MÁY VÉC-TƠ TỰA SONG SINH VÀ ÁP DỤNG

Nguyễn Thế Cường

Khoa Toán, Trường Đại Học Thông tin Liên lạc, Nha Trang, Khánh Hòa

Email: nckcbnckcb@gmail.com

Ngày nhận bài: 25/5/2020; ngày hoàn thành phản biện: 30/10/2020; ngày duyệt đăng: 02/11/2020

TÓM TẮT

Máy véc-tơ tựa (SVM) là một kĩ thuật phân lớp rất phổ biến và được vận dụng vào rất nhiều các lĩnh vực khác nhau của đời sống. Nhận thấy đây là một vấn đề thiết thực nên chúng tôi đã lựa chọn để nghiên cứu, nhằm mục tiêu tìm thêm các ứng dụng thực tiễn của thuật toán và các cải tiến tốt hơn. Một vài các cải tiến tiêu biểu như: máy véc-tơ tựa xấp xỉ (PSVM), máy véc-tơ tựa xấp xỉ thông qua trị riêng suy rộng (GEPSVM), máy véc-tơ tựa song sinh (TWSVM). SVM và TWSVM đều được giải dựa vào bài toán đối ngẫu Lagrange nhưng TWSVM dùng hai siêu phẳng để tách hai lớp dữ liệu. Bài báo này nhằm trình bày cơ sở toán học của PSVM, GEPSVM và TWSVM, bên cạnh đó chúng tôi đã cài đặt thuật toán TWSVM bằng ngôn ngữ lập trình Python để đánh giá hiệu quả của TWSVM so với SVM tiêu chuẩn.

Từ khóa: Máy véc-tơ tựa, Máy véc-tơ tựa song sinh.

1. GIỚI THIỆU

Trong bài báo này chúng tôi đề cập đến nguồn gốc toán học của các cải tiến của Support Vector Machine (SVM) [1, 2, 3, 9] là Proximal Support Vector Machine (PSVM) [6], Multisurface proximal support vector classification via generalized eigenvalues (GEPSVM) [4] và Twin Support Vector Machine (TWSVM) [5], trong đó đặc biệt chú trọng đến TWSVM. Như đã biết, SVM tiêu chuẩn [3] chú trọng đến việc tối đa lề giữa hai lớp dữ liệu và tối thiểu lỗi phân loại, bằng cách giải bài toán tối ưu dạng:

$$\begin{cases} f(x) \to \min \\ x \in M. \end{cases}$$

Nghiệm tìm được là một siêu phẳng tách hai lớp dữ liệu với lề lớn nhất.

TWSVM [5] tìm hai siêu phẳng không nhất thiết song song, cụ thể ta giải hai bài toán quy hoạch toàn phương (Quadratic Programming (QP)), còn trong SVM ta chỉ giải một bài toán QP. Điều thú vị ở chỗ, tuy giải hai bài toán QP nhưng tốc độ của TWSVM

lại nhanh hơn SVM tiêu chuẩn. Để hiểu tường minh hơn về hiệu quả của TWSVM so với SVM tiêu chuẩn độc giả nên tham khảo về cài đặt thuật toán đã được chúng tôi đưa lên https://github.com/makeho8/python/blob/master.

2. ĐÁNH GIÁ CÁC NGHIÊN CỨU LIÊN QUAN

Có thể kể đến các nghiên cứu đã được công bố trước TWSVM [5] như: máy véc-tơ tựa xấp xỉ (PSVM) [6], máy véc-tơ tựa xấp xỉ thông qua trị riêng suy rộng (GEPSVM) [4].

PSVM [6] tiếp cận tương tự như SVM tiêu thuẩn [3] bằng cách tìm siêu phẳng tách với lề lớn nhất, nhưng PSVM chú trọng tới việc tìm hai siêu phẳng song song tương ứng với hai lớp dữ liệu sao cho các điểm trên mỗi lớp dữ liệu tụ tập quanh nó, và hai siêu phẳng này tạo ra lề lớn nhất có thể. Ưu điểm của PSVM là giải bài toán QP lồi chặt với ràng buộc đẳng thức, điều này tăng tính ổn định của thuật toán. Tuy nhiên việc tìm hai siêu phẳng song song dẫn tới khả năng mô phỏng dữ liệu của PSVM kém hơn TWSVM.

Tiếp cận bài toán phân loại hai lớp dữ liệu dưới một góc nhìn khác, Mangasarian và Wild đã tìm hai siêu phẳng không nhất thiết song song mà mỗi lớp dữ liệu là gần với một trong hai siêu phẳng và xa lớp còn lại nhất có thể (GEPSVM [4]). Hai siêu phẳng này ứng với các vector riêng của các trị riêng nhỏ nhất trong hai bài toán tìm trị riêng suy rộng. Vì GEPSVM giải hai bài toán tối ưu không có ràng buộc nên độ chính xác và khả năng mô phỏng dữ liệu kém hơn TWSVM.

TWSVM [5] cũng có cách tiếp cận tương tự GEPSVM [4], với mục đích là tìm hai siêu phẳng không nhất thiết song song. Tuy nhiên công thức toán học của TWSVM là hoàn toàn khác với GEPSVM. TWSVM hiệu quả hơn về thời gian tính toán, hơn nữa bên cạnh việc phân loại dữ liệu, TWSVM còn có khả năng mô phỏng dữ liệu tốt hơn PSVM và GEPSVM, trong khi SVM chỉ thực hiện nhiệm vụ phân loại.

