

サポートベクターマシンの使い方と Web アプリケーション攻撃検出への応用

松 田 健

要 旨

サポートベクターマシンは2値分類を行うために利用される機械学習の一種であり、データを実ユークリッド空間上の点で表現し、ユークリッド空間上を2つに分割する超平面を求めることでデータの2値分類を実現する。サポートベクターマシンによる分類手法は理論的な観点からも効率の良い分類ができることが知られており、広い分野で応用されている。本稿では、サポートベクターマシンの使い方を Web アプリケーション攻撃の検出問題を例に用いて紹介する。

1 はじめに

サポートベクターマシン (Support Vector Machine) は1995年にAT & Tの[1]らによって考案されたデータの識別や分類を実現する方法の1つである。サポートベクターマシンはその英語の頭文字からSVMと略記されることが多く、本稿でも以下サポートベクターマシンのことをSVMと表記する。SVMはデータを2種類のパターンに分類できる問題に利用される方法で、その他いくつか存在する分類法の中でも分類の精度を高く実現できる手法として知られており、広く利用されている。SVMがデータを分類する手法は一言でいえば、データの特徴を数値化してある実ユークリッド空間の点として扱い、それらの点を超平面で分離して2種類のパターンに分類するというものである。SVMの応用例は非常に多岐に渡っている。例えば医療情報に関連する研究としては[2][3][4][5][6][7][8][9]，社会工学に関連する研究としては[10][11][12]，情報技術に関連する研究として[13][14][15][16][17]などの応用研究が存在する。

本稿ではSVMを理解するために必要となる数学的知識の紹介を行い、いくつかの簡単な例を示してSVMがどのようにしてデータを分類するかということについて考察を行う。SVMの適用例として扱う例は、著者の研究テーマの1つであるWebアプリケーション攻撃に関するものを利用する。そのため著者の研究紹介も簡単に行うが、本稿には一

サイバー大学 IT 総合学部・専任講師

原稿受付日：2012年11月28日

原稿受理日：2013年1月10日

部未発表の内容も含まれている。

以下、本稿の構成を紹介する。

2 章では SVM の理解に必要な数学を復習する。

3 章では SVM の概要を紹介する。

4 章では SQL インジェクション攻撃の概要と攻撃検出の従来研究についてまとめる。

5 章では SVM を用いた SQL インジェクション攻撃の検出の例を紹介し、SVM による攻撃検出の考察を行う。

そして最後の 6 章でまとめを行う。

2 SVM の数理

この章では SVM のデータ分類法を理解するために必要な数学の復習を行う。SVM はデータを実ユークリッド空間上の点として扱い、それらの点を 2 つの部分空間に分ける超平面を求めることでデータの分類を実現する。したがって、ユークリッド空間の定義、ユークリッド空間上の点と超平面に関する知識は最低限必要となる。さらに、SVM ではデータを 2 つのパターンに分類する際の超平面の計算が必要となるが、その際にデータ分類に最適な超平面を探すという最適化問題を解くことになる。本章では、その最適化計算を行う数理的な手法についても解説を行う。ただし、解説が難しくなりすぎることを防ぐために、数学的厳密性については一部犠牲にする。

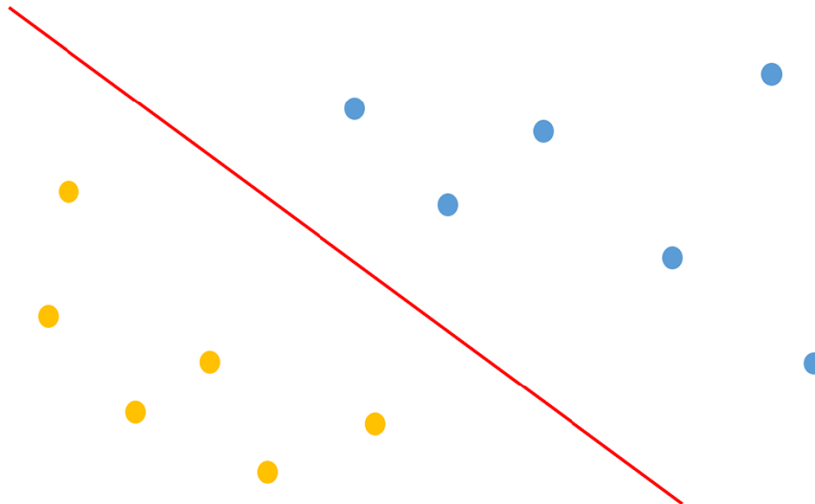


図 1 超平面によるデータ分類のイメージ

2.1 実ユークリッド空間とその距離の性質について

まず、ユークリッド空間の定義について述べる。

Definition 1. n 個の実数 $\mathbf{R}$ の組 $(x_1, x_2, \dots, x_n)$ を考える。この組からなる集合を

$$\mathbf{R}^n = \{(x_1, x_2, \dots, x_n) \mid x_i \in \mathbf{R} (i = 1, 2, \dots, n)\}$$

と定義し、2 点 $\mathbf{x} = (x_1, x_2, \dots, x_n)$, $\mathbf{y} = (y_1, y_2, \dots, y_n)$ の距離を

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

で定義した空間 $\mathbf{R}^n$ のことを n 次元実ユークリッド空間という。

$(x_1, x_2, \dots, x_n)$ のことを $\mathbf{R}^n$ の座標、または点という。以下、距離 $d(\mathbf{x}, \mathbf{y})$ について簡単に考察を行う。まず、 $i = 1, 2, \dots, n$ に関して、 $x_i, y_i \in \mathbf{R}$ であることから、 $(x_i - y_i)^2 \geq 0$ である。したがって、

$$d(\mathbf{x}, \mathbf{y}) \geq 0 \quad (1)$$

である。また、 $(x_i - y_i)^2 \geq 0$ の等号成立条件は $x_i = y_i$ であるから、 $d(\mathbf{x}, \mathbf{y}) = 0$ ならば、 $x_i = y_i$ である。逆に、 $x_i = y_i$ ならば $d(\mathbf{x}, \mathbf{y}) = 0$ であることが分かる。したがって、式 (1) の等号成立条件は、 $\mathbf{x} = \mathbf{y}$ である。また、 $(x_i - y_i)^2 = (y_i - x_i)^2$ であるから、

$$d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x}) \quad (2)$$

も成り立つ。さらに、 $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbf{R}^n$ を考えると、

$$d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z}) \geq d(\mathbf{x}, \mathbf{z}) \quad (3)$$

が成り立つ。

ユークリッド空間は他にも重要な性質をもっているが、SVM の数理を理解するために必要な知識は主に距離に関するものだけであるため、ユークリッド空間の性質に関する議論は以上とする。SVM を理解するには超平面の概念を理解することが重要である。

Denition 2. $\mathbf{R}^n$ における超平面 H とは、ある実数 $a_1, a_2, \dots, a_n, b$ を用いて、

$$H = \{(x_1, x_2, \dots, x_n) \mid \sum_{i=1}^n a_i x_i + b = 0\}$$

を満たす $\mathbf{R}^n$ の部分集合のことをいう。

SVM ではこの超平面をデータの分類で利用する。以下、超平面の例を紹介する。そのために、 $\mathbf{R}^n$ における直線の定義をしておく。

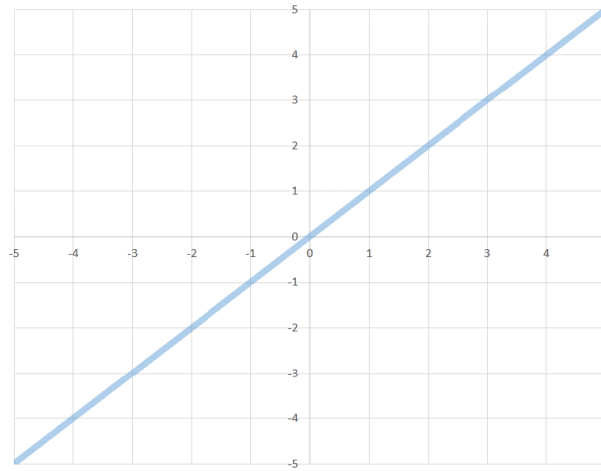


図2 直線 $y = x$ のグラフ

Denition 3. $\mathbf{R}^n$ 内の直線とは, ある実数 $a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_n$ を用いて,

$$L = \{(x_1, x_2, \dots, x_n) \mid x_i = ta_i + b_i (i = 1, 2, \dots, n)\}$$

を満たす $\mathbf{R}^n$ の部分集合のことをいう。ただし, $t \in \mathbf{R}$ である。

Example 1. (x, y) を座標にもつ $\mathbf{R}^2$ における直線 $y = x$ は, $\mathbf{R}^2$ における超平面でもある。

SVM では $\mathbf{R}^n$ における点とそれぞれの点から超平面までの距離を考えることが重要となる。 $\mathbf{R}^n$ 上の点 P から超平面 H までの距離は, 点 P から超平面 H に向かって引いた直線 L の長さに対応させれば良いが, そのような直線は無数に引くことができるため点 P から超平面 H までの距離がただ1つに定まるように定義する必要がある。そのために, 超平面 H と垂直に交わり, かつ点 P を通る直線 l を考え, 直線 l における点 P から超平面 H までの線分の長さを, 点 P から超平面 H までの距離として定義することにする。このとき, 以下の定理が成り立つ。

Theorem 1. $\mathbf{R}^n$ 上の点 $P = (p_1, p_2, \dots, p_n)$ と超平面 $H = \{(x_1, x_2, \dots, x_n) \mid \sum_{i=1}^n a_i x_i + b = 0\}$ までの距離を D とおくと

$$D = \frac{\left| \sum_{i=1}^n a_i p_i + b \right|}{\sqrt{\sum_{i=1}^n a_i^2}}$$

となる。

この定理の証明は容易である。いま超平面 H をベクトル $\mathbf{a}, \mathbf{x} \in \mathbf{R}^n$ の内積 $\langle \mathbf{a}, \mathbf{x} \rangle + b = 0$ で表し, H 上の点 $Q(q_1, q_2, \dots, q_n)$ と H の外にある点 $P(p_1, p_2, \dots, p_n)$ を考える。このとき,

点 P から点 Q までの距離 D は 2 つのベクトル $\mathbf{q} = (q_1, q_2, \dots, q_n)$ と $\mathbf{p} = (p_1, p_2, \dots, p_n)$ を用いて

$$D = |\mathbf{q} - \mathbf{p}|^2$$

で表される。 $|\langle \mathbf{q} - \mathbf{p}, \mathbf{a} \rangle| \leq |\mathbf{q} - \mathbf{p}| |\mathbf{a}|$ が成り立つことを利用すると

$$D \geq \frac{|\langle \mathbf{p}, \mathbf{a} \rangle + b|}{|\mathbf{a}|}$$

となることが分かる。

SVM を理解するには、この定理の他に $\mathbf{R}^n$ の部分空間とその領域に関する知識も必要となる。 $\mathbf{R}^n$ の部分集合

$$V = \{x \in \mathbf{R}^n | f(x) = 0\}$$

を考える。ここで、 $f(x) = 0$ は x を変数とする多項式である。 $\mathbf{R}^n$ はその部分集合 V を用いて、以下の 2 つの部分空間に分割される。

$$V_+ = \{x \in \mathbf{R}^n | f(x) > 0\}$$

$$V_- = \{x \in \mathbf{R}^n | f(x) < 0\}$$

これにより、 $\mathbf{R}^n = V_+ \cup V \cup V_-$ と分割できることが分かる。一般的に、高次元の空間において V_+ や V_- を考えることは困難である。したがって、 $\mathbf{R}^2$ において V_+ や V_- がどのようなものになっているか例を用いて説明する。

