ml-skill-proj2

1. Consider the toy data set $\{([0,0],-1),([2,2],-1),([2,0],+1)\}$. Set up the dual problem for the toy data set. Then, solve the dual problem and compute α^*, the optimal Lagrange multipliers. (Note that there will be three weights $\boldsymbol{w} = [w_0, w_1, w_2]$ by considering the bias.)

$$\max_{\alpha} \min_{w} \mathcal{L}(w, \alpha), s.t. \alpha_i >= 0$$

$$\max_{\alpha} \mathcal{L}(w, \alpha) = \frac{-1}{2} \sum_{i=1}^{3} \sum_{j=1}^{3} \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{3} \alpha_i, s.t. \alpha_i > 0$$

$$1/2 w^T w + \sum_{i=1}^{n} \alpha_i (1 - y_i (w^T x_i))$$

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

then i=3

$$= 1/2(w_0^2 + w_1^2 + w_2^2) + \alpha_1(1 - w_0) + \alpha_2(1 + w_0 + 2w_1 + 2w_2) + \alpha_3(1 - w_0 - 2w_1)$$

then

(1)derivative for w_0

$$w_0 + \alpha_1 + \alpha_2 - \alpha_3 = 0$$

(2)derivative for w_1

$$w_1 + 2\alpha_2 - 2\alpha_3 = 0$$

(3)derivative for w_2

$$w_2 + 2\alpha_2 = 0$$

(4)derivative for α_1

$$w_0 - 1 = 0$$

(5)derivative for α_2

$$1 + w_0 + 2w_1 + 2w_2 = 0$$

(6)derivative for α_3

$$1 - w_0 - 2w_1 = 0$$

Hence:

$$w_0 = -1$$

$$w_1 = 1$$

$$w_2 = -1$$

$$\alpha_1 = 3/2$$

$$\alpha_2 = 1/2$$

$$\alpha_3 = 1$$

a*=$[\frac{3}{2}\ \frac{1}{2}\ 1]^T$

2. In a separable case, when a multiplier $\alpha_i > 0$, its corresponding data point $(\boldsymbol{x}_i, y_i)$ is on the boundary of the optimal separating hyperplane with $y_i(\boldsymbol{w}^T\boldsymbol{x}_i) = 1$. Show that the inverse is not True. Namely, it is possible that $\alpha_i = 0$ and $(\boldsymbol{x}_i, y_i)$ is on the boundary satisfying $y_i(\boldsymbol{w}^T\boldsymbol{x}_i) = 1$.

   [Hint: Consider a toy data set with two positive examples at ([0,0], +1) and ([1, 0], +1), and one negative example at ([0, 1], -1).] (Note that there will be three weights $\boldsymbol{w} = [w_0, w_1, w_2]$ by considering the bias.)

$$\mathcal{L}(w, \alpha) = 1/2(w_0^2 + w_1^2 + w_2^2) + \alpha_1(1 - w_0) + \alpha_2(1 - w_0 - w_1) + \alpha_3(1 + w_0 + w_2)$$

(1)derivative for w_0

$$w_0 - \alpha_1 - \alpha_2 + \alpha_3 = 0$$

(2)derivative for w_1

$$w_1 - \alpha_2 = 0$$

(3)derivative for w_2

$$w_2 + \alpha_3 = 0$$

(4)derivative for α_1

$$w_0 - 1 = 0$$

(5)derivative for α_2

$$1 - w_0 - 2w_1 = 0$$

(6)derivative for α_3

$$1 + w_0 + w_2 = 0$$

Hence:

$$w_0 = 1$$

$$w_1 = 0$$

$$w_2 = -2$$

$$\alpha_1 = 3$$

$$\alpha_2 = 0$$

$$\alpha_3 = 2$$

Although α_2 equal to 0, but the data $([1,0],+1)$ still lies on the hyperplane.

3. **Non-separable Case SVM:** In Lecture 8 (page 18), we compare the hard-margin SVM and soft-margin SVM. Prove that the dual problem of soft-margin SVM is almost identical to the hard-margin SVM, except that α_is are now bounded by C (tradeoff parameter).