3. MÁY VÉC-TƠ TỰA XẤP XỈ

Giả sử tập mẫu cần phân loại gồm m điểm được kí hiệu bởi m vector hàng $A_i,\ i = 1,\dots,m$ trong không gian $\mathbb{R}^n$. Ở đây $A_i = (A_{i1},\dots,A_{in})^T$, đặt $y_i \in \{1, -1\}$ là lớp của điểm dữ liệu thứ i.

Trường hợp hai lớp tách được tuyến tính, SVM tiêu chuẩn tìm véc-tơ cột $w \in \mathbb{R}^n$ và $b \in \mathbb{R}$ sao cho

$$\begin{cases} A_i w + b \geq 1; & y_i = 1, \\ A_i w + b \leq -1; & y_i = -1. \end{cases}$$

Siêu phẳng tách $x^T w + b = 0$ nằm giữa hai siêu phẳng tựa $x^T w + b = 1$ và $x^T w + b = -1$. Hai lớp dữ liệu được tách với lề mỗi phía là $\frac{1}{\|w\|_2}$. Những điểm dữ liệu nằm trên các siêu phẳng tựa được gọi là các véc-tơ tựa. Vì muốn có lề cực đại ta đưa đến bài toán tối ưu sau:

$$\begin{cases} \min\limits_{w,b} \dfrac{1}{2} w^T w \\ \quad A_i w + b \geq 1; \quad y_i = 1, \\ \quad A_i w + b \leq -1; \quad y_i = -1. \end{cases}$$

Trường hợp hai lớp gần tách được tuyến tính. Lúc đó có những điểm vi phạm bất đẳng thức ràng buộc. Ta cần giảm nhẹ ràng buộc bằng cách thêm vào các biến phụ $q \in \mathbb{R}^m$

$$\begin{cases} A_i w + b + q_i \geq 1; & y_i = 1, \\ A_i w + b - q_i \leq -1; & y_i = -1, \\ q_i \geq 0; & i = 1, \ldots, m. \end{cases}$$

Ta coi q_i như là biến lỗi tương ứng với điểm dữ liệu thứ i. Nếu bài toán là tách được tuyến tính thì $q_i = 0$ với mọi i, tức là không có điểm nào vi phạm ràng buộc. Gọi $D \in \mathbb{R}^{m \times m}$ là ma trận đường chéo nhận giá trị 1 hoặc -1 tương ứng với lớp của điểm dữ liệu A_i, $i = 1, \ldots, m$. Cuối cùng ta có bài toán SVM lề mềm như sau:

$$\begin{cases} \min\limits_{w,b,q} \; ce^T q + \dfrac{1}{2} w^T w \\ \quad D(Aw + eb) + q \geq e; \\ \quad q \geq 0 \end{cases} \qquad (1)$$

ở đây c là hệ số phạt các điểm vi phạm ứng với $q_i > 0$, hơn nữa chọn c hợp lý còn để cân bằng giữa việc tối thiểu lỗi và tối đa lề, $e \in \mathbb{R}^m$ là véc-tơ với tất cả các thành phần bằng 1.

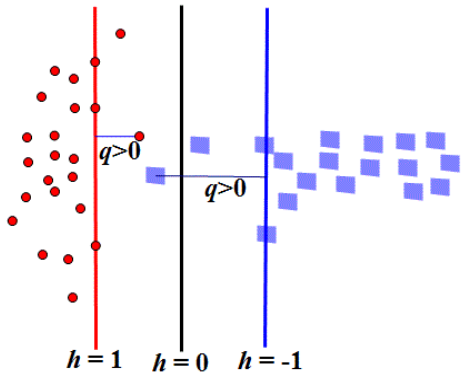
Ý tưởng của PSVM [6] xuất phát từ việc đưa bài toán (1) về dạng:

$$\begin{cases} \min\limits_{w,b,q} \; c\dfrac{1}{2}\|q\|^2 + \dfrac{1}{2}(w^T w + b) \\ \quad D(Aw + eb) + q \geq e; \end{cases} \qquad (2)$$

Như vậy, trong hàm mục tiêu của (2), $c\frac{1}{2}\|q\|^2$ thay cho $ce^T q$ và việc tối đa lề $\frac{2}{\|w\|_2}$ được thay bởi $\frac{2}{\left\|\frac{w}{b}\right\|_2}$. Chú ý rằng ràng buộc $q \geq 0$ được lược bỏ. Điều này là bởi nếu tồn tại $q_i < 0$ thì ta có thể thay bởi $q_i = 0$ mà không vi phạm ràng buộc và có giá trị hàm mục tiêu bé hơn. Hơn nữa, PSVM [6] thay thế ràng buộc bất đẳng thức trong (2) thành ràng buộc đẳng thức:

