FH Vorarlberg
University of Applied Sciences

Support Vector Machines (SVM)

Computational Intelligence II

Informatik - Software and Information Engineering
Fachhochschule Vorarlberg

Erstellt von
André Hopfgartner & Matthias Rupp

Dornbirn, am 5. März 2021

Inhaltsverzeichnis

Abkürzungsverzeichnis

SVM Support Vector Machine

1 Einführung

1.1 Intuition

Ziel: möglichst breites Band zwischen den 2 verschiedenen Klassen aufziehen.

1.2 Mathematische Herleitung

TODO TEXT HERE

1.2.1 Problemdefinition

Gegeben sei ein Gewichtsvektor $w \in \mathbb{R}^D$, ein Bias $b \in \mathbb{R}$ und ein beliebiger Punkt $x \in \mathbb{R}^D$. Eine Ebene im Raum kann definiert werden durch:

$$w^T x + b = 0 \tag{1.1}$$

Weil Gleichung 1.1 mit verschiedenen Skalarwerten skaliert werden kann, führen wir eine zusätzliche Bedingung ohne Beschränkung der Allgemeinheit ein. Sei $\hat{x} \in \mathbb{R}^D$ der am nächsten zur Ebene gelegene Punkt so soll gelten:

$$|w^T \hat{x} + b| = 1 \tag{1.2}$$

Als nächsten Schritt bestimmen wir den euklidischen Normalabstand D eines beliebigen Punkts $x_k \in \mathbb{R}^D$ zu der Ebene. Hierfür ist zuerst zu bemerken, dass w normal zur definierten Ebene steht.

Lemma 1.2.1. *Eine Ebene sei definiert durch $w^T x + b = 0$. Der Vektor w steht normal zu der definierten Ebene.*

Beweis. Man wähle zwei Punkte $x_1, x_2 \in \mathbb{R}^D$ die auf der Ebene liegen. Somit muss gelten:

$$\begin{aligned} w^T x_1 + b &= 0 \\ w^T x_2 + b &= 0 \\ w^T (x_1 - x_2) = 0 \leftrightarrow \|w^T\| \|x_1 - x_2\| \cos(\alpha) &= 0 \leftrightarrow \alpha = 90^\circ \end{aligned} \tag{1.3}$$

□

Um den Normalabstand D eines beliebigen Punkts x_k zu ermitteln wählt man einen Punkt x der auf der Ebene liegt und projiziert den Vektor $(x_k - x)$ auf den

Einheitsvektor von w. Weil nur der tatsächliche Abstand zur Ebene relevant ist nimmt man den Betrag.

$$\begin{aligned} D &= |\frac{w^T}{\|w\|}(x_k - x)| = \\ &= \frac{1}{\|w\|}|(w^T x_k - w^T x)| = \\ &= \frac{1}{\|w\|}|(w^T x_k + b - (w^T x + b))| \end{aligned} \tag{1.4}$$

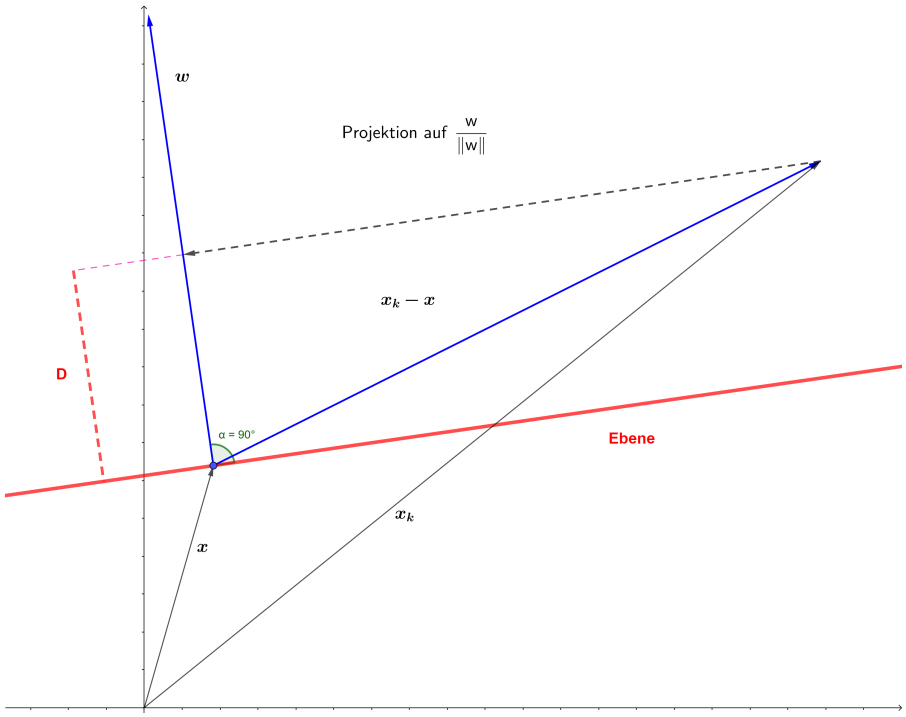

Abbildung 1.1: Durch die Projektion von $(x_k - x)$ auf den Einheitsvektor von w kann der Normalabstand D von x_k zu der Ebene bestimmt werden.

