ANDREW DUDLEY
STATISTICAL MACHINE LEARNING
HOMEWORK 2

1 SUPPORT VECTOR MACHINES

SVM is an algorithm which fits the max-margin hyperplane in a certain feature space.

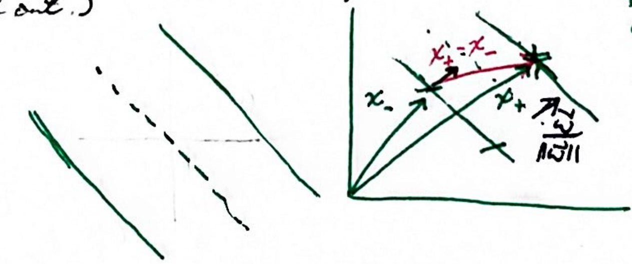
1.1 HARD MARGIN LINEAR SVM. GIVEN A LINEARLY SEPERABLE DATA SET WHERE EACH EXAMPLE IS EITHER FROM CLASS $y_i=+1$ OR $y_i=-1$, THEN A LINEAR SVM CAN ALWAYS FIND THE MAX-MARGIN CLASSIFIER THAT CORRECTLY CLASSIFIES ALL THE TRAINING DATA AS FOLLOWS.

$$\underset{\omega,b}{\arg\max}\ d = d^+ - d^- = \frac{w^T x_j}{\|w\|_2} - \frac{w^T x_k}{\|w\|_2}$$

SUPPOSE THAT WE KNOW THAT THERE ARE ONLY 2 TRAINING EXAMPLES ON THE MARGINS, i.e., THE SUPPORT VECTORS, $x_j, y_j = 1$ AND $x_k, y_k = -1$, AND THE PARAMETERS OF THE LINEAR SVM ARE w^* AND b^*. WRITE DOWN THE CONSTRAINTS THIS LINEAR SVM HAS TO SATISFY WITH x_j, x_k, w^*, b^*, AND FORM THE PROBLEM OF LINEAR SVM AS A CONTRAINED OPTIMIZATION PROBLEM. (HINT: The optimization problem should use both d from eqn. 1, as well as the constraints we just worked out.)

$\vec{\omega}\cdot\vec{\omega} \geq c$

$\omega\cdot\vec{u}+b \geq 0$ THEN +



DOT PRODUCT WITH UNIT VECTOR GIVES LENGTH IN THAT DIRECTION AND $\vec{w}$ IS NORMAL TO BOUNDARY, SO $(x_+ - x_-)\cdot\frac{\vec{w}}{\|\vec{w}\|} = \frac{\vec{x}_+\cdot\vec{w}}{\|\vec{w}\|} - \frac{\vec{x}_-\cdot\vec{w}}{\|\vec{w}\|}$ GIVES THE WIDTH OF THE MARGIN.

① ADD CONSTRAINTS

$\omega\cdot x_+ + b \geq 1$, $\omega\cdot x_- + b \leq -1$. $y_i = \begin{cases} + & \text{for } +\text{samples} \\ - & \text{for } -\text{samples}\end{cases}$

$$\Rightarrow y_i(\vec{\omega}\cdot\vec{x}_i + b) \geq 1 \Rightarrow y_i(\omega\cdot x_i + b) - 1 \geq 0$$

SO FOR $\vec{x}_i$ IN THE GUTTER, SET CONSTRAINT

$$\boxed{y_i(\vec{\omega}\cdot\vec{x}_i + b) - 1 = 0}$$

$$\boxed{\begin{aligned} &\Rightarrow \vec{\omega}\cdot\vec{x}_j + b - 1 = 0 \\ &\text{AND } \vec{\omega}\cdot\vec{x}_k + b + 1 = 0 \end{aligned}} \quad (i)$$

② FORMULATE EQN TO BE SOLVED.

WE KNOW WE WANT TO MAXIMIZE THE WIDTH OF THE MARGIN $\frac{w^T x_j}{\|w\|} - \frac{w^T x_k}{\|w\|}$, and using (i) we CAN SOLVE FOR $\underset{1-b}{\vec{\omega}\cdot x_j}$ AND $\underset{-(1+b)}{\vec{\omega}\cdot x_k}$, THEN PLUG THEM BACK INTO THE MARGIN EQN. AND SIMPLIFY TO GET

WIDTH $= \frac{2}{\|w\|}$, which we want to maximize.

$$\Rightarrow \boxed{\text{MINIMIZE } \|w\|} \equiv \text{minimize } \frac{1}{2}\|w\|^2$$

③ FORM LINEAR SVM PROBLEM AS A CONSTRAINT OPTIMIZATION PROBLEM.

