1, let's consider the worst case, that all m support vectors are classified wrong. This worst case leads to the upper bound of the LOOCV error $= \frac{m}{n}$

1.2 General case

The bound will still hold.

The definition of a kernel:

$$K(x, z) = \phi(x)^T \phi(z)$$

Then, everywhere we previously had $\langle \mathbf{x}, \mathbf{z} \rangle$ in our algorithm, we replace it with $K(x, z)$

Now assume we haved learned the (new) α's and b in the high dimensional feature space by using the kernel trick.

$$\mathbf{w}^T \phi(\mathbf{x}) + b = \sum_{i=1}^{n} \alpha_i y^i K(\mathbf{x}^i, \mathbf{x}) + b \tag{2}$$

The prediction function of general SVM:

$$f(\phi(\mathbf{x}); \mathbf{w}, b) = \begin{cases} 1 & \text{iff } (2) > 0 \\ -1 & \text{otherwise} \end{cases}$$

LOOCV error:

$$\frac{1}{n}\sum_{i=1}^{n}\delta(y^i, f(\phi(\mathbf{x}); \mathbf{w}^{-i}, b^{-i}))$$

where the superscript $-i$ denotes the parameters we found by removing the ith training example, and δ is an indicator function. Consider two cases:

1. Removing a support vector data point. The ith data point lies on the margin, and might be classified wrong. Because for such points, $\alpha_i > 0$, and might affect equation (2).

2. Removing a non-support vector data point. The ith data point lies outside the margin, and will be classified correctly for sure. Because for such points, $\alpha_i = 0$, and will not affect equation (2)

For case NO. 1, let's consider the worst case, that all m support vectors are classified wrong. This worst case leads to the upper bound of the LOOCV error $= \frac{m}{n}$

2 Implementation of SVMs

1. First, we need to check the API of quadprog in Matlab using: **doc quadprog**. We see that quadprog solves minimization problem. The dual form of SVM is a maximization problem, we can turn it into a minimization by multiplying -1:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i\alpha_i y_j\alpha_j K(x_i, x_j) - \sum_{j=1}^{n}\alpha_j \\ \text{s.t.} \quad & \sum_{j=1}^{n} y_j\alpha_j = 0 \\ & 0 \le \alpha_j \le C \ \forall j \end{aligned} \tag{3}$$

Let the data has d input attributes, and n number of samples.
Let the input training data be $\mathbf{X}$ shape(d, n), the labels be $\mathbf{y}$ shape $(n, 1)$.
By further comparing equation (3) and quadprog's API, we can infer the following:

$$\begin{aligned} \boldsymbol{x} &= \begin{bmatrix} \alpha_1 \\ ... \\ \alpha_n \end{bmatrix} \\ \boldsymbol{H} &= \begin{bmatrix} y_1 y_1 \mathbf{x_1}\mathbf{x_1}, \cdots, y_1 y_n \mathbf{x_1}\mathbf{x_n} \\ \cdots \\ y_n y_1 \mathbf{x_n}\mathbf{x_1}, \cdots, y_n y_n \mathbf{x_n}\mathbf{x_n} \end{bmatrix} = \mathbf{y}\mathbf{y}^T \times \mathbf{X}^T\mathbf{X} \\ \boldsymbol{f} &= \begin{bmatrix} -1 \\ ... \\ -1 \end{bmatrix} \text{ shape=}(n, 1) \\ \boldsymbol{A} &= [\ ] \\ \boldsymbol{b} &= [\ ] \\ \boldsymbol{Aeq} &= \begin{bmatrix} y_1, \cdots, y_n \end{bmatrix} = \mathbf{y}^T \\ \boldsymbol{beq} &= 0 \\ \boldsymbol{lb} &= \begin{bmatrix} 0 \\ ... \\ 0 \end{bmatrix} \text{ shape=}(n, 1) \\ \boldsymbol{ub} &= \begin{bmatrix} C \\ ... \\ C \end{bmatrix} \text{ shape=}(n, 1) \end{aligned}$$