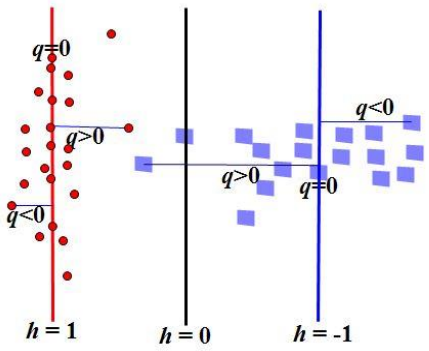
$$\begin{cases} \min\limits_{w,b,q} \; c\dfrac{1}{2}\|q\|^2 + \dfrac{1}{2}(w^T w + b) \\ \quad D(Aw + eb) + q = e; \end{cases} \qquad (3)$$

Trong (2) những điểm vi phạm ràng buộc tương ứng với $q_i > 0$, những điểm nằm trên hai siêu phẳng tựa hoặc không vi phạm tương ứng với $q_i = 0$. Như vậy hai siêu phẳng tựa gần như giới hạn hai lớp dữ liệu về hai phía.



Hình 1: SVM tiêu chuẩn.

Từ ràng buộc đẳng thức trong (3) ta thấy rằng, những điểm vi phạm ràng buộc tương ứng với $q_i > 0$, những điểm nằm trên hai lề tương ứng với $q_i = 0$ còn những điểm không vi phạm tương ứng với $q_i < 0$. Như vậy để tối thiểu hàm mục tiêu thì hai lề sẽ nằm xuyên vào giữa hai lớp dữ liệu tương ứng. Hai lề này được coi như các siêu phẳng xấp xỉ mà các điểm dữ liệu của lớp tương ứng tụ tập xung quanh chúng. Mặt khác, các siêu phẳng xấp xỉ này được đẩy ra xa nhau nhất có thể bởi số hạng $(w^T w + b^2)$ trong hàm mục tiêu. Siêu phẳng tách $x^T w + b = 0$ nằm giữa hai siêu phẳng xấp xi song song.



Hình 2: PSVM

Bài toán (3) là bài toán quy hoạch toàn phương lồi chặt với hàm Lagrange:

$$L(w, b, q, \lambda) = c\frac{1}{2}\|q\|^2 + \frac{1}{2}\left\|\begin{matrix} w \\ b \end{matrix}\right\|^2 - \lambda^T(D(Aw + eb) + q - e), \qquad (4)$$

với $\lambda \in \mathbb{R}^m$ là nhân tử Lagrange. Bằng cách giải hệ điều kiện K.K.T [7] ta thu được nghiệm $\lambda = \left(\frac{I}{c} + HH^T\right)^{-1} e$, với $H = D[A\ \ e]$. Vì ma trận lấy nghịch đảo ở đây có số chiều $m \times m$, thời gian tính toán sẽ rất lâu nếu bài toán có nhiều điểm dữ liệu. Thông thường số chiều dữ liệu n là nhỏ hơn rất nhiều so với số điểm dữ liệu m. Bằng cách sử dụng công thức Sherman-Morrison-Woodbury [8, tr50] đưa về tính ma trận nghịch đảo có số chiều là $(n+1) \times (n+1)$ ta có:

$$\lambda = c\left(I - H\left(\frac{I}{c} + HH^T\right)^{-1} H^T\right)e. \qquad (5)$$

Chú ý rằng, trong SVM các véc-tơ tựa tương ứng với thành phần khác không của nhân tử Lagrange, còn trong PSVM, nhân tử Lagrange $\lambda = cq$, nhưng véc-tơ lỗi q có các thành phần hầu hết khác không. Thực tế chỉ cần nhiều nhất $n+1$ điểm dữ liệu độc lập tuyến tính để xác định $(w, b) \in \mathbb{R}^{n+1}$. Ta gọi những điểm A_i tương ứng với $|q_i| < \varepsilon$ nào đó là các ε-vector tựa. Có thể chọn ε đủ nhỏ để chỉ khoảng 1% số điểm dữ liệu là các ε-vector tựa.

4. MÁY VÉC-TƠ TỰA XẤP XỈ THÔNG QUA CÁC TRỊ RIÊNG SUY RỘNG

Quay lại với bài toán phân loại nhị phân với tập dữ liệu gồm m điểm trong không gian thực $\mathbb{R}^n$. Giả sử lớp 1 gồm m_1 điểm được biểu diễn bởi ma trận $A \in \mathbb{R}^{m_1 \times n}$, lớp 2 gồm m_2 điểm được biểu diễn bởi ma trận $B \in \mathbb{R}^{m_2 \times n}$, với $m_1 + m_2 = m$. PSVM sẽ tìm hai siêu phẳng song song với lề lớn nhất. GEPSVM [4] có cách tiếp cận khác là tìm hai siêu phẳng không nhất thiết song song, mỗi siêu phẳng là gần nhất có thể với một lớp và xa nhất có thể với lớp còn lại.

