Homework 4 Answer

Zhao Wang u0905676

April 6, 2019

1 Paper Problems

1. (a) $\xi_i = 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b)$

   (b) $\xi_i = 0$

   (c) If we throw out the term $C \cdot \sum_i \xi_i$, the model may overfit the data.

2. The dual optimization for soft SVMs: minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$, $\sum_i \xi_i$ and b.
   The constraints include: $\forall i, y_i(\mathbf{w}^\top\mathbf{x}_i + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$

3. (a) If an example k stays outside the margin, parameter value $a*_k = 0$.

   (b) For the common, each zero gives an example outside the margin and other give examples inside the margin. For the difference, all $\{\xi_i\}$ should be below the $max(0, 1 - y_i(\mathbf{w}^\top\mathbf{x}_i + b))$ and all α_i should be smaller than 0 and larger than the hyperparameter C.

   (c) We should find the all α_i that has the value of $max(\alpha)$.

4. To use the kernel trick to enable SVM's to perform nonlinear classification, we can try to replace $\mathbf{x}_i^T\mathbf{x}_j$ to kernel $K(\mathbf{x}_i, \mathbf{x}_j)$.
   Then we have the new optimization problem

$$\min_{\{0\leq\alpha_i\leq C\},\sum_i \alpha_i y_i = 0} \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j K(\mathbf{x}_i, \mathbf{x}_j) y_i y_j - \sum_i \alpha_i$$

5. Let function $f_1(\mathbf{w}) = 1 - y_i(\mathbf{w}^\top\mathbf{x}_i + b)$, then we have

$$\nabla^2 f_1(\mathbf{w}) = \begin{Bmatrix} 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 \end{Bmatrix}$$

   which means that $f_1(\mathbf{w})$ is convex function. Therefore, we have function $f_2(\mathbf{w})$

$$f_2(\mathbf{w}) = max(0, f_1(\mathbf{w}))$$

which is the maximum of two convex function, is also convex. Therefore,

$$C\sum_i \max\left(0, 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\right)$$

is a convex function.

Now we suppose that $f_3(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top \mathbf{w}$, then we have

$$\nabla^2 f_3(\mathbf{w}) = \begin{Bmatrix} w_{00} & \dots & w_{0n} \\ \vdots & & \vdots \\ w_{n0} & \dots & w_{nn} \end{Bmatrix}$$

is a n $\times$ n identity matrix which is convex. Therefore, the objective function is convex to $\mathbf{w}$.

Now we try the functions with b

$$f_1''(b) \to 0$$
$$f_3''(b) \to 0$$

In conclusion, the objective function is convex to $\mathbf{w}$ and b.

6. Occam's razor is a principle the suppose there exist two explanations for an occurrence. In this case the one that requires the least speculation is usually better. Another way of saying it is that the more assumptions you have to make, the more unlikely an explanation. Just like the problem 5, we only need a few simple functions to solve a very complicated problem in optimization.

7. First step:

$$\nabla J^1 = [-0.172, 0.344, -0.103]$$

$$\nabla J^2 = [-0.344, -0.687, -0.687]$$

$$\nabla J^3 = [-0.515, -0.069, 0.859]$$

Second step:

$$\nabla J^1 = [-0.172, 0.344, -0.103]$$

$$\nabla J^2 = [-0.344, -0.687, -0.687]$$

$$\nabla J^3 = [-0.515, -0.0687, 0.859]$$

Third step:

$$\nabla J^1 = [-0.172, 0.344, -0.103]$$

$$\nabla J^2 = [-0.344, -0.687, -0.687]$$

$$\nabla J^3 = [-0.515, -0.0687, 0.859]$$

8. (a) $\mathbf{w}$ should be

$$\mathbf{w} = \sum_i c_i \mathbf{x}_i y_i$$

Then we have the predictions:

$$sgn(\mathbf{x}_t \sum_i c_i \mathbf{x}_i y_i)$$

(b) let $\mathbf{c}_0 = 0 \in \mathbb{R}^n$
foreach $(\mathbf{x}_i, y_i)$
  let prediction $y' = sgn(\mathbf{x}_i \sum_j \mathbf{c}_j \mathbf{x}_j y_j)$
  if $y' \neq y_i$
    $\mathbf{c}_i = \mathbf{c}_i + 1$
Return $\mathbf{c}$