Example 2. $\mathbf{R}^2$ における部分集合

$$V = \{(x, y) \in \mathbf{R}^2 | x - y = 0\}$$

を考える。部分集合 V は原点を通る直線を表す。いま、

$$V_+ = \{(x, y) \in \mathbf{R}^2 | x - y > 0\}$$

$$V_- = \{(x, y) \in \mathbf{R}^2 | x - y < 0\}$$

を考えると、図 4 のような平面の分割が得られる。

SVM の基本的な考え方は、Example 2 にあるように、 $\mathbf{R}^n$ 上の関数 $f(x) = 0$ を用いて、データを部分集合 V_+ と V_- に分類するというものである。

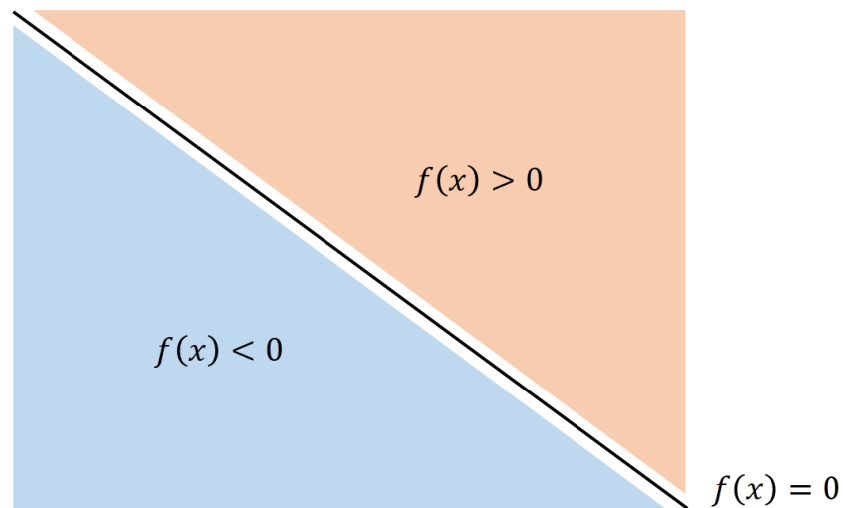


図3 空間の分割

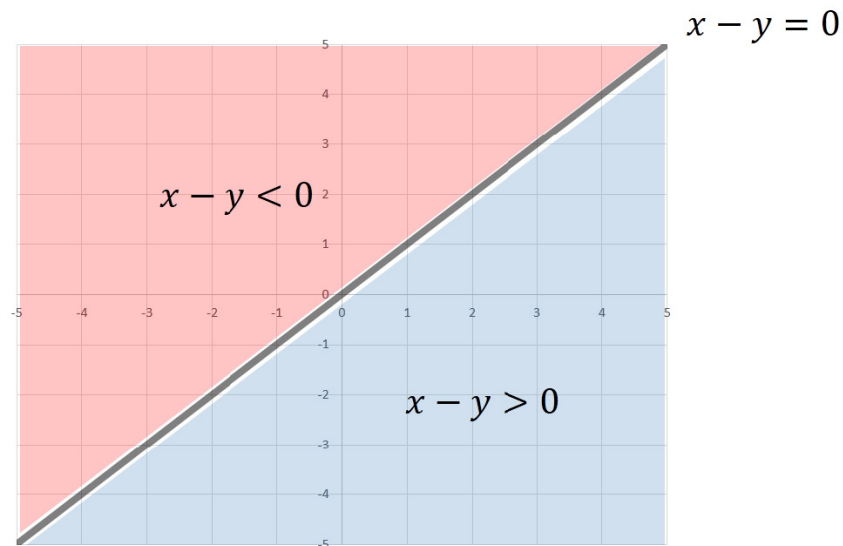


図4 直線 $y = x$ による平面の分割

2.2 最大化・最小化の数理

この節では、 $\mathbf{R}^n$ 上に定義された関数 $f(x)$, ($x \in \mathbf{R}^n$) の最大値・最小値を求める方法について紹介する。ワイヤシュトラスの定理により、 $\mathbf{R}^n$ 上の有界閉集合 K 上の実数値連続関数は最大値と最小値を K でとることが知られている。本稿で扱う関数は、最大値か最小値のどちらかは必ず存在する関数について考えていく。以下、関数の最小値を求める問題について考えて行くことにする。関数 $f(x)$ を最小にする点 p とは、 $x \neq p$ なるすべての x に対して、

$$f(p) \leq f(x)$$

を満足するものであり、 $f(p)$ を関数 $f(x)$ の最小値という。

Example 3. $\mathbf{R}$ 上連続の関数 $f(x) = x^2 - 1$ を考える x は実数であるから、 $x^2 \geq 0$ であり、 $x = 0$ のときに限り等号成立が成立する。したがって、すべての実数 x に対して $x^2 - 1 \geq -1$ が成り立つことがわかり、 $x = 0$ のとき関数 $f(x)$ は最小値 $f(0) = -1$ をとることが分かる。

Example 4. $\mathbf{R}^2$ 上連続の関数 $f(x, y) = x^2 + y^2 - 1$ を考える。 x, y は実数であるから、 $x^2 \geq 0, y^2 \geq 0$ であり、 $x = 0, y = 0$ のときに限り等号成立が成立する。したがって、すべての実数 x, y に対して $x^2 + y^2 - 1 \geq -1$ が成り立つことがわかり、 $(x, y) = 0$ のとき関数 $f(x, y)$ は最小値 $f(0, 0) = -1$ をとることが分かる。

関数 $f(x)$ の最小値を見つける際、もし関数 $f(x)$ が 1 階偏微分可能であれば、連立方程式

$$\frac{\partial f}{\partial x_i}(x) = 0$$

を満足する点 x を求めることで最小値が求められる場合もある。しかしながら、必ずしも偏微分計算によって最小値が求められるとは限らない。

Example 5. $\mathbf{R}^2$ の関数 $f(x, y) = x^2 - y^2$ を考える。この関数は連続かつ偏微分可能であり、

$$\begin{aligned}\frac{\partial f}{\partial x}(x, y) &= 2x \\ \frac{\partial f}{\partial y}(x, y) &= -2y\end{aligned}$$

から、 $\frac{\partial f}{\partial x}(x, y) = 0, \frac{\partial f}{\partial y}(x, y) = 0$ を解いて $x = 0, y = 0$ が得られ、 $f(0, 0) = 0$ であることがわかる。ところが、例えば $x = 0, y = 1$ のとき、 $f(0, 1) = 0^2 - 1^2 = -1$ であることがわかり、 $f(0, 0) \geq f(0, 1)$ となる点が存在することが分かる。 $f(x, y) = x^2 - y^2$ 上の点 $(0, 0)$ は鞍点と呼ばれ、極大点でも極小点でもない点である。

また Example 5 の例の他に、連続関数 $f(x)$ は極大点や極小点を複数もっている場合があるため、偏微分から最小値を求める方法はあまり有効的な手法であるとはいえない。さらに Example 6 の例に示すように、何らかの制約条件が付いている場合も存在する。

Example 6. $\mathbf{R}^2$ の関数 $f(x, y) = x^2 + y^2$ に対して制約条件 $y = x - 1$ を与えたもとの最小値を求める問題を考える。制約条件の式 $y = x - 1$ を $f(x, y) = x^2 + y^2$ に代入すると、

$$f(x, y) = f(x, x - 1) = x^2 + (x - 1)^2 = 2x^2 - 2x + 1$$

を得る。

$$\begin{aligned} 2x^2 - 2x + 1 &= 2(x^2 - x) + 1 \\ &= 2\left(x - \frac{1}{2}\right)^2 + \frac{1}{2} \end{aligned}$$

と変形すると, $\left(x - \frac{1}{2}\right)^2 \geq 0$ で等号成立が $x = \frac{1}{2}$ となり, 関数 $f(x, y)$ は $(x, y) = \left(\frac{1}{2}, -\frac{1}{2}\right)$ のときに最小値 $f\left(\frac{1}{2}, -\frac{1}{2}\right) = \frac{1}{2}$ をとることが分かる。

Example 6 では容易に最小値が計算できる例を紹介したが, 実際に SVM を計算するときには高次元の空間上での計算が必要になったり, 式が複雑で手計算が出来ないといった問題が現れる。そこである制約条件のもとで関数を最大化したり最小化する方法として以下のラグランジュの未定乗数法が知られている。

Theorem 2. $\mathbf{R}^n$ 上の関数 $f(\mathbf{x})$ が点 $\mathbf{s} = (s_1, s_2, \dots, s_n)$ において条件

$$g_i(x_1, x_2, \dots, x_n) = 0 \quad (i = 1, 2, \dots, m)$$

のもとで極値をとるものとする。もし, 関数 $\{g_1, g_2, \dots, g_m\}$ のヤコビ行列 $\left(\frac{\partial g_i}{\partial x_j}\right)_{1 \leq i, j \leq m}$ の階数が m で,

- (1) 点 $\mathbf{s} = (s_1, s_2, \dots, s_n)$ の近傍において $f(\mathbf{x}), g_1(\mathbf{x}), \dots, g_m(\mathbf{x})$ は連続かつ微分可能
 - (2) $\{\text{grad } g_1(\mathbf{s}), \text{grad } g_2(\mathbf{s}), \dots, \text{grad } g_m(\mathbf{s})\}$ は 1 次独立
- であるとき,

$$\text{grad } f(\mathbf{a}) = \sum_{i=1}^m \lambda_i \text{grad } g_i(\mathbf{a})$$

を満足する $\lambda_1, \lambda_2, \dots, \lambda_m$ が存在する。ただし,

$$\text{grad } f(\mathbf{x}) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n}\right)$$

である。

以下, ラグランジュの未定乗数法を証明するための概要について触れておく。

$m \times n$ 行列

$$M = \begin{pmatrix} \frac{\partial g_1}{\partial x_1}(\mathbf{s}) & \cdots & \frac{\partial g_1}{\partial x_{n-m+1}}(\mathbf{s}) & \cdots & \frac{\partial g_1}{\partial x_n}(\mathbf{s}) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial g_i}{\partial x_1}(\mathbf{s}) & \cdots & \frac{\partial g_i}{\partial x_{n-m+1}}(\mathbf{s}) & \cdots & \frac{\partial g_i}{\partial x_n}(\mathbf{s}) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial g_m}{\partial x_1}(\mathbf{s}) & \cdots & \frac{\partial g_m}{\partial x_{n-m+1}}(\mathbf{s}) & \cdots & \frac{\partial g_m}{\partial x_n}(\mathbf{s}) \end{pmatrix}$$