$$x^T w^{(1)} + b^{(1)} = 0;\ x^T w^{(2)} + b^{(2)} = 0. \qquad (6)$$

Ở đây siêu phẳng 1 của (6) gần nhất với các điểm của lớp 1 và xa nhất với các điểm thuộc lớp 2, điều tương tự xảy ra với siêu phẳng 2. Việc tìm siêu phẳng 1 tương đương với bài toán tối ưu sau:

$$\min_{(w,b)\neq 0} \frac{\|Aw + e_1 b\|^2 / \left\|\begin{bmatrix} w \\ b \end{bmatrix}\right\|^2}{\|Bw + e_2 b\|^2 / \left\|\begin{bmatrix} w \\ b \end{bmatrix}\right\|^2}, \qquad (7)$$

với $e_1 \in \mathbb{R}^{m_1}$, $e_2 \in \mathbb{R}^{m_2}$ là các véc-tơ gồm tất cả tọa độ bằng 1, $\|\cdot\|$ là chuẩn L_2. Ta thấy rằng tử số chính là tổng của bình phương khoảng cách tất cả các điểm thuộc lớp 1 tới siêu phẳng 1, mẫu số là tổng của bình phương khoảng cách tất cả các điểm thuộc lớp 2 tới siêu phẳng 1. Đơn giản hóa (7) cho ta:

$$\min_{(w,b)\neq 0} \frac{\|Aw + e_1 b\|^2}{\|Bw + e_2 b\|^2}, \qquad (8)$$

Để chính quy hóa (8), ta thêm số hạng chính quy Tikhonov [9] vào tử số, bài toán trở thành:

$$\min_{(w,b)\neq 0} \frac{\|Aw + e_1 b\|^2 + \delta \left\| \begin{bmatrix} w \\ b \end{bmatrix} \right\|^2}{\|Bw + e_2 b\|^2}, \quad \delta > 0 \quad (9)$$

Đặt

$$\begin{cases} G & := [A \;\; e_1]^T [A \;\; e_1] + \delta I, \\ H & := [B \;\; e_2]^T [B \;\; e_2], \;\; z := \begin{bmatrix} w \\ b \end{bmatrix}, \end{cases}$$

bài toán tối ưu (9) trở thành:

$$\min_{z\neq 0} r(z) = \frac{z^T G z}{z^T H z}, \quad (10)$$

ở đây G, H là các ma trận đối xứng trong $R^{(n+1)\times(n+1)}$. Ta có

$$\nabla r(z) = 2\frac{(Gz - r(z)Hz)}{z^T H z}, \quad (11)$$

hàm $r(z)$ được gọi là thương Rayleigh [1, tr357]. Hàm này có tính chất $r(\lambda z) = r(z), \;\; \forall z, \;\; \forall \lambda \neq 0$ do đó (10) tương đương $\min_{z\in S(0,\,1)} r(z)$. Vì $r(z)$ liên tục trên tập compact $S(0,\,1)$ nên đạt cực tiểu và cực đại trên đó. Giả sử $r(z)$ đạt cực tiểu và cực đại tương ứng tại $z_{\min}$, $z_{\max}$. Lúc đó $\nabla r(z_{\min}) = \nabla r(z_{\max}) = 0 \Rightarrow Gz_{\min} - r(z_{\min})Hz_{\min} = 0$ và $Gz_{\max} - r(z_{\max})Hz_{\max} = 0$. Như vậy, $r(z_{\min}),\; r(z_{\max}) \in [\lambda_1,\, \lambda_{n+1}] \Rightarrow r(z) \in [\lambda_1,\, \lambda_{n+1}]$ khi $z \in S(0,\; 1)$, ở đây $\lambda_1,\; \lambda_{n+1}$ tương ứng là trị riêng nhỏ nhất và lớn nhất của bài toán trị riêng suy rộng:

$$Gz = \lambda Hz, \quad z \neq 0. \quad (12)$$

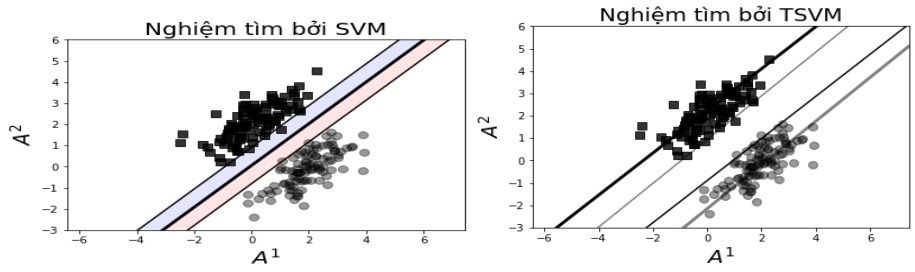
Giả sử rằng các cột của ma trận $[B \;\; e_2]$ là độc lập tuyến tính, nghiệm toàn cục của (10) là véc-tơ riêng của (12) tương ứng với trị riêng nhỏ nhất λ_1. Ký hiệu véc-tơ riêng này là $z^{(1)}$ thì $\left[w^{(1)T} \;\; b^{(1)}\right]^T = z^{(1)}$ xác định siêu phẳng 1 trong (6).

Bằng cách làm tương tự ta sẽ có được siêu phẳng 2.

5. MÁY VÉC-TƠ TỰA SONG SINH

5.1 Trường hợp hai lớp dữ liệu gần tách được tuyến tính

Với bài toán như trong mục 4. Ý tưởng của TWSVM cho bài toán phân loại nhị phân cũng là đi tìm hai siêu phẳng không song song như GEPSVM, nhưng về mặt công thức thì hoàn toàn khác.