2. See code in **question_2.m**

3. See code in **question_2.m**

4. I set my eps = eps('single'), around 10e-7. So slack = 0 when slack < eps.
   See table 1 and figure 1.

Accuracy	0.97275
Objective	26.8491
Number of SV	339

Table 1: C = 0.1

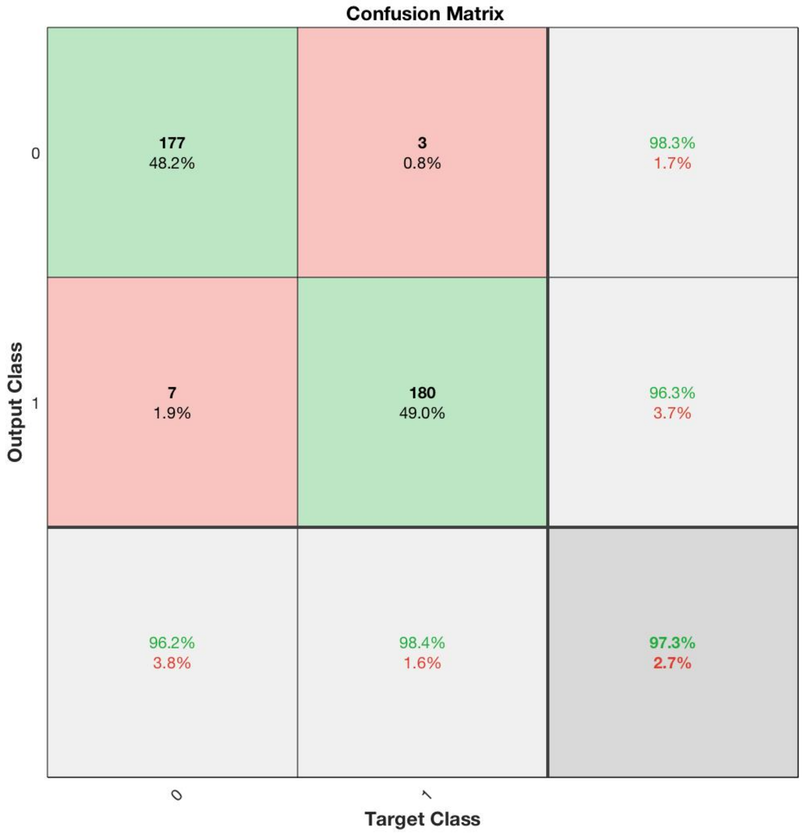

Figure 1: C = 0.1

5. See table 2 and figure 2.

Accuracy	0.9782
Objective	624.5045
Number of SV	125

Table 2: C = 10

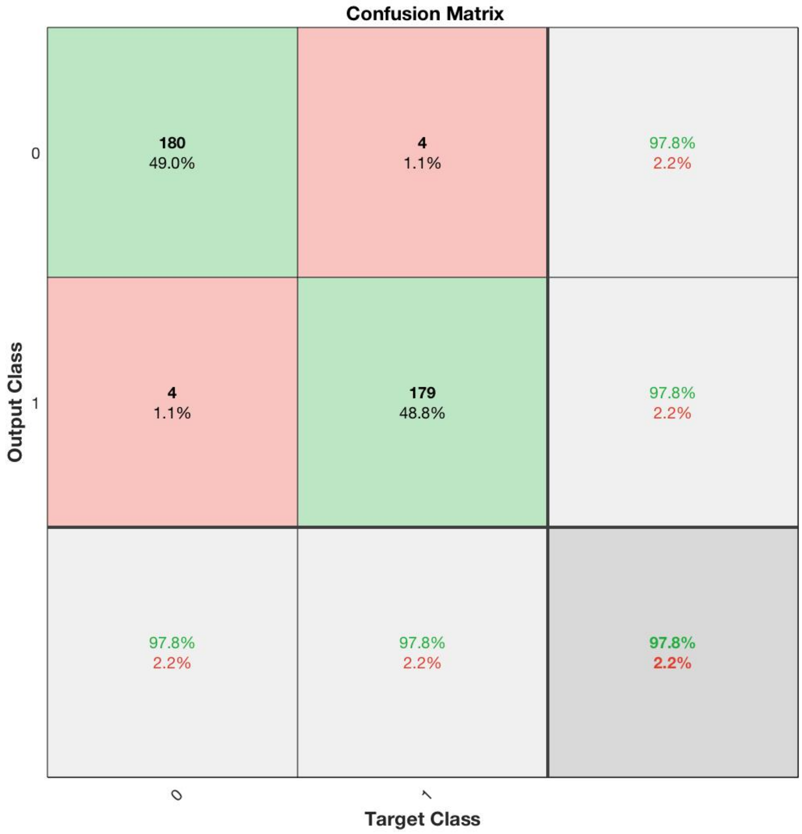

Figure 2: C = 10

6. Implemented the multi-SVM by the one-versus-rest approach. Tried both linear kernel and radius basis function kernel. My best accuracy is 0.74399, using linear kernel with C = 10.