を考える。 M は $\text{grad } g_i(\mathbf{s})$ を第 i 行ベクトルとする行列である。定理の仮定 (2) から行列 M の階数は m であることが分かり、行列を転置しても階数が変わらないことから、行列 M の適当な m 個の列ベクトルを選べばそれらのベクトルは 1 次独立である。ここで、行列 M を 2 つの行列 M_1, M_2 に分割する。

$$M_1 = \begin{pmatrix} \frac{\partial g_1}{\partial x_1}(\mathbf{s}) & \cdots & \frac{\partial g_1}{\partial x_{n-m}}(\mathbf{s}) \\ \vdots & \ddots & \vdots \\ \frac{\partial g_m}{\partial x_1}(\mathbf{s}) & \cdots & \frac{\partial g_m}{\partial x_{n-m}}(\mathbf{s}) \end{pmatrix}$$

$$M_2 = \begin{pmatrix} \frac{\partial g_1}{\partial x_{n-m+1}}(\mathbf{s}) & \cdots & \frac{\partial g_1}{\partial x_n}(\mathbf{s}) \\ \vdots & \ddots & \vdots \\ \frac{\partial g_m}{\partial x_{n-m+1}}(\mathbf{s}) & \cdots & \frac{\partial g_m}{\partial x_n}(\mathbf{s}) \end{pmatrix}$$

M_1 は $m \times (n-m)$ 行列で、 M_2 は m 次の正則行列である。さらに $\mathbf{R}^n$ の座標 $\mathbf{x}$ を $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$ と分け、 $\mathbf{x}_1 \in \mathbf{R}^{n-m}$, $\mathbf{x}_2 \in \mathbf{R}^m$ とする。このとき陰関数定理から、 $\mathbf{s} = (\mathbf{s}_1, \mathbf{s}_2) \in \mathbf{R}^{n-m} \times \mathbf{R}^m$ の近傍において連続かつ微分可能な関数 $\varphi: \mathbf{R}^{n-m} \rightarrow \mathbf{R}^m$ が存在して $\mathbf{x}_2 = \varphi(\mathbf{x}_1)$ が成り立つ。具体的には、

$$\begin{pmatrix} \frac{\partial \varphi_1}{\partial x_{1,1}} & \cdots & \frac{\partial \varphi_1}{\partial x_{1,n-m}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \varphi_m}{\partial x_{1,1}} & \cdots & \frac{\partial \varphi_m}{\partial x_{1,n-m}} \end{pmatrix} = -M_2^{-1}M_1$$

これにより $f(\mathbf{x}) = f(\mathbf{x}_1, \varphi(\mathbf{x}_1))$ とすることで元の制約つき問題と等価となり、 $F(\mathbf{x}_1) = f(\mathbf{x}_1, \varphi(\mathbf{x}_1))$ とおくと $\text{grad } F = 0$ が極値の条件となる。一方、

$$\left(\frac{\partial F}{\partial x_{1,1}}, \cdots, \frac{\partial F}{\partial x_{1,n-m}} \right) = \left(\frac{\partial f}{\partial x_{1,1}}, \cdots, \frac{\partial f}{\partial x_{1,n-m}} \right) - \left(\frac{\partial f}{\partial x_{2,n-m+1}}, \cdots, \frac{\partial f}{\partial x_{2,n}} \right) M_2^{-1} M_1$$

であるから、 $\lambda = \left(\frac{\partial f}{\partial x_{2,n-m+1}}, \cdots, \frac{\partial f}{\partial x_{2,n}} \right) M_2^{-1}$ とおけば、

$$\left(\frac{\partial f}{\partial x_{1,1}}, \cdots, \frac{\partial f}{\partial x_{1,n-m}} \right) = \lambda M_1$$

$$\left(\frac{\partial f}{\partial x_{2,n-m+1}}, \cdots, \frac{\partial f}{\partial x_{2,n}} \right) = \lambda M_2$$

となり, λ は m 次元のベクトルであるから $\lambda = (\lambda_1, \dots, \lambda_m)$ とおくと,

$$\text{grad } f = \lambda M = \sum_{i=1}^m \lambda_i \text{grad } g_i$$

本稿では, SVM による SQL インジェクション攻撃の分類を簡単な例を用いて行う。そのために, 本稿で扱う関数は主に 2 変数関数となる。そこで, 2 変数関数 $z = f(x, y)$ の極値に関する性質をまとめておく。以下, $f_x = \frac{\partial f}{\partial x}$, $f_y = \frac{\partial f}{\partial y}$, $f_{xx} = \frac{\partial^2 f}{\partial x^2}$, $f_{yy} = \frac{\partial^2 f}{\partial y^2}$, $f_{xy} = \frac{\partial^2 f}{\partial x \partial y}$, $f_{yx} = \frac{\partial^2 f}{\partial y \partial x}$ とおく。

Theorem 3. 関数 $z = f(x, y)$ において, $f_x(a, b) = f_y(a, b) = 0$ を満たす点 (a, b) において, 行列

$$H(a, b) = \begin{pmatrix} f_{xx}(a, b) & f_{xy}(a, b) \\ f_{yx}(a, b) & f_{yy}(a, b) \end{pmatrix}$$

を考えたとき,

- $\det H(a, b) > 0$ かつ $f_{xx}(a, b) < 0$ ならば関数 $z = f(x, y)$ は (a, b) で極大,
- $\det H(a, b) > 0$ かつ $f_{xx}(a, b) > 0$ ならば関数 $z = f(x, y)$ は (a, b) で極小,
- $\det H(a, b) < 0$ ならば関数 $z = f(x, y)$ は (a, b) で鞍点,
- $\det H(a, b) = 0$ のときは極値に関する情報はこの方法では分らない。

となる。

ラグランジュの未定乗数法で制約条件 $g(x, y) = 0$, $g_x(a, b) \neq 0$, $g_y(a, b) \neq 0$ のもとで関数 $z = f(x, y)$ の極値を計算する場合は, $F(x, y, \lambda) = f(x, y) - \lambda g(x, y)$ とおいて連立一次方程式

$$F_x(x, y, \lambda) = F_y(x, y, \lambda) = F_\lambda(x, y, \lambda) = 0$$

を解けばよい。しかしながら, ラグランジュの未定乗数法は $F(x, y, \lambda)$ の停留点 $(a, b, \hat{\lambda})$ を求めているに過ぎないため, この場合は

$$H(a, b, \hat{\lambda}) \begin{vmatrix} 0 & g_x(a, b) & g_y(a, b) \\ g_x(a, b) & F_{xx}(a, b, \hat{\lambda}) & F_{xy}(a, b, \hat{\lambda}) \\ g_y(a, b) & F_{yx}(a, b, \hat{\lambda}) & F_{yy}(a, b, \hat{\lambda}) \end{vmatrix}$$

を計算して $H(a, b, \hat{\lambda}) > 0$ ならば極大値, $H(a, b, \hat{\lambda}) < 0$ ならば極小値, $H(a, b, \hat{\lambda}) = 0$ ならば極値の判定はできない, ということになる。

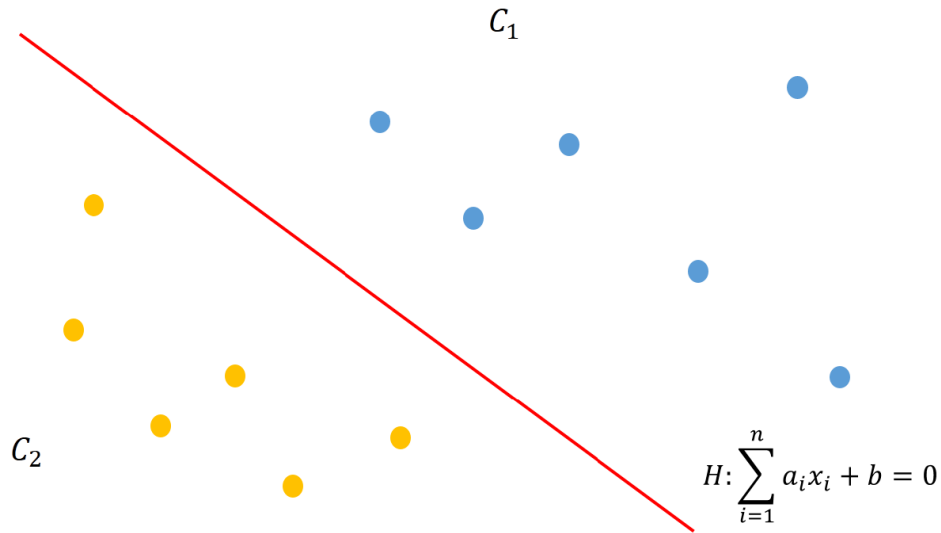


図5 データ分類のイメージ

3 SVM の概要

SVM は以下の図5のように、あるユークリッド空間上の超平面 H を用いてデータを2つのパターンに分類する手法の一つである。超平面 H はデータを2つのグループに分けるものであることから、分離超平面と呼ばれている。いま2つのグループに分けることができる M 個の何らかのデータが与えられ、それをもとに $\mathbf{R}^n$ 上の M 個の点 $\{(x_{1m}, x_{2m}, \dots, x_{nm})\}_{m=1}^M$ が与えられたとする。SVM では、

- (1) 既にどちらのグループに属するかが明らかになっているデータを用意し、
- (2) これらのデータから分離超平面を求め、

(3) どちらのグループに属するか分からないデータを分離超平面を用いて分類するという手続きのもとでデータのグループ分けを行う。SVM では、データを2つのグループに分類することを目的とするため、2つのグループを C_1, C_2 と表し、例えば C_1 に属する点には $y = 1$ というラベルをつけ、 C_2 に属する点には $y = -1$ というラベルをつけることでデータがどちらのグループに属しているのかを表現することができる。以上の準備のもとでSVMは、 C_1, C_2 のどちらに属するか分からないデータ $\hat{x}$ 、つまりラベル y の値が1か-1か分からないデータが与えられたときに、分離超平面 H を基準に $\hat{x}$ が C_1, C_2 のどちらに属するデータかを判断するものであると言える。したがって、サポートベクターマシンを用いてデータ分類を行うためには、データをもとにして作られた $\mathbf{R}^n$ 上の M 個の点から、分離超平面

$$H = \{(x_1, x_2, \dots, x_n) \mid \sum_{i=1}^n a_i x_i + b = 0\}$$

の方程式を定めること、より具体的には H を定義するための係数 $a_1, a_2, \dots, a_n, b$ の値をデータから求める必要がある。この章では、以下、分離超平面の求め方について紹介する。

3.1 線形分離可能な場合

M 個のデータを $\mathbf{x}_m = (x_{1m}, x_{2m}, \dots, x_{nm})$ ($m = 1, 2, \dots, M$) とおき、それぞれのデータにラベル

$$y_m = \begin{cases} 1 & (\mathbf{x}_m \in C_1) \\ -1 & (\mathbf{x}_m \in C_2) \end{cases}$$