   Lagrangian function for hard-margin svm

   L = $1/2|\ w\ |^2 + \sum_{i=1}^{n} \alpha_i(1 - y_i(w^T x_i))$

Lagrangian function for soft-margin svm

L = $1/2|\ w\ |^2 + \sum_{i=1}^{n} \alpha_i(1 - y_i(w^T x_i)) + C\sum_{i=1}^{n} \epsilon_i - \sum_{i=1}^{n} \mu_i \epsilon_i$

$y_i(w^T x_i) \geq$ 1- ϵ_i $\epsilon_i \geq 0$, i =1,.....,N

w = $\sum_{i=1}^{n} \alpha_i y_i x_i$

substitute w for soft-L

Hence,

$\max_\alpha \mathcal{L}(w,\alpha) = \frac{-1}{2}\sum_{i=1}^{3}\sum_{j=1}^{3} \alpha_i\alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{3} \alpha_i, s.t. 0 \leq \alpha_i \leq$C

4. **Kernel Function:** A function K computes $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = -\boldsymbol{x_i}^T \boldsymbol{x}_j$. Is this function a valid kernel function for SVM? Prove or disprove it.

   Kernel Matrix is symmetric positive semi-definite and the positive eigenvalues. Mercer's condition: $\sum_{i=1}^{n}\sum_{j=1}^{n} k(x_i, x_j) > 0$, It is very obvious for observing that there is a negative sign in front of the equation. It represents that summing up all the data after this kernel function will lead to the negative outcome. Hence, this is not a valid kernel function.

5. **Support Vector Machine for Handwritten Digits Recognition**: You need to use the software package LIBSVM http://www.csie.ntu.edu.tw/ cjlin/libsvm/ to finish this assignment. Two functions **svm_train()** and **svm_predict()** from LIBSVM library will be used in this question. The package has already been included in the code folder. You only need to run "make" command at the package location to use them. Please read the "LIBSVM tutorial" section in "Readme.txt" file carefully to understand how to use these functions. The handwritten digits

files are in the "data" folder: train.txt and test.txt. The starting code is in the "code" folder. In the data file, each row is a data example. The first entry is the digit label ("1" or "5"), and the next 256 are grayscale values between -1 and 1. The 256 pixels correspond to a 16×16 image. You are expected to implement your solution based on the given codes. The only file you need to modify is the "solution.py" file. You can test your solution by running "main.py" file. Note that code is provided to compute a two-dimensional feature (symmetry and average intensity) from each digit image; that is, each digit image is represented by a two-dimensional vector. These features along with the corresponding labels should serve as inputs to your solution functions.

(a) Complete the **svm_with_diff_c()** function. In this function, you are asked to try different values of cost parameter c.

(b) Complete the **svm_with_diff_kernel()** function. In this function, you are asked to try different kernels (linear, polynomial and radial basis function kernels).

(c) Summarize your observations from (a) and (b) into a short report. In your report, please report the accuracy result and total support vector number of each model. A briefly analysis based on the results is also needed. For example, how the number of support vectors changes as parameter value changes and why.

A. When c increase, score increase in the beginning. When c = 5, score decrease. Total support vector number decrease when c increase. So over-fitting happen in the beginning and then under-fitting when c = 5.

c=[0.01,0.1,1,2,3,5], score = [0.95754717 0.96226415 0.96226415 0.96226415 0.96226415 0.95990566], support vector number = [[248 248] [84 82] [46 44] [42 42] [41 39] [37 38]]

B. When k is rbf, score is maximum. The total support vector number: poly <rbf<linear<sigmoid.
So over-fitting happen when k is sigmoid, and under-fitting happen when k is poly.

k = ['linear','poly', 'rbf', 'sigmoid'], score = [0.95990566 0.95754717 0.96226415 0.9009434], support vector number = [[81 81] [37 38] [46 44] [302 302]]