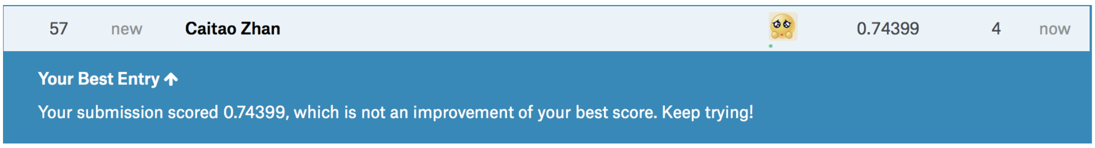

Figure 3: Kaggle Submit

3 Question 3 - SVM for Object Detection

1. AP = 0.6351

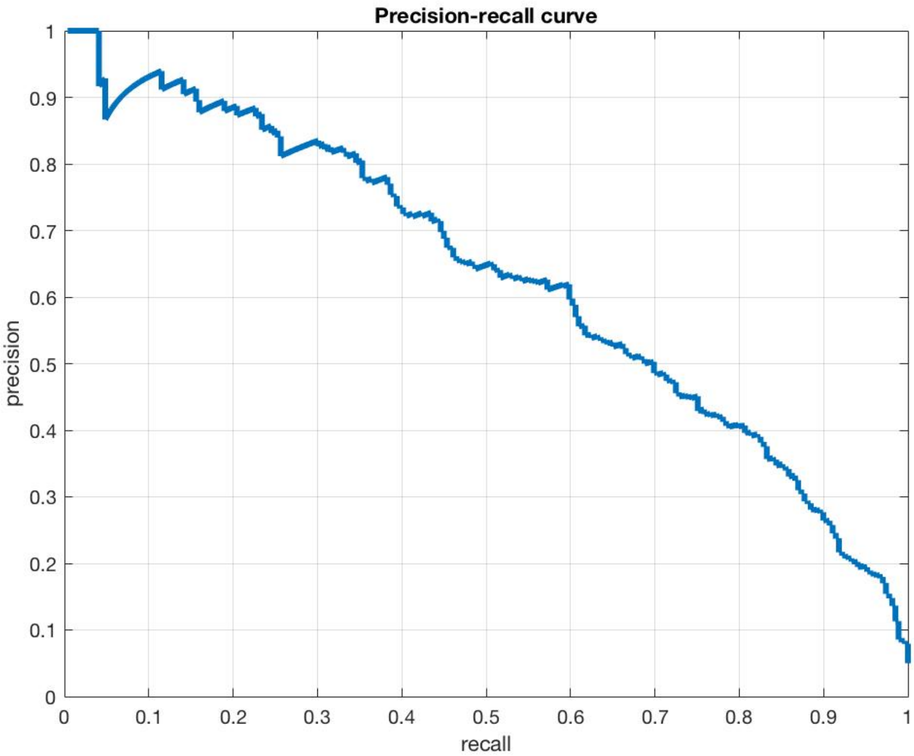

Figure 4: Precision Recall Curve, C = 10, linear kernel

2. I implemented the following hard negative mining algorithm. My m = 548. $548 = 176 + 93 \times 4$, where 176 is the positive, others are the negative. I used a linear kernel. C = 10. I set my eps = eps('single'), around 10e-7. So slack = 0 when slack < eps. Honestly speaking, my objective went down a some point. I know this should let to decrease some points :(But my average precision is pretty good. Hope not lose too much points on the objective values. Thanks TA.

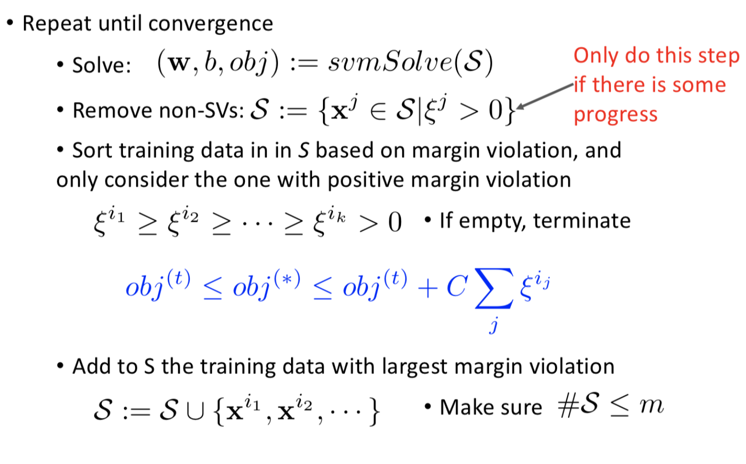

Figure 5: Hard Negative Mining

Record the objective values on training data:

0	132.64
1	2635.30
2	3452.10
3	2577.90
4	2562.60
5	2514.50
6	2559.70
7	3113.90
8	5301.10
9	6267.50
10	6309.50