をつける。また、分離超平面の方程式を $f(\mathbf{x}) = \sum_{i=1}^n a_i x_i + b$ とおき、 $\mathbf{x}_m$ が C_1 に属する場合 ($\mathbf{x}_m \in C_1$) は

$$f(\mathbf{x}_m) \geq 1$$

$\mathbf{x}_m$ が C_2 に属する場合 ($\mathbf{x}_m \in C_2$) は

$$f(\mathbf{x}_m) \leq -1$$

であると定義する。こうすることで、 $\mathbf{x}_m \in C_1$ であるとき $y=1$, $\mathbf{x}_m \in C_2$ であるとき $y = -1$ であることからすべての $m = 1, 2, \dots, M$ に対して

$$y_m f(\mathbf{x}_m) \geq 1$$

が成り立つことが分かる。このように定義することで、図 6 にあるようにデータを分類できることが分かる。図 6 のように超平面を用いてデータの分類が可能な場合、線形分離可能という。超平面が 1 次（線形）の方程式で定義されることから線形分離可能と呼ばれている。しかしながら、どのようなデータでもこのように分類できるとは限らない。例えば、図 7 のようにデータが平面上に存在する場合も考えられる。このような場合は、 $\mathbf{R}^n$ 上の点 $\mathbf{x}_m$ を非線形な関数 $\varphi(\mathbf{x})$ を用いて、 $\mathbf{R}^s$ 上の点 $(\varphi_1(\mathbf{x}_m), \varphi_2(\mathbf{x}_m), \dots, \varphi_s(\mathbf{x}_m))$ とし、図 6 のように分離超平面が求められる空間で考える方法がとられている。この方法に関しては 3.2 の線形分離不可能な場合で取り扱う。データが線形分離可能となれば、以下の手順で分離超平面 H を決定する。分離超平面 H は、 C_1 に属する点の中で最も H に近い点と H までの距離と、 C_2 に属する点の中で最も H に近い点と H までの距離を最大にする超平面 H がデータの分類に最適であると考えられる。前章の定理 1 より、 $\mathbf{R}^n$ 上の点 $\mathbf{x}_m = (x_{1m}, x_{2m}, \dots, x_{nm})$ から分離超平面 H までの距離は

$$\frac{\left| \sum_{i=1}^n a_i x_{im} + b \right|}{\sqrt{\sum_{i=1}^n a_i^2}}$$

で定義され、 C_1, C_2 に属する点の中で最も H に近い点は、関数 $f(\mathbf{x})$ の定義から

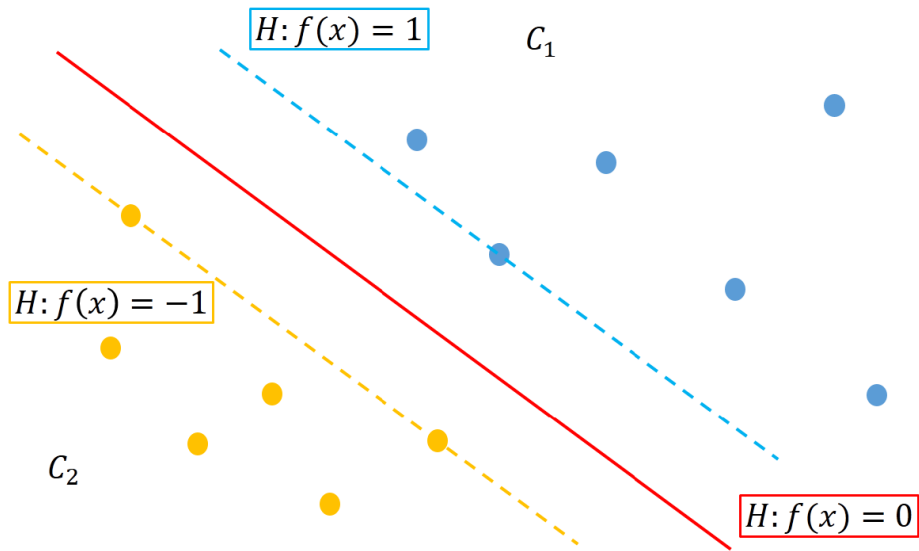


図 6 SVM によるデータ分類のイメージ

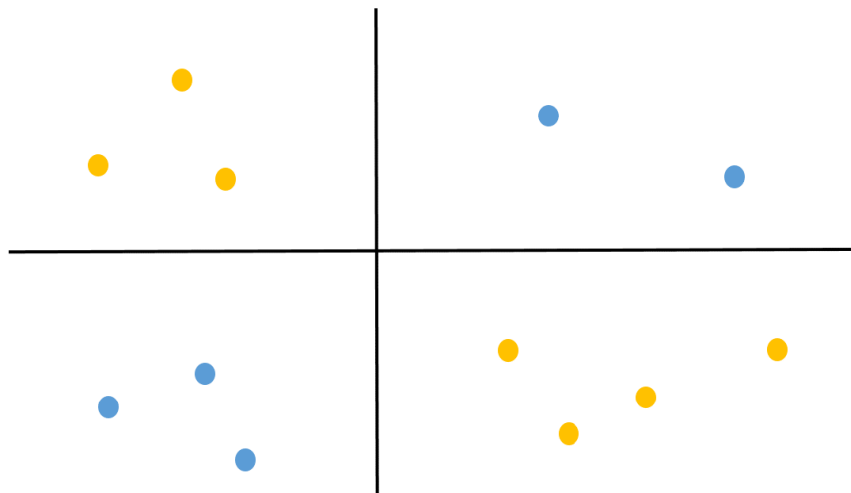


図 7 線形分離できない場合のパターン

$$\left| \sum_{i=1}^n a_i x_{im} + b \right| = 1$$

を満足することが分かる。すると、 C_1 に属する点の中で最も H に近い点と H までの距離は

$$d(a_1, a_2, \dots, a_n, b) = \frac{1}{\sqrt{\sum_{i=1}^n a_i^2}}$$

で与えられ, $d(a_1, a_2, \dots, a_n)$ を最大にする $a_1, a_2, \dots, a_n, b$ を求めることで, 分離超平面 H が求められ, これを用いてデータ分類を行うものが SVM である。ただし, 単純に $d(a_1, a_2, \dots, a_n)$ を最大化すればよいわけではなく, データは $m = 1, 2, \dots, M$ に対して, 不等式

$$y_m f(x_m) \geq 1$$

満足する必要があるため, これらの制約条件のもとで $d(a_1, a_2, \dots, a_n)$ を最大化することを考える。ここで, $d(a_1, a_2, \dots, a_n)$ を最大化することは, $\|a\|^2 = \sum_{i=1}^n a_i^2$ を最小化することと同値であることから, ラグランジュの未定乗数法を用いて $d(a_1, a_2, \dots, a_n)$ を最大とする $a_1, a_2, \dots, a_n, b$ を決定する。具体的には, M 個のデータ $(x_{11}, \dots, x_{n1}), \dots, (x_{1M}, \dots, x_{nM})$ が与えられた下では,

$$L(a, b, \lambda) = \frac{1}{2} \|a\|^2 - \sum_{m=1}^M \lambda_m \{y_m (\langle a, x_m \rangle + b - 1)\}$$

を最小にする問題を考えることになり, $i = 1, 2, \dots, n$ について

$$\begin{aligned} \frac{\partial L}{\partial a_i} &= a_i - \sum_{m=1}^M \lambda_m y_m x_{im} = 0 \\ \frac{\partial L}{\partial b} &= \sum_{m=1}^M \lambda_m y_m = 0 \end{aligned}$$

から,

$$L(a, b, \lambda) = F(\lambda) = \sum_{m=1}^M \lambda_m - \frac{1}{2} \sum_{m_1, m_2=1}^M \lambda_{m_1} \lambda_{m_2} y_{m_1} y_{m_2} \langle x_{m_1}, x_{m_2} \rangle$$

が得られ, 結局 $F(\lambda)$ を

$$\begin{aligned} \sum_{m=1}^M \lambda_m y_m &= 0 \\ \lambda &\geq 0 \end{aligned}$$

の条件の下で最大化する問題になることが分かる。この問題を考えることで a が求まる。一方, $a_i = \sum_{m=1}^M \lambda_m y_m x_{im}$ であることから, $\lambda_m > 0$ となるような x_m のみが a_i を決めることが分かる。このような x_m をサポートベクターといい, サポートベクターからなる集合 S から

$$b = y_s - \langle a, x_s \rangle$$

を求める。

3.2 線形分離不可能な場合

図 7 にあるようにデータを超平面で分離することが出来ない場合は, データを高次元の空間に写像して分離超平面を求める方法が考案されている。前節の図 7 のように点を与えられた場合, 平面上に与えられた点達は第 1 象限と第 3 象限のデータのグループと, 第 2 象限と第 4 象限のデータのグループに分けることができそうである。この平面上の点の座

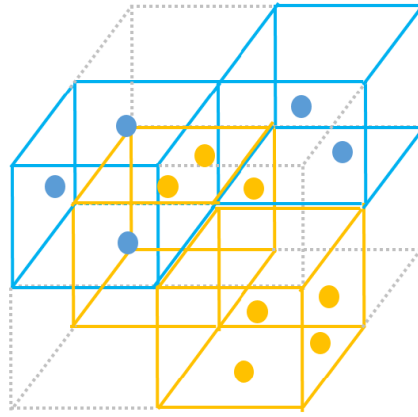


図 8 高次元空間への写像の例

標を (x_1, x_2) とおき，座標軸上の点を除けば

- 第 1 象限では $x_1 > 0, x_2 > 0$
- 第 2 象限では $x_1 < 0, x_2 > 0$
- 第 3 象限では $x_1 < 0, x_2 < 0$
- 第 4 象限では $x_1 > 0, x_2 < 0$

となることから，例えば

$$\begin{aligned}\varphi(x_1, x_2) &= (\varphi_1(x_1, x_2), \varphi_2(x_1, x_2), \varphi_3(x_1, x_2)) \\ &= (x_1, x_2, x_1 x_2)\end{aligned}$$

と定義すればデータは図 8 のような $\mathbf{R}^3$ の空間に写像される。一般の場合は，上述の例のように具体的な写像 φ を与えることは容易ではないが，線形分離可能でない場合でも，高次元への写像 φ を用いて線形分離可能な空間で SVM によるデータ分類を行うことができる。この場合，分離超平面の方程式は

$$f(\mathbf{x}) = a_i \varphi_i(\mathbf{x}) + b$$

で与えられ，線形分離可能な場合と同様にラグランジュの未定乗数法を適用すると，

$$F(\lambda) = \sum_{m=1}^M \lambda_m - \frac{1}{2} \sum_{m_1, m_2=1}^M \lambda_{m_1} \lambda_{m_2} y_{m_1} y_{m_2} \langle \varphi(\mathbf{x}_{m_1}), \varphi(\mathbf{x}_{m_2}) \rangle$$

を

$$\begin{aligned}\sum_{m=1}^M \lambda_m y_m &= 0 \\ \lambda &\geq 0\end{aligned}$$