WE CAN COMBINE THE EQN TO BE MINIMIZED WITH OUR CONSTRAINTS USING LAGRANGIAN MULTIPLIERS

$$\boxed{L = \frac{1}{2}\|\vec{w}\|^2 - \alpha_j\left[\vec{w}\cdot x_j + b - 1\right] - \alpha_k\left[\vec{w}\cdot x_k + b + 1\right]}$$

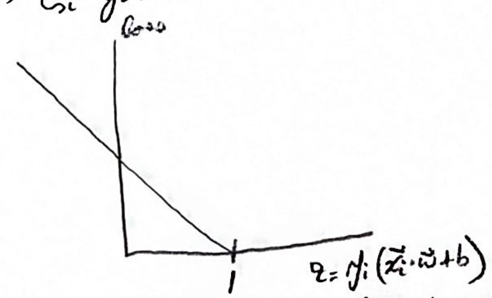
1.2 However, in real world cases it is difficult to confirm if a dataset is linearly separable or not, even in kernel space. Therefore, soft margin SVM is introduced to handle those samples which can't correctly be classified.

(a). PLEASE DESCRIBE HOW INCORRECTLY CLASSIFIED SAMPLES ARE HANDLED BY A SOFT-MARGIN SVM.

Instead of just minimizing the width of the margin, we now want to minimize the width of the margin + the number of mistakes. But not all mistakes are equally bad, so we'll use their position to the margin to weight the penalty.

$$\min \frac{1}{2}\|\vec{w}\|^2 + \underset{\text{PARAM}}{C}\sum_{i=1}^{n}\xi_i$$

The loss ξ_i for each x_i is calculated with



(b) Three types of examples will exist after training SVM based on the value of the slack variables. Describe them and state whether removing them would result in a change of decision boundary

ξ_i corresponding to $\alpha_i = C$, x_i is A SUPPORT VECTOR ON THE MARGIN BOUNDARY

ξ_i " " α_i s.t. $0 < \alpha_i < C$, x_i is A SUPPORT VECTOR ON THE WRONG SIDE OF THE MARGIN BOUNDARY

ξ_i " " $\alpha_i = 0$, x_i IS ON THE CORRECT SIDE OF THE DECISION BOUNDARY, AND IS NOT ON THE BOUNDARY.

THESE ARE SUPPORT VECTORS, AND BY DEF. WILL CHANGE THE DECISION BOUNDARY IF REMOVED

2 AdaBoost

2.1 AdaBoost is an ensemble of weak classifiers whose accuracy is slightly over 50%. What will happen to the AdaBoost algorithm if your weak classifier has exactly 50% accuracy?

It will stop learning (or not learn anything, if this is the initial classifier h_0). This is because the voting power α is calculated by $\frac{1}{2}\ln\left(\frac{1-\varepsilon}{\varepsilon}\right)$, which equals 0 when $\varepsilon = 0.5$. No future classifiers will receive any voting power.

2.2 What will happen if you use a weak binary classifier whose classification accuracy is <u>less</u> than 50%, say 45%?

It will learn to classify the opposite class, and you simply have to flip the sign.

-0.5 c c^1 2.5 5.5 8.5 9.5

0	1	2	3	4	5	6	7	8	9
1	1	1	-1	-1	-1	1	1	1	-1

Table 1: Training set

For the weak classifier c, we will use a single threshold θ, so that

$$c(x) = \begin{cases} +1 & x < \theta \\ -1 & x \geq \theta \end{cases}$$

2.3 For the first iteration, we let the threshold $\theta = 2.5$ which minimizes the classification error at the current iteration. Show us how you derive the weights D_2 after this iteration.

	D_1	D_2
w_0	1/10	1/14
w_1	1/10	1/14
w_2	1/10	1/14
w_3	1/10	1/14
w_4	1/10	1/14
w_5	1/10	1/14
w_6	1/10	1/6
w_7	1/10	1/6
w_8	1/10	1/6
w_9	1/10	1/14

c^1 misclassifies x_6, x_7, x_8

$\varepsilon^1 = 3/10$

So $w_6^{t+1} + w_7^{t+1} + w_8^{t+1} = 1/2$

w_6^t, w_7^t, w_8^t have ratio 1:1:1, so update weights to $\frac{1}{6}$ for each

Same for the 7 remaining weights (classified correctly, w_i^t equal ratio, w_i^{t+1} should sum to $\frac{1}{2}$), so update w_i^{t+1} $\forall i \in$ correct to $\frac{1}{14}$

2.4 Show us how you derive D_3 & D_4, respectively.

SEE NEXT PAGE...