Figure 6: Objective Value

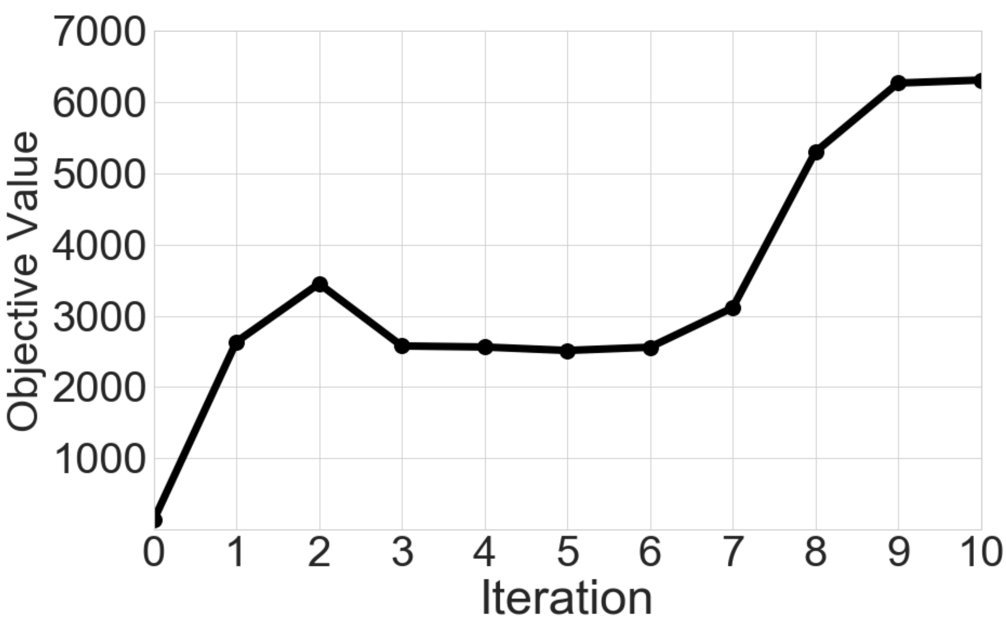

Figure 7: Objective Value

Record the APs on validation data:

0	0.66465
1	0.73465
2	0.80345
3	0.81205
4	0.81868
5	0.81353
6	0.81398
7	0.81172
8	0.81832
9	0.83741
10	0.84199

Figure 8: Average Precision

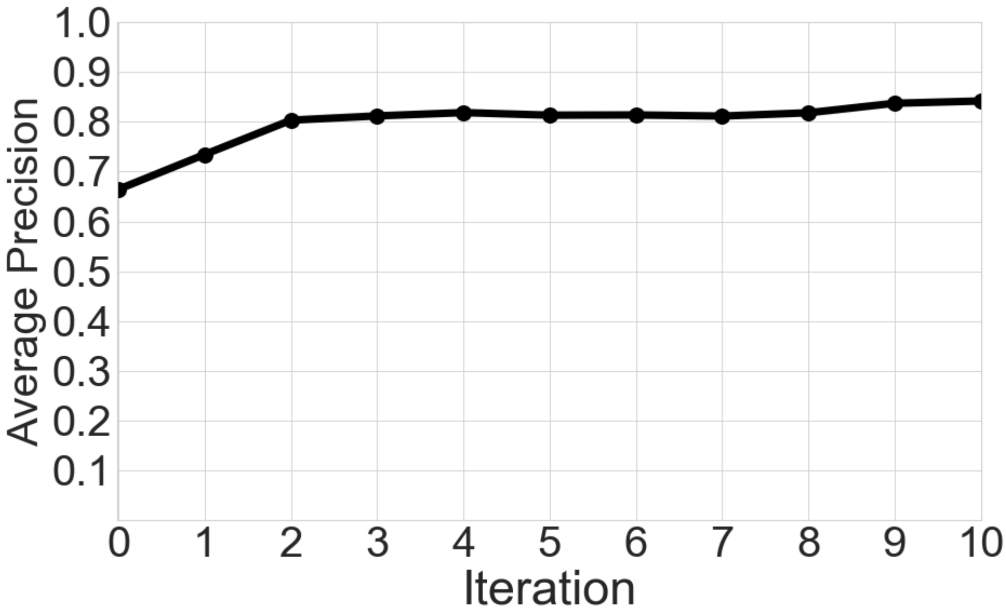

Figure 9: Average Precision

3. I submitted with file 111634527.mat. However, since I submitted pretty late (2 hours before the deadline), I don't see my AP on the leaderboard. It is not updated yet. Sorry for the inconvenience.