の条件の下で最大化する $\lambda_1, \dots, \lambda_M$ を求める問題となる。いま，

$$K(x_{m1}, x_{m2}) = \langle \varphi(x_{m1}), \varphi(x_{m2}) \rangle = \sum_{m=1}^M \varphi_m(x_{m1}) \varphi_m(x_{m2})$$

とおくと,

$$a_i = \sum_{m=1}^M \lambda_m y_m \varphi(x_{im})$$

となることから,

$$\begin{aligned} \langle a, \varphi(x) \rangle &= \langle \sum_{m=1}^M \lambda_m y_m \varphi(x_m), \varphi(x) \rangle \\ &= \sum_{m=1}^M \lambda_m y_m K(x_m, x) \end{aligned}$$

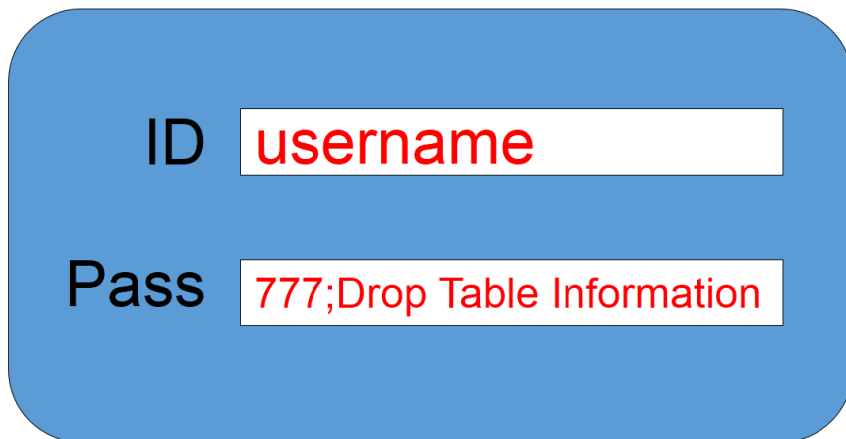
とできることから, 分離超平面の方程式は

$$\begin{aligned} f(x) &= \sum_{i=1}^n a_i x_i + b \\ &= \sum_{m=1}^M \lambda_m y_m K(x_m, x) + b \end{aligned}$$

と表されることが分かる。あとは, 線形分離可能な場合と同じである。この手法は

$$k(x_{m1}, x_{m2}) = \langle \varphi(x_{m1}), \varphi(x_{m2}) \rangle$$

を満足する関数が存在するときに有効な手法であり, このような関数はカーネル関数と呼ばれている。



The image shows a blue rounded rectangle representing a login form. It contains two input fields. The first field is labeled 'ID' and has the text 'username' entered. The second field is labeled 'Pass' and has the text '777;Drop Table Information' entered.

図 9 フォームからの入力

4 SQL インジェクション攻撃

本稿では、SVM を用いた SQL インジェクション攻撃を検出する方法を SVM の応用例として紹介する。SQL インジェクション攻撃とは脆弱性をもつデータベース駆動型の Web アプリケーションに対して、ブラウザのフォームなどから悪意のある SQL 文を入力し、Web アプリケーションがもつデータベースに不正にアクセスをする攻撃のことを指す。例えば、何らかのウェブ上に展開されている商用のサービスを利用すると、多くの場合図 9 に示すような入力フォームにユーザを特定するための情報、例えばユーザ ID やパスワードの入力が求められる。セミコロン (;) に対して脆弱性をもつ Web アプリケーションに、図 9 にあるような入力を実際にされた場合は SQL 文にセミコロン以下の命令

; Drop Table Information

がフォームへの入力に基づいて生成される SQL 文に追加され、データベースから Infomation という名前の表が削除されてしまう。SQL インジェクション攻撃では、脆弱性のある Web アプリケーションに対しては ID やパスワードを知らなくとも不正にデータベースにアクセス出来てしまったり、データベースにどのような情報があるかを知ることができてしまったりする。

SQL インジェクション攻撃は Web アプリケーションを導入する際にサニタイズやプリペアドステートメントといった技術的対策を行うことでこの攻撃の被害に遭遇する可能性は低くなると言われている。しかしながら、実際には SQL インジェクション攻撃の被害報告数は増加傾向にあり、Web アプリケーションを SQL インジェクション攻撃から防御する手段として過去に観測された実際の攻撃のパターンをブラックリスト化したり、パターン認識や SVM のような機械学習と呼ばれる手法を用いて攻撃を検出する方法[16][17]などが提案されている。これらの攻撃検出手法に対して、著者らは SQL インジェクション攻撃の文字列に含まれるシングルクォートやセミコロンなどの記号に着目して SQL インジェクション攻撃を特徴づける文字（以下、攻撃特徴文字という）を抽出し、攻撃特徴文字の文字列における含有率を用いて攻撃を検出するアルゴリズムを提案[18][19][20]し、提案アルゴリズムを確率モデルに適用した際の予測に関する理論的な解析[19]の他、非定常な確率分布を用いて攻撃検出を行った場合の考察などを行っている[21]。さらに、攻撃特徴文字の出現位置を加味した Web アプリケーション攻撃検出法[22][23]を考案し、実験によりその有効性を確認している。本稿では、SQL インジェクション攻撃の攻撃特徴文字によって構成されるユークリッド空間を考え、SVM がどのようにデータの分類を行うかということについて考察を行う。そのために、文献[18]で提案した SQL インジェクション攻撃の検出法を紹介する。

初めに、攻撃特徴文字の候補 s_a (a は自然数) を用意する。用意する文字は表 1 にあるようにセミコロンやシングルクォートなどの記号やスペースなどのものであり、文献[18]

では 20 個の文字から攻撃特徴文字の抽出を試みた。 l_i ($i = 1, 2, \dots, I$) を Web のフォームなどに入力された文字列, $|l_i|$ をその文字列長とし, l_i に含まれる文字を 1 文字ずつ調べ, l_i に文字 s_a が何回出現するかを数える。 l_i に文字 s_a が出現する回数を $|s_a|$ と定義する。いま,

$$x_{i,a} = \frac{|s_a|}{|l_i|}$$

と定義し, 0 から 1 までの値をとるある実数 α の下で $x_{i,a} \geq \alpha$ ならば l_i は攻撃文字列 (SQL インジェクション攻撃である文字列), $x_{i,a} < \alpha$ ならば l_i は正常文字列 (SQL インジェクション攻撃でない文字列) と判定することにする。これを判別関数

$$h(x_{i,a}) = \begin{cases} 1 & l_i \text{ は攻撃文字列 } (x_{i,a} \geq \alpha) \\ 0 & l_i \text{ は正常文字列 } (x_{i,a} < \alpha) \end{cases}$$

で表すことにする。文献[24]などから収集した $I_A = 624$ 個の攻撃文字列サンプルと, $I_N = 234$ 個の正常文字列サンプルを用いて

$$P_A(s_a, I_A) = \frac{1}{I_A} \sum_{i=1}^{I_A} h(x_{i,a})$$

$$P_N(s_a, I_N) = 1 - \frac{1}{I_N} \sum_{i=1}^{I_N} h(x_{i,a})$$

を計算したところ, 表 1 の結果が得られた。 $P_A = P_A(s_a, I_A)$ は攻撃文字列を攻撃と正しく

表 1 攻撃検出率 P_A と正常誤検出率 P_N

攻撃特徴文字候補	P_A	P_N
s_1 (スペース)	0.971	0.098
s_2 (セミコロン)	0.664	0.000
s_3 (シングルクォーテーション)	0.483	0.000
s_4 (左側丸括弧)	0.459	0.060
s_5 (右側丸括弧)	0.414	0.090
s_6 (左側中括弧)	0.000	0.060
s_7 (右側中括弧)	0.000	0.026
s_8 (左側大括弧)	0.000	0.026
s_9 (右側大括弧)	0.000	0.026
s_{10} (シャープ)	0.032	0.021
s_{11} (パーセント)	0.024	0.026
s_{12} (ダブルクォーテーション)	0.010	0.000
s_{13} (アンバサンド)	0.019	0.021
s_{14} (バックスラッシュ)	0.032	0.000
s_{15} (パイプ)	0.040	0.103
s_{16} (等号)	0.294	0.060
s_{17} (大なり記号)	0.000	0.090
s_{18} (小なり記号)	0.000	0.038
s_{19} (アスタリスク)	0.128	0.094
s_{20} (スラッシュ)	0.112	0.073

検出した割合（攻撃検出率）を、 $P_N = P_N(s_a, I_N)$ は正常文字列を正常と正しく検出した割合（正常検出率）を表す。文献[18]では攻撃特徴文字を複数組み合わせで攻撃検出することを検討し、その結果、攻撃特徴文字の組み合わせを上手く選ぶと以下に定義する検出率

$$\mu = \beta P_A + (1-\beta)P_N, (0 \leq \beta \leq 1)$$

の値を β の値によらずに大きくできる攻撃特徴文字の組み合わせ

スペース, セミコロン, ダブルクォート, 右側丸括弧

があることを実験で確認した。 μ は攻撃検出率と正常検出率の実数 β による加重平均である。文献[18]では上述の 4 文字を攻撃特徴文字として攻撃検出に利用しているが、文献[22][23]ではあらゆる文字を攻撃特徴文字とし、それらの文字に出現頻度や出現位置の情報から点数付けをおこなうことで攻撃特徴量を定義し、攻撃特徴量を用いて攻撃検出を行う方法を提案している。本稿では簡単のため、5 つの攻撃特徴文字

スペース, セミコロン, ダブルクォート, 右側丸括弧, 左側丸括弧

を攻撃特徴として SVM を用いて攻撃検出する枠組みを簡単な例を用いて紹介し、SVM の攻撃検出の様子を具体的に調べていく。

5 SVM を用いた SQL インジェクション攻撃の検出

この章では、前章で紹介した攻撃特徴文字から定義される数量を $\mathbf{R}^2$ 上の点で表現し、SVM を用いた SQL インジェクション攻撃検出に関する考察を行う。ここで用いる攻撃特徴文字はスペース, セミコロン, ダブルクォート, 右側丸括弧の 4 文字とし、文字列 l に含まれる攻撃特徴文字の総出現頻度を $|s|$ とする。このとき、文字列 l に対して $\mathbf{R}^2$ 上の点 $(x, y) = (|s|, |l| - |s|)$ を定義することができる。 $\mathbf{R}^2$ の点に攻撃文字列と正常文字列のラベル z をつけるために集合 $\{-1, 1\}$ を考え、攻撃文字列にはラベル 1 を、正常文字列にはラベル -1 を対応させることで、直積集合 $\mathbf{R}^2 \times \{0, 1\}$ が定義される。この直積集合の座標を (x, y, z) で表す。以上の設定の下で、求める分離超平面 H の方程式は $ax + by + c = 0$ とできることが分かる。