***Hình 3**:* SVM và TWSVM

Thực tế, mỗi bài toán QP trong TWSVM là tương tự như SVM, tuy nhiên không phải tất cả các điểm dữ liệu đều xuất hiện trong ràng buộc cùng một lúc. Ta sẽ thu được hai siêu phẳng trong TWSVM bằng cách giải hai bài toán QP sau:

$$(TWSVM1) \quad \begin{cases} \min\limits_{w^{(1)},b^{(1)},q} \dfrac{1}{2}\left(Aw^{(1)} + e_1 b^{(1)}\right)^T \left(Aw^{(1)} + e_1 b^{(1)}\right) + c_1 {e_2}^T q \\ -\left(Bw^{(1)} + e_2 b^{(1)}\right) + q \geq e_2, \quad q \geq 0, \end{cases} \tag{13}$$

và

$$(TWSVM2) \quad \begin{cases} \min\limits_{w^{(2)},b^{(2)},q} \dfrac{1}{2}\left(Bw^{(2)} + e_2 b^{(2)}\right)^T \left(Bw^{(2)} + e_2 b^{(2)}\right) + c_2 {e_1}^T q \\ -\left(Aw^{(2)} + e_1 b^{(2)}\right) + q \geq e_1, \quad q \geq 0. \end{cases} \tag{14}$$

Ở đây $c_1,\ c_2$ là các hệ số phạt để cân bằng sự vi phạm và tổng khoảng cách bình phương trong mỗi bài toán, $e_1 \in R^{m_1},\ e_2 \in R^{m_2}$ là các vector toàn 1. Thuật toán đi tìm hai siêu phẳng đại diện cho hai lớp.

Trước hết ta xét bài toán (13). Ta thấy rằng, biểu thức thứ nhất trong hàm mục tiêu của (13) chính là tổng khoảng cách bình phương của các điểm thuộc lớp 1 tới siêu phẳng 1. Ràng buộc là các điểm thuộc lớp -1 giữ khoảnh cách ít nhất bằng 1 đối với siêu phẳng 1. Biến phụ $q > 0$ tại các điểm vi phạm ràng buộc và bằng 0 tại các điểm không vi phạm. Hàm Lagrange tương ứng với bài toán TWSVM1 (13) được cho bởi:

$$\begin{aligned} L\left(w^{(1)}, b^{(1)}, q, \alpha, \beta\right) = {} & \frac{1}{2}\left(Aw^{(1)} + e_1 b^{(1)}\right)^T \left(Aw^{(1)} + e_1 b^{(1)}\right) + c_1 {e_2}^T q \\ & -\alpha^T\left(-\left(Bw^{(1)} + e_2 b^{(1)}\right) + q - e_2\right) - \beta^T q, \end{aligned} \tag{15}$$

với $\alpha = \left(\alpha_1, \ldots, \alpha_{m_2}\right)^T,\ \beta = \left(\beta_1, \ldots, \beta_{m_2}\right)^T$ là vector nhân tử Lagrange. Giải hệ điều kiện cần và đủ K.K.T của bài toán ta có:

$$0 \leq \alpha \leq c_1 e_2. \tag{16}$$

và

$$\begin{bmatrix} A^T \\ e_1^T \end{bmatrix} [A \ \ e_1] \begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix} + \begin{bmatrix} B^T \\ e_2^T \end{bmatrix} \alpha = 0. \tag{17}$$

Đặt

$$H = [A \;\; e_1],\;\; G = [B \;\; e_2],\;\; u = \begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix}, \qquad (18)$$

(17) có thể viết lại thành

$$H^T H u + G^T \alpha = 0,$$

tức là

$$u = -(H^T H)^{-1} G^T \alpha. \qquad (19)$$

Để đảm bảo tồn tại ma trận nghịch đảo $(H^T H)^{-1}$ ta thêm số hạng chính quy $\epsilon I,\;\; \epsilon > 0$ như trong [2], với I là ma trận đơn vị có số chiều thích hợp, (19) trở thành

$$u = -(H^T H + \epsilon I)^{-1} G^T \alpha. \qquad (20)$$

Tuy nhiên, trong phần sau của bài ta vẫn sử dụng (19) để thuận tiện cho quá trình tìm hiểu bài toán, còn khi lập trình thuật toán ta sẽ sử dụng công thức (20). Thay vào hàm Lagrange ta thu được bài toán đối ngẫu của TWSVM1 như sau:

$$(DTWSVM1) \quad \begin{cases} \max\limits_{\alpha} & e_2^T \alpha - \dfrac{1}{2} \alpha^T G (H^T H)^{-1} G^T \alpha \\ & 0 \le \alpha \le c_1 e_2. \end{cases} \qquad (21)$$

Bằng cách tương tự, ta xét bài toán TWSVM2 và thu được bài toán đối ngẫu sau:

$$(DTWSVM2) \quad \begin{cases} \max\limits_{\gamma} & e_1^T \gamma - \dfrac{1}{2} \gamma^T H (G^T G)^{-1} H^T \gamma \\ & 0 \le \gamma \le c_2 e_1. \end{cases} \qquad (22)$$

và vector $v = \begin{bmatrix} w^{(2)} \\ b^{(2)} \end{bmatrix}$ được cho bởi

$$v = (G^T G)^{-1} H^T \gamma. \qquad (23)$$

Hai ma trận $H^T H$ và $G^T G$ cùng có cỡ $(n+1) \times (n+1)$, trong thực tế số chiều n là nhỏ hơn nhiều so với số phần tử của hai lớp dữ liệu. Hai siêu phẳng tách đạt được là:

$$x^T w^{(1)} + b^{(1)} = 0,\;\; x^T w^{(2)} + b^{(2)} = 0. \qquad (24)$$

Với một điểm dữ liệu mới $x \in \mathbb{R}^n$ ta tính khoảng cách vuông góc từ x tới hai siêu phẳng, x gần siêu phẳng của lớp nào hơn thì thuộc về lớp đó.