Weil der Punkt x auf der Ebene liegt gilt $w^T x + b = 0$ (Gleichung 1.1):

$$D = \frac{1}{\|w\|}|(w^T x_k + b)| \tag{1.5}$$

Aus Gleichung 1.2 gilt weiters $|(w^T x_k + b)| = 1$ wenn $x_k = \hat{x}$ der am nächsten zu der Ebene liegende Punkt ist. Somit ergibt sich der kleinste Abstand zur Ebene als:

$$D = \frac{1}{\|w\|} \tag{1.6}$$

1.2.2 Optimierungsproblem

Gleichung 1.6 beschreibt den Normalabstand zu dem am nächsten an der Ebene liegenden Punkt $\hat{x}$. Ziel einer Support Vector Machine (SVM) ist die Maximierung dieses Abstands für alle N Eingabevektoren $x_1..x_N \in \mathbb{R}^D$. Hierbei handelt es sich um ein Optimierungsproblem mit Nebenbedingungen:

$$\max_{w} \quad \frac{1}{\|w\|} \tag{1.7a}$$

$$\text{mit} \quad \min_{n=1..N} |w^T x_n + b| = 1 \tag{1.7b}$$

Gleichung 1.7b beschreibt hier den am nächsten zur Ebene gelegenen Punkt $\hat{x}$ in allgemeiner Form. Der Betrag lässt sich umschreiben durch die Einführung von Labels $y_n \in \{-1, +1\}$. Diese beschreiben jeweils die Klasse des zugehörigen Vektors x_n. Für eine korrekte Klassifizierung der SVM gilt:

$$y_n = sign(w^T x_n + b) \tag{1.8}$$

Somit gilt für einen korrekt klassifizierten Vektor x_k:

$$|w^T x_n + b| = y_n(w^T x_n + b) \tag{1.9}$$

Durch Anwendung von Gleichung 1.9 in Gleichung 1.7b, Umformulierung der Maximierung in eine Minimierung und der Verallgemeinerung von $\hat{x}$ auf beliebige Punkte x_n erhält man:

$$\min_{w} \quad \frac{1}{2} w^T w \tag{1.10a}$$

$$\text{mit} \quad y_n(w^T x_n + b) \geq 1 \text{ für } n = 1..N \tag{1.10b}$$

Die Verallgemeinerung von Gleichung 1.7b auf Gleichung 1.10b auf beliebige Punkte ist so möglich, weil durch Gleichung 1.2 sichergestellt ist, dass der kleinste Wert für $(w^T x_n + b)$ 1 ist, und somit die Werte für alle anderen Punkte größer oder gleich 1 sein müssen.

1.2.3 Lagrange Optimierung

Das beschriebene Optimierungsproblem beinhaltet eine Ungleichung in Gleichung 1.10b. Diese Optimierung kann mittels des Karush–Kuhn–Tucker Ansatzes gelöst werden. Zuerst wird die Nebenbedingung umgeformt:

$$\min_{w} \quad \frac{1}{2} w^T w \tag{1.11a}$$

$$\text{mit} \quad y_n(w^T x_n + b) - 1 \geq 0 \text{ für } n = 1..N \tag{1.11b}$$

$y_n(w^T x_n + b) - 1$ kann hierbei als eine Art Schlupf verstanden werden. Das Problem kann nun formuliert werden:

$$\min_{w,b} \quad \mathcal{L}(w, b, \alpha) = \frac{1}{2} w^T w - \sum_{n=1}^{N} \alpha_n (y_n (w^T x_n + b) - 1) \tag{1.12a}$$

$$\max_{\alpha_n} \quad \alpha_n \geq 0 \text{ für } n = 1..N \tag{1.12b}$$

Nun kann die uneingeschränkte Optimierung von Gleichung 1.12a nach w und b gelöst werden indem die Ableitungen bestimmt und 0 gesetzt werden.