まず 1 つの攻撃サンプルと 1 つの正常サンプルから分離超平面を求める。

Example 7. 攻撃サンプル l_1 として

`';SELECT * FROM members WHERE username = 'admin'-' AND,`

正常サンプル l_2 として

01-2345-6789

を考える。このとき、 l_1 は点 $P_1 = (x_1, y_1, z_1) = (6, 71, 1)$ に、 l_2 は点 $P_2 = (x_2, y_2, z_2) = (0, 12, -1)$ に対応する。 H から点 P_i ($i = 1, 2$) までの距離 D は、定理 1 から

$$D = \frac{|ax_i + by_i + c|}{\sqrt{a^2 + b^2}}$$

となることが分かる。ここで、点 P_1, P_2 は明らかにサポートベクターの上にあるので、 $|ax_i + by_i + c| = 1$ であり、制約条件 $z_i\{ax_i + by_i + c - 1\} \geq 0$ のもとで $\frac{1}{\sqrt{a^2 + b^2}}$ を最小にする a, b を求めることで超平面 H が求められることが分かる。このような a, b の候補はラグランジュの未定乗数法により

$$L(a, b, \lambda_1, \lambda_2) = \frac{1}{2}(a^2 + b^2) - \lambda_1\{z_1(ax_1 + by_1 + c) - 1\} - \lambda_2\{z_2(ax_2 + by_2 + c) - 1\}$$

を最大とする a, b を求めることで得ることができる。

$$\begin{aligned}\frac{\partial L}{\partial a} &= a - (\lambda_1 z_1 x_1 + \lambda_2 z_2 x_2) \\ \frac{\partial L}{\partial b} &= b - (\lambda_1 z_1 y_1 + \lambda_2 z_2 y_2) \\ \frac{\partial L}{\partial c} &= c - (\lambda_1 z_1 x_1 + \lambda_2 z_2 x_2)\end{aligned}$$

から、 $\frac{\partial L}{\partial a} = \frac{\partial L}{\partial b} = \frac{\partial L}{\partial c}$ を解いて、

$$L(\lambda_1) = -\frac{1}{2}\lambda_1^2 z_1^2 (x_1 - x_2)^2 - \frac{1}{2}\lambda_1^2 z_1^2 (y_1 - y_2)^2 + 2\lambda_1$$

が得られる。この式に点 P_1, P_2 の値を代入すると、

$$L(\lambda_1) = -\frac{3517}{2}\left(\lambda_1 - \frac{2}{3517}\right)^2 + \frac{2}{3517}$$

となり、 $L(\lambda_1)$ は $\lambda_1 = \frac{2}{3517}$ で最大となる。 $\frac{\partial L}{\partial c} = 0$ から $\lambda_1 = \lambda_2 = \frac{2}{3517}$ が得られ、 $\frac{\partial L}{\partial a} = 0, \frac{\partial L}{\partial b} = 0$ から、

$$\begin{aligned}a &= \frac{12}{3517} \\ b &= \frac{108}{3517}\end{aligned}$$

が超平面 H からサポートベクターまでの距離 D を最大とする a, b であることが分かる。求められた λ_1, λ_2 は制約条件 $\lambda_1 \geq 0, \lambda_2 \geq 0$ を満足することに注意する。超平面の方程式を定めるにはあとは c の値を求めるだけである。 c を求めるにはサポートベクターである P_1, P_2 のいずれかの情報を用いればよい。例えば、 $P_2 = (0, 12, -1)$ を用いれば、超平面の方程式

$$\frac{12}{3517}x + \frac{108}{3517}y + c = -1$$

に $x = 0, y = 12$ を代入して, $c = -\frac{4813}{3517}$ を得る。以上の計算のもとで, 分離超平面 H の方程式は

$$\frac{12}{3517}x + \frac{108}{3517}y - \frac{4813}{3517} = 0$$

と求められることが分かる。

Example 7 では, 距離 D を最大化する a, b を求める際に $L(\lambda_1)$ の式の平方完成を用いたが, サンプルの数が多くなると平方完成でこれらの計算を行うのはかなり大変になる。そこで, 最急降下法と呼ばれる数値計算を用いて $L(\lambda_1)$ を最大にする λ_1 を求める方法が広く利用されている。最急降下法を C 言語でコーディングすると図 10 のようになる。

```
#include <stdio.h>
#include <stdlib.h>

double f(double a) {
    return(a*(a-4.0/3517.0));
}

double df(double a) {
    return(a-2.0/3517.0);
}

main() {
    double a;
    double alpha = 0.1;

    a = 0.0;
    int i;

    for (i = 1; i < 85; i++) {
        a = a - alpha * df(a);
        printf("Value of a at Step %d is %f, ", i, a);
    }
}
```

図 10 最急降下法

プログラム中の a が Example 7 の λ_1 に対応することに注意する。この場合の実行結果は以下ようになる。

```
Value of a at Step 1 is 0.000057
Value of a at Step 2 is 0.000108
Value of a at Step 3 is 0.000154
Value of a at Step 4 is 0.000196
Value of a at Step 5 is 0.000233
```

Value of a at Step 6 is 0.000266
Value of a at Step 7 is 0.000297
Value of a at Step 8 is 0.000324
Value of a at Step 9 is 0.000348
Value of a at Step 10 is 0.000370
Value of a at Step 11 is 0.000390
Value of a at Step 12 is 0.000408
Value of a at Step 13 is 0.000424
Value of a at Step 14 is 0.000439
Value of a at Step 15 is 0.000452
Value of a at Step 16 is 0.000463
Value of a at Step 17 is 0.000474
Value of a at Step 18 is 0.000483
Value of a at Step 19 is 0.000492
Value of a at Step 20 is 0.000500
Value of a at Step 21 is 0.000506
Value of a at Step 22 is 0.000513
Value of a at Step 23 is 0.000518
Value of a at Step 24 is 0.000523
Value of a at Step 25 is 0.000528
Value of a at Step 26 is 0.000532
Value of a at Step 27 is 0.000536
Value of a at Step 28 is 0.000539
Value of a at Step 29 is 0.000542
Value of a at Step 30 is 0.000545
Value of a at Step 31 is 0.000547
Value of a at Step 32 is 0.000549
Value of a at Step 33 is 0.000551
Value of a at Step 34 is 0.000553
Value of a at Step 35 is 0.000554
Value of a at Step 36 is 0.000556
Value of a at Step 37 is 0.000557
Value of a at Step 38 is 0.000558
Value of a at Step 39 is 0.000559
Value of a at Step 40 is 0.000560
Value of a at Step 41 is 0.000561
Value of a at Step 42 is 0.000562

Value of a at Step 43 is 0.000563
Value of a at Step 44 is 0.000563
Value of a at Step 45 is 0.000564
Value of a at Step 46 is 0.000564
Value of a at Step 47 is 0.000565
Value of a at Step 48 is 0.000565
Value of a at Step 49 is 0.000565
Value of a at Step 50 is 0.000566
Value of a at Step 51 is 0.000566
Value of a at Step 52 is 0.000566
Value of a at Step 53 is 0.000567
Value of a at Step 54 is 0.000567
Value of a at Step 55 is 0.000567
Value of a at Step 56 is 0.000567
Value of a at Step 57 is 0.000567
Value of a at Step 58 is 0.000567
Value of a at Step 59 is 0.000568
Value of a at Step 60 is 0.000568
Value of a at Step 61 is 0.000568
Value of a at Step 62 is 0.000568
Value of a at Step 63 is 0.000568
Value of a at Step 64 is 0.000568
Value of a at Step 65 is 0.000568
Value of a at Step 66 is 0.000568
Value of a at Step 67 is 0.000568
Value of a at Step 68 is 0.000568
Value of a at Step 69 is 0.000568
Value of a at Step 70 is 0.000568
Value of a at Step 71 is 0.000568
Value of a at Step 72 is 0.000568
Value of a at Step 73 is 0.000568
Value of a at Step 74 is 0.000568
Value of a at Step 75 is 0.000568
Value of a at Step 76 is 0.000568
Value of a at Step 77 is 0.000568
Value of a at Step 78 is 0.000569
Value of a at Step 79 is 0.000569

Value of a at Step 80 is 0.000569

Value of a at Step 81 is 0.000569

Value of a at Step 82 is 0.000569

Value of a at Step 83 is 0.000569

Value of a at Step 84 is 0.000569

この実行結果からわかるように, $\lambda_1 = \frac{2}{3517} = 0.000569$ であることから Step78 で最急降下法が収束していることが分かる。図 10 で示したプログラムでは

$$L(\lambda_1) = -\frac{3517}{2} \left(\lambda_1 - \frac{2}{3517} \right)^2 + \frac{2}{3517}$$

の最大化を, プログラミング中の関数の表記では $f(a) = a(a - \frac{4}{3517})$ の最小値を求める

方法で実現している。こうせずに $L(\lambda_1)$ の形のままで計算させると $\lambda_1 = \frac{4}{3517} = 0.000569$

に収束しないことが確かめられる。なお, ラグランジュの未定乗で求めた分離超平面

$$\frac{12}{3517}x + \frac{108}{3517}y - \frac{4813}{3517} = 0$$

は線分 P_1P_2 の垂直二等分線となることが理想であると考えられるが, そうなるとは限らない。また, 点 P_1 もこの超平面上の点でないことも容易に確かめられる。しかしながら, 分離超平面と線分 P_1P_2 は垂直に交わっており, 分離超平面は 2 点 P_1, P_2 の中点の近くを通っている。

このようなことが起こることは以下のような簡単な例を考えることで容易に分かる。

Example 8. 2つのグループ C_1, C_2 に分けることができる $\mathbf{R}^2$ の 2つの点 $A = (\frac{3}{2}, \frac{3}{2})$, $B = (\frac{1}{2}, \frac{1}{2})$ があるとする。 $A \in C_1, B \in C_2, (x_1, y_1, z_1) = (\frac{3}{2}, \frac{3}{2}, 1), (x_2, y_2, z_2) = (\frac{1}{2}, \frac{1}{2}, -1)$

とした場合の分離超平面の方程式は $x + y - 2 = 0$ であり, 点 A を通る超平面の方程式は $x + y - 2 = 1$, 点 B を通る超平面の方程式は $x + y - 2 = -1$ となっている。しかしながら, $C = (\frac{3}{2}, \frac{5}{4}) \in C_1, B = (\frac{1}{2}, \frac{1}{2}) \in C_2$ の分離超平面 $ax + by + c = 0$ の係数 c を求めるには, 以下の 3 パターンが考えられる。

[パターン 1]

$ax + by + c = 1$ に $(x, y) = (\frac{3}{2}, \frac{5}{4})$ を代入する

[パターン 2]

$ax + by + c = 1$ に $(x, y) = (\frac{1}{2}, \frac{1}{2})$ を代入する

[パターン 3]