Từ hệ điều kiện K.K.T ta thấy rằng, các điểm thuộc lớp -1 tương ứng với $0 < \alpha_i < c_1,\;\; (i = 1, \ldots, m_2)$ nằm trên siêu phẳng $x^T w^{(1)} + b^{(1)} = -1$, những vector thuộc lớp -1 này là những vector tựa của lớp 1. Điều tương tự xảy ra với bài toán TWSVM2.

Đánh giá độ phức tạp: Quan sát hai bài toán đối ngẫu (21) và (22) thấy rằng các ma trận vuông $G(H^T H)^{-1} G^T$ và $H(G^T G)^{-1} H^T$ có cỡ xấp xỉ $\frac{m}{2}$. Trong bài toán đối ngẫu của SVM [3], ma trận vuông có cỡ m. Như vậy nếu ta coi độ phức tạp của SVM là m^3

thì độ phức tạp của TWSVM là xấp xỉ $2(\frac{m}{2})^3$, qua đó thấy rằng tốc độ tính toán của TWSVM là nhanh hơn SVM xấp xỉ 4 lần.

5.2 TRƯỜNG HỢP PHI TUYẾN

Khi hai lớp dữ liệu là không tách được tuyến tính, tương tự như SVM tiêu chuẩn, ta sẽ sử dụng kỹ thuật kernel. Hai mặt phân loại sẽ có dạng:

$$K(x^T, C^T)w^{(1)} + b^{(1)} = 0, \quad K(x^T, C^T)w^{(2)} + b^{(2)} = 0, \qquad (25)$$

với $C = \begin{bmatrix} A \\ B \end{bmatrix}$ và K là một kernel nào đó. Bài toán tối ưu có dạng như sau:

$$(KTWSVM1) \quad \begin{cases} \min\limits_{w^{(1)},b^{(1)},q} \frac{1}{2}\left\|K(A,C^T)w^{(1)} + e_1 b^{(1)}\right\|^2 + c_1 e_2^T q \\ -\left(K(B,C^T)w^{(1)} + e_2 b^{(1)}\right) + q \geq e_2, \quad q \geq 0. \end{cases} \qquad (26)$$

Hàm Lagrange của bài toán:

$$L\left(w^{(1)}, b^{(1)}, q, \alpha, \beta\right) = \frac{1}{2}\left\|K(A,C^T)w^{(1)} + e_1 b^{(1)}\right\|^2 + c_1 {e_2}^T q - \alpha^T\left(-\left(K(B,C^T)w^{(1)} + e_2 b^{(1)}\right) + q - e_2\right) - \beta^T q, \quad (27)$$

Dựa vào hệ điều kiện K.K.T của bài toán ta có:

$$\begin{bmatrix} K(A, C^T)^T \\ e_1^T \end{bmatrix}[K(A, C^T) \;\; e_1]\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix} + \begin{bmatrix} K(B, C^T)^T \\ e_2^T \end{bmatrix}\alpha = 0. \qquad (28)$$

Đặt $S = [K(A, C^T) \;\; e_1]$, $R = [K(B, C^T) \;\; e_2]$, $z^{(1)} = \begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix}$,

biểu thức (28) có thể được viết lại thành

$$S^T S z^{(1)} + R^T \alpha = 0, \text{ hay } z^{(1)} = -(S^T S)^{-1} R^T \alpha. \qquad (29)$$

Bài toán đối ngẫu của KTWSVM1 là:

$$(KDTWSVM1) \quad \begin{cases} \max\limits_{\alpha} \quad e_2^T \alpha - \frac{1}{2}\alpha^T R(S^T S)^{-1} R^T \alpha \\ 0 \leq \alpha \leq c_1 e_2. \end{cases} \qquad (30)$$

Tương tự ta có bài toán tối ưu KTWSVM2 và KDTWSVM2 cho mặt $K(x^T, C^T)w^{(2)} + b^{(2)} = 0$ như sau:

$$(KTWSVM2) \quad \begin{cases} \min\limits_{w^{(2)},b^{(2)},q} \frac{1}{2}\left\|K(B,C^T)w^{(2)} + e_2 b^{(2)}\right\|^2 + c_2 e_1^T q \\ \left(K(A,C^T)w^{(2)} + e_1 b^{(2)}\right) + q \geq e_1, \quad q \geq 0. \end{cases} \qquad (31)$$

$$(KDTWSVM2) \quad \begin{cases} \max\limits_{\gamma} \quad e_1^T \gamma - \frac{1}{2}\gamma^T S(R^T R)^{-1} S^T \gamma \\ 0 \leq \gamma \leq c_2 e_1. \end{cases} \qquad (32)$$

Nghiệm $z^{(2)} = \begin{bmatrix} w^{(2)} \\ b^{(2)} \end{bmatrix} = (R^T R)^{-1} S^T \gamma$. Bằng cách sử dụng công thức Sherman-Morrison-Woodbury [8] cho ma trận nghịch đảo, ta không phải tính nghịch đảo của ma trận $m + 1$ chiều như trên mà chỉ phải tính nghịch đảo của ma trận m_1 chiều và m_2 chiều.