(c) Just like what we do in problem 4, replace $\mathbf{x}_i^T \mathbf{x}_j$ to kernel $K(\mathbf{x}_i, \mathbf{x}_j)$.

let $\mathbf{c}_0 = 0 \in \mathbb{R}^n$
foreach $(\mathbf{x}_i, y_i)$
  let prediction $y' = sgn(\mathbf{x}_i \sum_j \mathbf{c}_j K(\mathbf{x}_i, \mathbf{x}_j) y_j)$
  if $y' \neq y_i$
    $\mathbf{c}_i = \mathbf{c}_i + 1$
Return $\mathbf{c}$

2 Practice

1. See github repo

2. (a) With $\gamma_0 = 0.01$, $d = 0.01$, we have training and testing error:

C	1/873	10/873	50/873	100/873	100/291	500/873	700/873
Training Error	0.05	0.041	0.04	0.04	0.04	0.04	0.04
Testing Error	0.072	0.048	0.046	0.046	0.054	0.048	0.048

(b)

C	1/873	10/873	50/873	100/873	100/291	500/873	700/873
Training Error	0.049	0.04	0.039	0.039	0.042	0.047	0.045
Testing Error	0.072	0.05	0.046	0.048	0.05	0.052	0.052

(c) Schedule B has much larger weight vectors than Schedule A as C increasing. However, there's no big different between training and testing error.

C	Schedule A	Schedule B	Training Diff	Testing Diff
1/873	[-0.373, -0.17, -0.145, -0.079]	[-0.375, -0.171, -0.143, -0.086]	0.001	0
10/873	[-0.729, -0.366, -0.371, -0.195]	[-0.8, -0.423, -0.423, -0.227]	0.001	0.002
50/873	[-1.105, -0.594, -0.644, -0.247]	[-1.471, -0.832, -0.892, -0.428]	0.001	0
100/873	[-1.27, -0.682, -0.745, -0.281]	[-2.497, -1.595, -1.697, -0.759]	0.001	0.002
100/291	[-1.618, -0.82, -0.934, -0.311]	[-4.853, -3.411, -3.539, -1.912]	0.002	0.004
500/873	[-1.752, -0.907, -1.011, -0.334]	[-8.175, -5.267, -5.774, -2.799]	0.007	0.004
700/873	[-1.806, -0.929, -1.053, -0.346]	[-15.785, -9.629, -10.899, -4.483]	0.005	0.004

3. (a) **w** is smaller than both 2(b) and 2(c). This is because we only use C = 100/873, 500/873 and 700/873 so it needs larger offset to be more accurate.

C	b	w
100/873	-5.344	[-0.9611, -0.6343, -0.7513, -0.0411]
500/873	-8.758	[-1.634, -1.007, -1.1784, -0.161]
700/873	-11.064	[-2.043, -1.353, -1.5131, -0.235]

(b) The best combination is $C = 700/873$ and $\gamma = 100$. Compared with linear SVM, the dual SVM has much better performance.

Training								
C	0.01	0.1	0.5	1	2	5	10	100
100/873	0	0	0	0	0	0.01	0.01	0.05
500/873	0	0	0	0	0	0.01	0.01	0.01
700/873	0	0	0	0	0	0.01	0.01	0.01
Testing								
C	0.01	0.1	0.5	1	2	5	10	100
100/873	0.056	0.024	0.024	0.024	0.024	0.03	0.028	0.088
500/873	0.056	0.024	0.024	0.024	0.024	0.03	0.026	0.026
700/873	0.056	0.024	0.024	0.024	0.024	0.028	0.028	0.018

(c) From the result we get, the total of support vectors decreases when gamma increasing. That means the margin must be shrinked for more examples.

gamma	0.01 & 0.1	0.1 & 0.5	0.5 & 1	1 & 2	2 & 5	5 & 10	10 & 100
count	1.0	1.0	1.0	0.94	0.85	0.8	0.51