$ax + by + c = 1$ に $(x, y) = (1, \frac{7}{8})$ を代入する

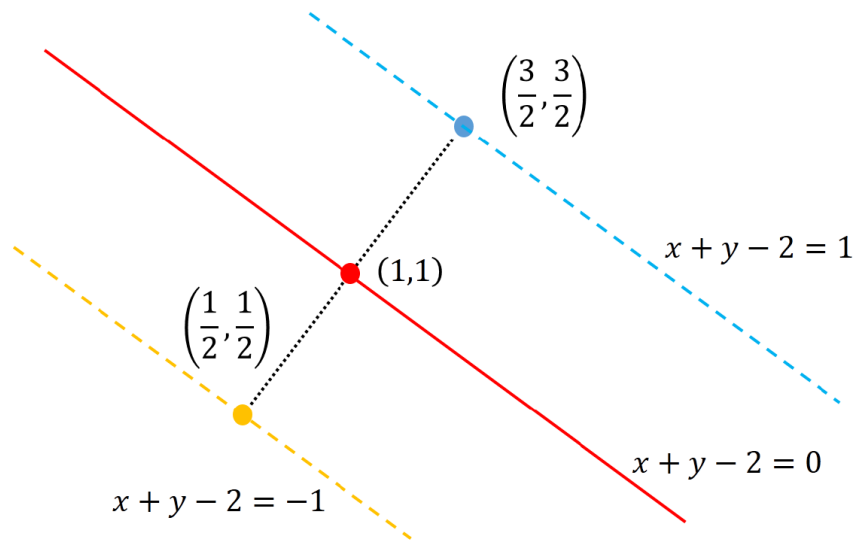


図 11 超平面 $x + y - 2 = 0$ による分類

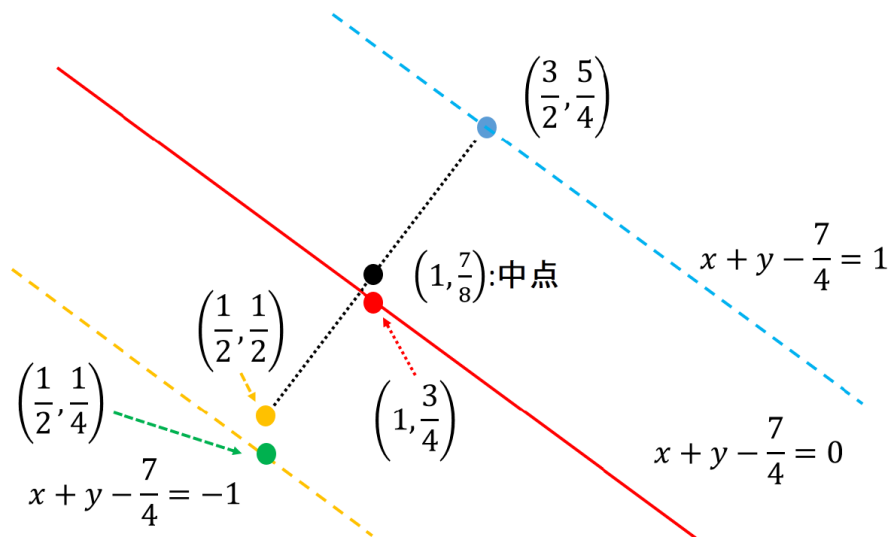


図 12 点 C で分離超平面を定めた場合

[パターン 3] は 2 点 C, D の中点である。 a, b の値はラグランジュの未定乗数法から $a = b = 1$ と定まる。パターン 1 で分離平面を計算すると図 12 のようになる。

Example 8 の例のように $\mathbf{R}^2$ におけるサンプルが 2 つの場合であっても、分離超平面が垂直 2 等分線とならないパターンはいくらでも作ることができる。

次にサンプルが 3 つの場合について考える。

Example 9. Example 7 で利用したサンプルに別の攻撃サンプル l_3

ID: 10; DROP TABLE members /*

を考える。 l_3 は $\mathbf{R}^2$ の点 $P_3 = (x_3, y_3, z_3) = (6, 29, 1)$ に対応することになる。このとき分離超平面の方程式 $ax + by + c = 0$ の係数 a, b はラグランジュの未定乗数法から、

$$L(a, b, c, \lambda_1, \lambda_2, \lambda_3) = \frac{1}{2} (a^2 + b^2) - \sum_{i=1}^3 \lambda_i \{z_i(ax_i + by_i + c) - 1\}$$

を最大にする a, b から求められる。 $\frac{\partial L}{\partial a} = \frac{\partial L}{\partial b} = \frac{\partial L}{\partial c} = 0$ を計算して、

$$L(\lambda_1, \lambda_2) = -\frac{1}{2} \{\lambda_1 z_1 (x_1 - x_3) + \lambda_2 z_2 (x_2 - x_3)\}^2 - \frac{1}{2} \{\lambda_1 z_1 (y_1 - y_3) + \lambda_2 z_2 (y_2 - y_3)\}^2 \\ + \lambda_1 \left(1 - \frac{z_1}{z_3}\right) + \lambda_2 \left(1 - \frac{z_2}{z_3}\right)$$

と変形することができる。ただし、 $\lambda_1 \geq 0, \lambda_2 \geq 0$ である。点 P_1, P_2, P_3 の値を代入すれば、

$$L(\lambda_1, \lambda_2) = -\frac{3517}{2} \lambda_1^2 - 1039 \lambda_1 \lambda_2 - \frac{325}{2} \lambda_2^2 + 2\lambda_1 + 2\lambda_2$$

となる。この式を平方完成すれば、

$$L(\lambda_1, \lambda_2) = -\frac{3517}{2} \left\{ \lambda_1 + \left(\frac{1039}{3517} \lambda_2 - \frac{2}{3517} \right) \right\}^2 - \frac{31752}{3517} \left(\lambda_2 + \frac{3517}{63504} \right)^2 + \frac{12623305}{46687136}$$

となるが、 $L(\lambda_1, \lambda_2)$ を最大にする $\lambda_1 = \frac{3781171}{223343568}, \lambda_2 = -\frac{3517}{63504}$ は制約条件 $\lambda_1, \lambda_2 \geq 0$ を満たさない。ここで、例えば点 P_3 を取り除いて分離超平面を求めると Example 7 の方程式が得られる。

文献[18]で著者らは、文字列に含まれる攻撃特徴率の含有率を用いて SQL インジェクション攻撃を検出するというシンプルな攻撃検出法提案している。この方法は以下のような枠組みで SVM を適用すれば、文献[18]で提案した方法と同じような意味をもつことになる。

Example 10. この例では、文字列 l に対応させる $\mathbf{R}^2$ の点を $(x, y) = \left(\frac{|s|}{|l|}, 1 - \frac{|s|}{|l|} \right)$ と定義する。すると $x + y = 1$ であるから、文字列 l に対応する点は直線 $x + y - 1 = 0$ 上に並ぶことがわかる。これから、Example 7, 9 で利用した攻撃サンプル l_1, l_3 , 正常サンプル l_2 に SVM を適用して、攻撃検出を行うための分離超平面の計算をする。文字列 l_1, l_2, l_3 から $P_1 = (x_1, y_1, z_1) = \left(\frac{6}{77}, \frac{71}{77}, 1 \right), P_2 = (x_2, y_2, z_2) = (0, 1, -1), P_3 = (x_3, y_3, z_3) = \left(\frac{6}{35}, \frac{29}{35}, 1 \right)$ が定まる。この場合は

$$L(a, b, c, \lambda_1, \lambda_2, \lambda_3) = \frac{1}{2} (a^2 + b^2) - \sum_{i=1}^3 \lambda_i \{z_i(ax_i + by_i + c) - 1\}$$

を最大にする a, b を、

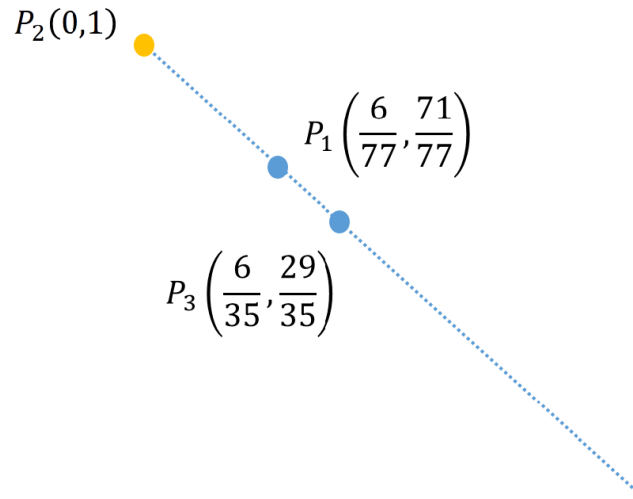


図 13 含有率で分離超平面を定めた場合

$$L(\lambda_1, \lambda_2) = -\frac{36}{1225} \left(\lambda_2 - \frac{1225}{36} \left(1 - \frac{1512}{94325} \lambda_1 \right) \right)^2 - \frac{8820}{8895} \lambda_1 + \frac{1225}{36}$$

を最大にする λ_1, λ_2 から求めることになる。実際, $\lambda_1 = 0, \lambda_2 = \frac{1225}{36}$ のとき $L(\lambda_1, \lambda_2)$ は最大となり, $a = \frac{36}{5}, b = -\frac{36}{5}$ を得る。このとき, 分離超平面は $\mathbf{R}^2$ における傾きが 1 の直線であり, サンプルが傾き -1 の直線上にあることから分離超平面の方程式として適切な計算結果になっているものといえる。係数 c は点 P_2 の座標を代入するなどして決めればよい。

本稿ではカーネル関数を用いた議論はしないが, そのような関数を用いる意義に関する考察を行うため最後に以下の例を紹介する。

Example 11. 2つの攻撃サンプル

```
;DR|**|OP|*bypass blacklist*|sampletable
SELECT CONCAT(CHAR(75),CHAR(76),CHAR(77))
```

と 1 つの正常サンプル

```
(userID*
```

について考える。これらの文字列を上から順に l_1, l_2, l_3 とし, それぞれ $P_1 = \left(\frac{1}{40}, \frac{39}{40}, 1 \right), P_2 = \left(\frac{1}{5}, \frac{4}{5}, 1 \right), P_3 = \left(\frac{1}{8}, \frac{7}{8}, 1 \right)$ と定義する。(P_3 の座標は厳密には $\left(\frac{5}{40}, \frac{36}{40} \right)$ となる) このとき, 3 点 P_1, P_2, P_3 は直線 $y = -x + 1$ 上にあり, この例を用いた場合は線形分離不