6. CÀI ĐẶT THUẬT TOÁN MÁY VÉC-TƠ TỰA SONG SINH VÀ ÁP DỤNG

Thuật toán TWSVM: *Cho* $\boldsymbol{m}$ *điểm dữ liệu trong* $\boldsymbol{R^n}$ *được biểu diễn bởi ma trận* $\boldsymbol{C^{m \times n}}$. *Giả sử tập dữ liệu gồm hai lớp, lớp {+} gồm* $\boldsymbol{m_1}$ *điểm và lớp {-} gồm* $\boldsymbol{m_2}$ *điểm tương ứng được biểu diễn bởi các ma trận* $\boldsymbol{A^{m_1 \times n}}$ *và* $\boldsymbol{B^{m_2 \times n}}$. *Với K là một kernel nào đó;*

i) *Đặt* $S = [K(A, C^T)\ e_1]$, $R = [K(B, C^T)\ e_2]$, $z^{(1)} = \begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix}$, *với* $e_1 \in R^{m_1 \times 1}, e_2 \in R^{m_2 \times 1}$ *là các vector toàn 1.*

ii) *Xác định* α *thông qua (30).*

iii) *Xác định* $z^{(1)}$ *thông qua (29).*

iv) *Phân loại một điểm x mới dựa vào (24).*

Ngôn ngữ lập trình được sử dụng ở đây là Python 3.8.3. Trong các thư viện của Python mới chỉ cài đặt thuật toán SVM mặc dù đã có khá nhiều cải tiến tốt hơn, tiêu biểu là thuật toán TWSVM. Tôi đã cài đặt thuật toán TWSVM và đưa phần code lên web, bạn đọc quan tâm có thể tìm thấy tại https://github.com/makeho8/python/blob/master. Về cơ bản ta cần sử dụng một vài thư viện của Python:

```
import numpy as np
from cvxopt import matrix, solvers
from sklearn.base import BaseEstimator
K = matrix(GB.dot(np.linalg.inv(HA.T.dot(HA) + 0.0001*I)).dot(GB.T))
q = matrix((-e_B))
G = matrix(np.vstack((-np.eye(m_B), np.eye(m_B))))
h = matrix(np.vstack((np.zeros((m_B,1)), self.c*np.ones((m_B,1)))))
solvers.options['show_progress'] = False
sol = solvers.qp(K, q, G, h)
alpha = np.array(sol['x'])
self.u = -np.linalg.inv(((HA.T).dot(HA) + 0.0001*I)).dot(GB.T).dot(alpha)
```

self.b_A = self.u[-1]

Dưới đây tôi đưa ra kết quả áp dụng thực tế để so sánh giữa TWSVM và SVM tiêu chuẩn. Bạn đọc có thể thấy rằng TWSVM là tốt hơn SVM cả về tốc độ tính toán và khả năng mô phỏng dữ liệu trong hầu hết các trường hợp. Dữ liệu được lấy từ bộ dữ liệu sẵn có UCI [10]. Các hệ số phạt đều được cài đặt là 1. Sử dụng đánh giá chéo 10 lần (10-fold cross validation) để đánh giá độ chính xác kiểm thử cho tất cả các tập dữ liệu. Tập dữ liệu được chia theo tỉ lệ 80/20, Laptop sử dụng là MSI Bravo15.

Bảng 1: Độ chính xác (%)

Dữ liệu	TWSVM	SVM
CMC (1473x9)	68.253 +/- 3.065	66.977 +/- 3.921
Heart (920x13)	79.902 +/- 5.098	80.170 +/- 4.629
Image (2100x18)	84.345 +/- 2.306	76.905 +/- 2.564
Hepatitis (155x19)	82.821 +/- 8.107	78.013 +/- 9.178
Australian (689x14)	85.107 +/- 4.153	78.750 +/- 7.659
BUPA-liver (345x6)	66.283 +/- 7.592	68.016 +/- 10.710
Credit (690x16)	86.416 +/- 2.695	85.513 +/- 3.955
Diabetis (768x9)	77.205 +/- 4.605	74.923 +/- 5.606
Flare-solar (1066x13)	89.793 +/- 2.500	90.379 +/- 2.193
German (999x25)	76.475 +/- 4.446	75.228 +/- 5.326
Ionosphere (350x35)	89.643 +/- 5.858	86.786 +/- 4.532
Spect (265x23)	80.649 +/- 8.113	80.152 +/- 7.070

Bảng 2: Tốc độ chạy (ms)