$$\begin{aligned} \nabla_w \mathcal{L} &= w - \sum_{n=1}^{N} \alpha_n y_n x_n \stackrel{!}{=} \vec{0} \\ w &= \sum_{n=1}^{N} \alpha_n y_n x_n \end{aligned} \tag{1.13}$$

$$\begin{aligned} \frac{\partial}{\partial b} \mathcal{L} &= -\sum_{n=1}^{N} \alpha_n y_n \stackrel{!}{=} 0 \\ \sum_{n=1}^{N} \alpha_n y_n &= 0 \end{aligned} \tag{1.14}$$

Die Ergebnisse von Gleichung 1.13 und Gleichung 1.14 können in Gleichung 1.12a eingesetzt werden.

$$\begin{aligned} \mathcal{L}(w, b, \alpha) &= \frac{1}{2} w^T w - \sum_{n=1}^{N} \alpha_n (y_n (w^T x_n + b) - 1) = \\ &= \frac{1}{2} w^T w - [\sum_{n=1}^{N} \alpha_n y_n b - \sum_{n=1}^{N} \alpha_n + \sum_{n=1}^{N} \alpha_n y_n w^T x_n] \end{aligned} \tag{1.15}$$

Weil $\sum_{n=1}^{N} \alpha_n y_n = 0$ aus Gleichung 1.14 fällt der Term $\sum_{n=1}^{N} \alpha_n y_n b$ weg:

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2} w^T w - [-\sum_{n=1}^{N} \alpha_n + \sum_{n=1}^{N} \alpha_n y_n w^T x_n] \tag{1.16}$$

Vergleicht man den Term $\sum_{n=1}^{N} \alpha_n y_n w^T x_n$ mit dem Ergebnis von Gleichung 1.13 erkennt man, dass $\sum_{n=1}^{N} \alpha_n y_n w^T x_n = w^T w$ gilt. Dies kann ausgeschrieben werden als:

$$\mathcal{L}(\alpha) = \sum_{n=1}^{N} -\frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} y_n y_m \alpha_n \alpha_m x_n^T x_m \tag{1.17}$$

Gleichung 1.17 beschreibt das Optimierungsproblem ohne Abhängigkeit von w und b, wir haben jetzt also eine Maximierung für α mit Nebenbedingungen:

$$\max_{\alpha} \quad \mathcal{L}(\alpha) = \sum_{n=1}^{N} -\frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{M} y_n y_m \alpha_n \alpha_m x_n^T x_m \tag{1.18a}$$

$$\text{mit} \quad \alpha_n \geq 0 \text{ für } n = 1..N \tag{1.18b}$$

$$\sum_{n=1}^{N} \alpha_n y_n = 0 \text{ für } n = 1..N \tag{1.18c}$$

Das in Gleichung 1.18 beschriebene Problem kann beispielsweise mittels eines Quadratic Programming Solvers gelöst werden. Als Ergebnis erhält man einen Vektor α mit allen α_n. Durch Einsetzen in $w = \sum_{n=1}^{N} \alpha_n y_n x_n$ kann w bestimmt werden.

Betrachtet man den Ergebnisvektor α wird man feststellen, dass sehr viele Werte 0 ergeben. In Gleichung 1.12a befindet sich der Term $\alpha_n(y_n(w^T x_n + b) - 1)$ und $(y_n(w^T x_n + b) - 1)$ wurde bereits zuvor als Schlupf bezeichnet. Das Produkt von Schlupf und α_n kann nur 0 werden, wenn entweder der Schlupf 0 ist oder α_n. Umgekehrt bedeutet dies, dass alle Vektoren, die einen minimalen Abstand zu der Trennebene haben, ein $\alpha_n \neq 0$ haben. Diese Vektoren werden Stützvektoren genannt.

Mit dieser Erkenntnis kann Gleichung 1.13 erneut analysiert werden:

$$w = \sum_{n=1}^{N} \alpha_n y_n x_n \tag{1.19}$$

Weil nur Stützvektoren ein $\alpha_n \neq 0$ aufweisen und somit auch nur Stützvektoren einen Beitrag zu w leisten kann Gleichung 1.19 stark vereinfacht werden:

$$w = \sum_{n \text{ ist Stützvektor}} \alpha_n y_n x_n \tag{1.20}$$

Der Gewichtsvektor w hängt also lediglich von einigen, in der Regeln wenigen, Stützvektoren ab.

Noch offen ist die Bestimmung des Bias b. Weil für Stützvektoren $y_n(w^T x_n + b) = 1$ gilt kann der Bias b aus jedem beliebigen Stützvektor bestimmt werden:

$$b = \frac{1}{y_n} - w^T x_n \tag{1.21}$$

1.2.4 Quadratic Programming Solver

TODO