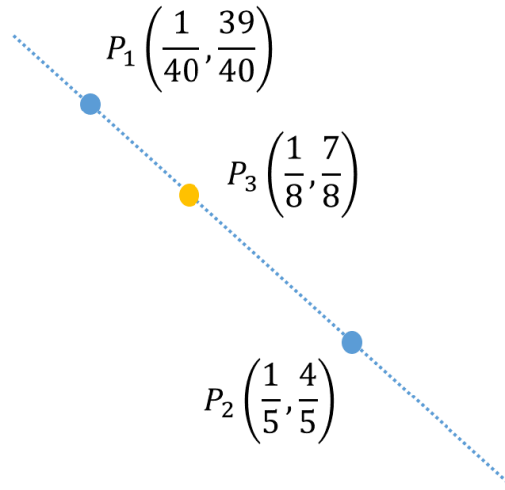


図 14 含有率で分離超平面を定めた場合 2

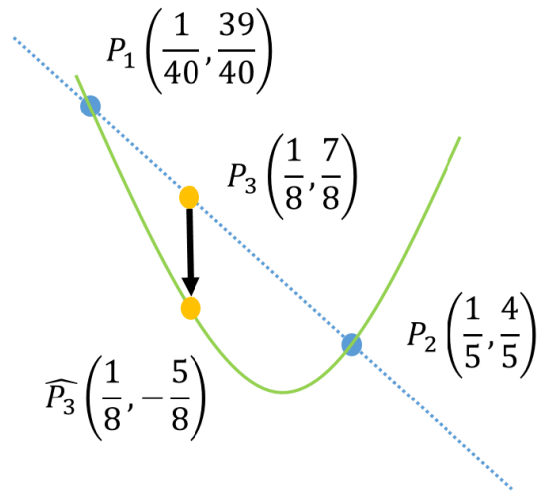


図 15 放物線上の射影によるデータの分離

可能であることが分かる。いま図 15 のように 2 つの点 P_1, P_3 を通る放物線 $y = k_1x^2 + k_2x + k_3$ を考えるとその方程式は

$$k_1x^2 + k_2x + k_3 - (-x + 1) = k_1\left(x - \frac{1}{40}\right)\left(x - \frac{1}{5}\right)$$

となり, k_2, k_3 が整数値となるように $k_1 = 200$ とおけば, $k_2 = -46, k_3 = 2$ が得られる。これを用いて, 変数変換

$$\varphi(x, y) = (x, 200x^2 - 46x + 2)$$

を考えれば, 点 P_1, P_2 の座標に変化はないが, 点 P_3 は

$$\hat{P}_3 = \left(\frac{1}{8}, -\frac{5}{8} \right)$$

に移動し、これで線形分離可能な状態にできることがわかる。すると、Example 9 と同様にして分離超平面を求めることができるが、Example 9 の例で生じた現象が起こる可能性も考えられる。

6 考察とまとめ

本稿では SVM を理解するために必要な数学と、SVM を用いた SQL インジェクション攻撃の検出に関する考察を行った。SVM では分離超平面 $\left\{ \sum_{i=1}^n a_i x_i + b = 0 \right\}$ のパラメータ $a_1, a_2, \dots, a_n, b$ をサンプルから定めるために $a_1, a_2, \dots, a_n, b$ で定義される関数を何らかの制約条件の下で最大化する問題を扱うことを紹介した。その際に、ラグランジュの未定乗数法を利用すれば変数 $\lambda_1, \lambda_2, \dots, \lambda_m$ で定義される関数 L をある制約条件の下で最大化することを考えるが、関数 L は変数 $\lambda_1, \lambda_2, \dots, \lambda_m$ たちの 2 次の次数で定義される関数になるため、必ず与えられた制約条件のもとで最大値をもつことは分かる。5 章で紹介した例では、分離超平面を求めるために利用したサンプルが高々 3 つであるため、関数 L の変数も高々 3 つであり、しかも制約条件の式から関数 L の変数を高々 1 つにすることができた。そのため、関数 L の方程式がそれほど複雑にならないため、数値計算を必要とせずに分離超平面を求めるパラメータを求めることができた。しかしながら、SVM を用いてデータの分類を行う場合、通常は多くのサンプルを用いて分離超平面を決定するため数値計算に頼らざるを得ないことになる。その場合、Example 7 のあとで紹介した最急降下法の問題と直面することになると考えられる。また、Example 9 で紹介したような現象もしばしば起こるものと推察される。Example 9 の内容を一般化することで、どのような点を除外すれば制約条件を満たすことができるかを求めることができると考えられる。本稿では、SQL インジェクション攻撃の特徴を攻撃文字で行い、文字列に含まれる攻撃文字数で定義される空間を定義して SVM を適用し、攻撃検出を行うためのルールづくりを行った。他の従来研究の SVM を用いた SQL インジェクション攻撃を自動検出する手法は、SQL インジェクション攻撃の特徴を他のものを用いて行っている。SVM を用いて SQL インジェクション攻撃を検出するためには、SQL インジェクション攻撃を何らかの数に対応させる必要がある。本稿では、攻撃特徴文字の文字列における出現回数に着目して SVM に適用したが、この場合は以下の問題が起きることになる。例えば、5 章の定義に基づいて 10 個の文字からなる文字列に 2 個の攻撃特徴文字が含まれていたとする。このとき、この文字列は $P = (2, 8)$ または $P = \left(\frac{1}{5}, \frac{4}{5} \right)$ のように $\mathbf{R}^2$ の点に対応することになるが、SQL インジェクション攻撃に関してはこの点に対応する攻撃文字列や正常文字列がいくらかでも作成可能である。これと同様な現象は、他の特徴文字を用いても同様である。文字列を数値化する作業は、文字列から何らかの数の空間への写像を考えることと同値であり、この写像を限りなく一対一対応の写像になるように SQL インジェクション攻撃の特

微抽出の方法を考えることが今後の重大な課題であるといえる。

参考文献

- [1] C. Corinna and V. Vladimir: *Support-Vector Networks*, Machine Learning, 20, (1995).
- [2] C.J. Malgorzata: *Performance of the Support Vector Machines for Medical Classification Problems*, Biocybernetics and Biomedical Engineering 2009, Volume 29, Number 4, pp. 63-81, (2009).
- [3] I. Zyout and I. Abdel-Qader: *An Improvement of Texture-Based Classification of Microcalcification Clusters in Mammography using PSO-SVM Approach*, Proceedings in the 5th International Conference on Communications, Computers and Applications, MIC-CCA2012, pp. 1-5, (2012).
- [4] Y. Guo, R. Gaizauskas, I. Roberts, G. Demetriou and M. Hepple: *Identifying Personal Health Information Using Support Vector Machines*, In Proceedings of the AMIA 2006 Workshop on Challenges in Natural Language Processing for Clinical, (2006).
- [5] M. S. Kumar: *An improved support vector machine kernel for medical image retrieval system*, 2012 International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME), pp. 257-260, (2012).
- [6] 福村肇, 岡田志麻, 牧川方昭: 非接触計測した睡眠時の額部・鼻部皮膚温と SVM を用いた睡眠段階の推定, 生体医工学 50(1), 131-137, (2012).
- [7] 篠原孝幸, 岩堀祐之, 福井真二, 春日井邦夫: ベイジアンフィルタを用いた内視鏡画像からのポリープ自動検出, 電子情報通信学会技術研究報告, MI, 医用画像 111 (389), 305-310, (2012).
- [8] 佐野俊文, 松崎周一, 和田安弘: *fNIRS* 信号からの神経情報デコーディングの可能性の検討: ヒト腕等尺性力方向の推定, 電子情報通信学会論文誌. D, 情報・システム J94-D(10), 1676-1684, (2011).
- [9] 青木謙二, 末吉健二, 石橋太郎, 坂本真人, 古谷博史: サポートベクターマシンを用いた血友病 B データベースの解析——アミノ酸置換の第 IX 因子活性への影響——, 情報処理学会研究報告. MPS, 数理モデル化と問題解決研究報告 2011-MPS-85(7), 1-8, (2011).
- [10] 沼杏子, 屋代智之: SVM を用いた歩行者の状態推定手法の提案, 情報処理学会研究報告. MBL, [モバイルコンピューティングとユビキタス通信研究会研究報告] 2011-MBL-60(3), 1-6, (2011).
- [11] 松木洋, 藤本康孝: SVM および PCA を用いた運転ノウハウの抽出による上水送水系の運転支援システム, 電気学会論文誌. D, 産業応用部門誌 = IEEJ transactions on industry applications 132(10), 990-996, (2012).
- [12] 森和也, 津高健太郎, 和田友孝 [他]: パニック型災害における非常時緊急救命避難支援システム (ERESS) の開発: バッファリング SVM による災害検知アルゴリズム, 電子情報通信学会技術研究報告: 信学技報 111(469), 127-132, (2012).
- [13] 藤井雄太郎, 吉村卓也, 伊藤孝行: ベイズ分類器による文書の有害確率を素性に用いた SVM による有害文書分類手法, 電子情報通信学会技術研究報告. AI, 人工知能と知識処理 111(447), 41-46, (2012).
- [14] 櫻田玲子, 近藤悟, 森谷高明, 赤埴淳一: HTTP フローからの Web サイトへの情報発信型書込み行為判別手法, 電子情報通信学会論文誌. B, 通信 J95-B(4), 523-533, (2012).
- [15] 池田和史, 柳原正, 服部元, 松本一則, 小野智弘, 滝嶋康弘: HTML 要素に基づく有害サイト検出手法, 情報処理学会論文誌 52(8), 2474-2483, (2011).
- [16] R. Rawat, S. K. Shrivastav: *SQL injection attack Detection using SVM*, International Journal of Computer Applications (0975-8887), Volume 42, No. 13, pp. 1-4, (2012).
- [17] R. C. Chen: *Using Rough Set and Support Vector Machine for Network Intrusion Detection System*, First Asian Conference on Intelligent Information and Database Systems, ACIIDS, pp. 465-470, (2009).

- [18] M. Sonoda, T. Matsuda, D. Koizumi and S. Hirasawa,; *On the Automatic Detection of the SQL Injection Attacks by the Feature Extraction of the Single Character*, Proceeding of 4th International Conference on Security Information and Networks, pp. 81-86, ACM, (2011).
- [19] T. Matsuda, D. Koizumi, M. Sonoda and S. Hirasawa, *On Predictive Errors of SQL Injection Attack Detection by the Feature of the Single Character*, Proceeding in 2011 IEEE International Conference on Systems, Man and Cybernetics, pp. 1722-1727, (2011).
- [20] D. Koizumi, T. Matsuda and M. Sonoda, *A Learning Algorithm of Threshold Value on the Automatic Detection of SQL Injection Attack*, The 2012 International Conference on Parallel and Distributed Processing Techniques and Applications, Vol. 2012-MPS-89 No. 10, pp. 1-6 (2012).
- [21] D. Koizumi, T. Matsuda and M. Sonoda, *On the Automatic Detection Algorithm of Cross Site Scripting (XSS) with the Non-Stationary Bernoulli Distribution*, Proceedings in the 5th International Conference on Communications, Computers and Applications, pp. 101-105, (2012).
- [22] 井上繁之, 松田健, 関連単語抽出アルゴリズムによる SQL インジェクション攻撃特徴抽出について, 情報処理学会研究報告. MPS, 数理モデル化と問題解決研究報告 2012-MPS-87 (30), pp. 1-2 (2012).
- [23] T. Matsuda, D. Koizumi, M. Sonoda, *Cross Site Scripting Attacks Detection Algorithm Based on the Appearance Position of Characters*, Proceedings in the 5th International Conference on Communications, Computers and Applications, pp. 101-105, (2012).
- [24] J. Clarke,; *SQL Injection Attacks And Defense*, Syngress Publishing Inc., 2009.

Support Vector Machine and Its Application to Web Application Attacks

Takeshi Matsuda

Support vector machine (SVM) is well known as the binary classification method which realizes the efficient classification from the theoretical standpoint. In this paper, we introduce the outline of SVM, and the application for the detection of web application attacks.

Keywords; SVM, Web application attacks