Dữ liệu	TWSVM	SVM
CMC (1473x9)	*269.06*	*3619.811*
Heart (920x13)	*78.017*	*1297.29*
Image (2100x18)	*1029.228*	*8343.87*
Hepatitis (155x19)	*6*	*34*
Australian (689x14)	*46.011*	*905.202*
BUPA-liver (345x6)	*14*	*161*
Credit (690x16)	*59.015*	*759.171*
Diabetis (768x9)	*64.014*	*846.189*
Flare-solar (1066x13)	*136.032*	*1659.375*
German (999x25)	*148.032*	*1480.331*
Ionosphere (350x35)	*14.002*	*167.037*
Spect (265x23)	*12.002*	*108.023*

7. KẾT LUẬN

Như vậy tư tưởng toán học của máy véc-tơ tựa song sinh thực chất là tìm cách tách các lớp dữ liệu bởi hai siêu phẳng không nhất nhiết song song, trong đó mỗi siêu phẳng là gần nhất với một lớp và xa nhất có thể với lớp còn lại. Bằng một phương pháp nhất quán là sử dụng quy tắc nhân tử Lagrange để giải hai bài toán QP, chúng tôi đã trình bày cơ sở toán học của máy véc-tơ tựa song sinh và các nghiên cứu liên quan như, SVM tiêu chuẩn, PSVM, GEPSVM trong kĩ thuật phân lớp dữ liệu. Cùng với đó là cách cài đặt thuật toán máy véc-tơ tựa song sinh bằng ngôn ngữ lập trình Python, nhằm mục đính đánh giá về tốc độ và khả năng mô phỏng dữ liệu của TWSVM so với SVM tiêu chuẩn, cho các trường hợp khác nhau, từ đơn giản đến phức tạp. Cánh tiếp cận của TWSVM trong trường hợp phi tuyến có thể được giải quyết thông qua bài toán quy hoạch nửa xác định dương và quy hoạch nón bậc hai, đây là hướng nghiên cứu tiếp theo của chúng tôi nhằm mục đích cải tiến thuật toán.

TÀI LIỆU THAM KHẢO

[1]. B.N Parlett (1998), *The symmetric eigenvalue problem*. Philadelphia: SIAM.

[2]. C. Saunders, A. Gammerman, and V. Vovk (1998), Ridge regression learning algorithm in dual variables, Proc. 15th Int'l Conf. *Machine learning*, pp.515-521.

[3]. B. Scholkopf and A. Smola (2002), Learning with kernel. *Cambridge, Mass*.: MIT Press.

[4]. O. L. Mangasarian and E. W. Wild (2006), "Multisurface proximal support vector classification via generalized eigenvalues," *IEEE Trans. Pattern analysis and machine learning*, vol. 28, no. 1, pp. 69-74.

[5]. Jayadeva, R. Khemchandani and S. Chandra (2007), Twin support vector machines for pattern classification. *IEEE Transactions on Pattern Analysis and Machine intelligence*, vol. 29, no. 5.

[6]. G. Fung and O. L Mangasarian (2001), "Proximal support vector machine," *Proc. Seventh Int'l Conf. Knowledge discovery and Data mining*, pp. 77-86.

[7]. O. L. Mangasarian (1994), Nonlinear programming. *SIAM, Philadelphia*, PA.

[8]. G. H. Golub and C. F. Van Loan (1996), Matrix computations. *The John Hopkins University Press*, Baltimore, Maryland, 3rd edition.

[9]. A.N. Tikhonov and V. Y. Arsen (1977), Solutions of Ill-Posed Problems. *New York*: John Wiley and Sons.

[10]. C. L. and C.J. Merz (1998), "UCI Repository for Machine Learning Database," *Dept. of information and computer sciences*, Univ. of California, Irvine, http://www.ics.uci.edu/mlearn/MLRepository.html.

TWIN SUPPORT VECTOR MACHINE AND APPLICATION

Nguyen The Cuong

Department of Mathematics, Communications University Nha Trang, Khanh Hoa

Email: nckcbnckcb@gmail.com

ABSTRACT

The support vector machines are popular classification technique and has been successfully applied to many different areas of the life. Because this is a pratical problem, we chose to study, aiming to find more applications and better improvings. There are some typical improvings such as Proximal Support Vector Machine (PSVM), Proximal Support Vector Machine Via Generalized Eigenvalues (GEPSVM) and Twin Support Vector Machine (TWSVM). SVM and TWSVM can be solved by using Lagrange duality problem, but TWSVM used two nonparallel hyperplanes to separate data classes. This paper airms to show mathematical foundation of PSVM, GEPSVM and TWSVM, and we have setted up TWSVM algorithm by using Python programming to evaluate effect of TWSVM compared to standard SVM.

Keywords: Support vector machines, Twin support vector machines.

Nguyễn Thế Cường sinh ngày 09/12/1986 tại Sơn La. Năm 2009, ông tốt nghiệp cử nhân ngành Toán Tin tại Trường Đại học Sư phạm Hà Nội. Năm 2014, ông tốt nghiệp thạc sỹ ngành Toán Giải tích tại Trường Đại học khoa học, ĐH Huế. Tháng 12/2018, ông bắt đầu làm NCS tại Trường Đại học Khoa học, ĐH Huế. Hiện ông đang công tác tại Trường ĐH Thông tin Liên lạc, Nha Trang, Khánh Hòa.

Lĩnh vực nghiên cứu: Khoa học máy tính.