2.4 continued: SHOW US HOW YOU DERIVE D_3 + D_4, RESPECTIVELY

[illegible] h^2: [illegible]

	D_1	D_2	D_3	D_4
w_0	1/10	1/14	3/66	1/8
w_1	1/10	1/14	3/66	1/8
w_2	1/10	1/14	3/66	1/8
w_3	1/10	1/14	11/66	11/108
w_4	1/10	1/14	11/66	11/108
w_5	1/10	1/14	11/66	11/108
w_6	1/10	1/6	7/66	7/108
w_7	1/10	1/6	7/66	7/108
w_8	1/10	1/6	7/66	7/108
w_9	1/10	1/14	3/66	1/8

QUESTION FOR MINUS:	MISCLASSIFIES	ϵ_{D3}	ϵ_{D4}	ϵ_{5}
$x \geq -0.5$	0,1,2,6,7,8	10/14	23/66	
$x \geq 2.5$	6,7,8	7/14	21/66	
$x \geq 5.5$	3,4,5,6,7,8	10/14	54/66	
$x \geq 8.5$	3,4,5	3/14	1/2	
$x < -0.5$	3,4,5,9	4/14	36/66	
$x < 2.5$	0,1,2,3,4,5,9	7/14	45/66	
$x < 5.5$	0,1,2,9	4/14	12/66	
$x < 8.5$	0,1,2,6,7,8,9	11/14	1/2	

For wrong:

$w_3^t = w_4^t = w_5^t = \frac{1}{14}$

$w_3^{t+1} = w_4^{t+1} = w_5^{t+1} = \frac{1}{6}$

For right:

$w^{t+1} = \frac{w^t}{2} \cdot \frac{1}{1-\epsilon^t} = \frac{1}{14 \cdot 2} \cdot \frac{1}{1-3/14} = \frac{1}{28} \cdot \frac{1}{11/14} = \frac{14}{11 \cdot 14 \cdot 2} = \frac{1}{22}$

w_0, w_1, w_2, w_9

$= \frac{1}{6 \cdot 2} \cdot \frac{14}{11} = \frac{7}{66} = w_6, w_7, w_8$

D4:

h^3: $x < 5.5$

For wrong:

$w_0 = w_1 = w_2 = w_9$

$w_0 + w_1 + w_2 + w_9 = 1/2$

$w_0 = w_1 = w_2 = w_9 = 1/8$

For right:

$w^{t+1} = \frac{w^t}{2} \cdot \frac{1}{1-\epsilon^t}$

when $w^t = 11/66$:

$\frac{11}{66 \cdot 2} \cdot \frac{66}{54} = \frac{11}{108}$

when $w^t = 7/66$

$\frac{7}{66 \cdot 2} \cdot \frac{66}{54} = \frac{7}{108}$

2.5 SHOW US WHAT HAPPENS ON THE FOURTH ITERATION

Based on the phrasing of the hint, all ϵ should be 1/2, but that's not what I'm getting.

3. KNN CLASSIFIER

3.1 A LAZY CLASSIFIER

Consider an online learning setting where besides the observed training data, we have new training data coming in as time goes by, SOME CLASSIFIERS HAVE TO BE RETRAINED FROM SCRATCH.

NOTE: THE "HINT" FOR THIS PROBLEM ISN'T A HINT AT ALL. IT'S A RE-DEFINING OF WHAT IT MEANS TO "LEARN FROM SCRATCH", WHICH IS SILLY.

(a) BETWEEN SVM, KNN, + NB, WHICH HAVE TO BE RETRAINED "FROM SCRATCH" WHEN NEW DATA IS RECEIVED?

SVM has to be retrained when the new data falls on or on the wrong side of the margin. (IF THE SVM WAS MODELLING DATA GUARANTEED TO BE LINEARLY SEPERABLE, IT WOULD BE OK IF THE NEW DATA WAS ON THE MARGIN [i.e. not using slack vars].

USING THIS DEFINITION, NB HAS TO BE RETRAINED "FROM SCRATCH".

KNN DO NOT HAVE A TRAINING PHASE, SO OF COURSE THEY DON'T NEED TO BE RETRAINED. (OF COURSE, YOU COULD ALSO CONSIDER ADDING NEW POINTS TO THE TRAIN SET AS "TRAINING", IN WHICH CASE, IT WOULD INCLUDE KNN $O(1)$)

(b)

FOR SVM, TIME COMPLEXITY FOR TESTING WILL DEPEND ON IF IT IS A LINEAR SVM OR A KERNEL SVM.

$O(\#\text{ OF SUPPORT VECTORS})$

FOR NB,

$O(1)$

FOR KNN,

testing is linear in terms of the # OF TRAINING DATA.

$O(n \cdot [\#\text{ of dimensions}])$

(NOTE THAT THIS IS USING MEDIAN OF MEDIANS TO FIND KTH SMALLEST DISTANCE).

3.2

(a)

ALGORITHM KNN

```
procedure predict
  for test_sample in test_set:
    diff = element-wise difference between test_sample + each train_sample in train_set
    distance = L2-norm of diff matrix.
    min_idx = indices of first k smallest values in distance vector
    neighbors = train_targets[min_idx]
    classification = most frequently occuring value in neighbors vector.
  return all classifications
end procedure
```

(b) SEE ATTACHED IMAGE + CODE!