Support Vector Machines mit Kerneln

Jonas Krug, Hendrik Sieck und Tim Schlottmann
TU Hamburg

30. Januar 2018

1 Grundlagen

Support Vector Maschinen sind ein binärer Klassifizierer. Datenmengen werden also in zwei Klassen eingeteilt. Zur Unterteilung der Klassen wird eine Hyperebene benutzt. Wie die Unterteilung stattfindet soll in diesem Abschnitt erklärt werden.

Folgende Definitionen bilden die Grundlage für SVMs:

- Anzahl an Datenpunkten: $m \in \mathbb{R}$
- Input: $\boldsymbol{x} \in \mathbb{R}^N$
- Output: $y \in \{-1, +1\}$
- Trainingsset: $S \in (\mathbb{R}^N \times \{-1, +1\})^m$

Ziel ist es, eine Hypothese zu finden, die einen Input $\boldsymbol{x}$ auf eine der beiden Klassen y abbildet:

$$\begin{aligned} h : \mathbb{R}^N &\to \{+1, -1\} \\ \boldsymbol{x} &\mapsto y \end{aligned}$$

1.1 Hyperebene

Betrachten wir ein Trainingsset S mit eingezeichneter Hyperebene H wie in Abbildung 1. Es fällt auf, dass ein Spalt (engl. margin) entsteht, dessen Grenzen sich durch zwei weitere Hilfsebenen H_+ und H_- darstellen lassen. H_+ und H_- sind hierbei parallel zu H. In Abbildung 1 ist H_+ die Begrenzung zur $+1$-Klasse und H_- die Begrenzung zur -1-Klasse. Die Vektoren, durch die die Hilfsebenen verlaufen, heißen Supportvektoren und sind in der Abbildung schwarz umrandet.

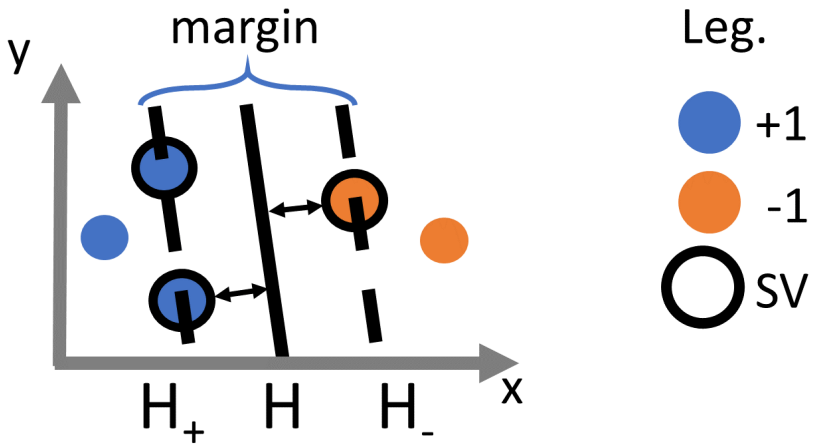

Abbildung 1: Hyperebene H zur Trennung der beiden Klassen $+1$ und -1

Nun gilt es die Hypothese zu finden. Hierfür betrachten wir zuerst die Hyperbenengleichung $H = \boldsymbol{w}^T\boldsymbol{x} + b = 0$. Hierbei werden die Parameter $\boldsymbol{w}$ und b so gewählt, dass für die Supportvektoren gilt: $y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) = 1$ So lässt sich die Hypothese wie folgt definieren:

$$h(\boldsymbol{x}_i) = \begin{cases} +1 & \text{wenn } \boldsymbol{w}^T\boldsymbol{x}_i + b \geq 0 \\ -1 & \text{wenn } \boldsymbol{w}^T\boldsymbol{x}_i + b \leq 0 \end{cases}$$

1.2 Minimierungsproblem

Die algorithmische Bestimmung der Parameter $\boldsymbol{w}$ und b erfolgt über ein Minimierungsproblem. Die Spalte zwischen den Hilfsebenen H_+ und H_- soll maximiert werden. Über die Projektionseigenschaft des Skalarproduktes lässt sich die Breite der Spalte über den Ausdruck $\frac{2}{\|\boldsymbol{w}\|}$ bestimmen. Die Breite der Spalte soll nun maximiert werden, was zu folgendem Minimierungsproblem führt:

$$\max_{\boldsymbol{w},b} \frac{2}{\|\boldsymbol{w}\|} \Leftrightarrow \min_{\boldsymbol{w},b} \|\boldsymbol{w}\| \Leftrightarrow \min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$\text{u.d.N. } y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 \geq 0$$

Durch die Verwendung von Lagrange Multiplikatoren lässt sich das Minimierungsproblem in eine Form bringen, die nur noch von der Langrangevariable α sowie den Supportvektoren bzw. den neu zu klassifizierenden Vektoren abhängt:

$$\boldsymbol{w} = \sum_i \alpha_i y_i \boldsymbol{x}_i$$
$$\sum_i \alpha_i y_i = 0$$
$$h(\boldsymbol{x}) = \begin{cases} +1 & \sum_i \alpha_i y_i \boldsymbol{x}_i^T\boldsymbol{x} + b \geq 0 \\ -1 & \sum_i \alpha_i y_i \boldsymbol{x}_i^T\boldsymbol{x} + b \leq 0 \end{cases}$$

2 Kernel Trick

2.1 Abbildfunktion

Mit der gezeigten Definition von SVMs lassen sich linear seperierbare Daten gut trennen. Es ist jedoch nicht möglich Daten mit komplexeren anordnungen mit hoher Wahrscheinlichkeit richtig zu klassifizieren.

Mit Hilfe einer Abbildfunktion $\phi(\vec{x})$ lässt sich diese einschrenkung umgehen. Hierzu transformieren wir den Eingabevector $\vec{x}$ in einen neun Raum in dem eine lineare Trennung möglich ist.

$$\phi : \mathbb{R}^n \to \mathbb{R}^m$$
$$\boldsymbol{x} \mapsto \boldsymbol{f}$$

Um die transformierten Vektoren für die SVM zu nutzen ersetzen wir das Skalarprodukt in Minimierungs- und Hypothesenfunktion durch durch ein Skalarprodukt der transformierten Vektoren.

2.2 Kernel

Wenn die Dimension der transformierten Vektoren größer ist als die der Ursprungsvektoren erhöht dies den Rechenaufwand. Zusätzlich wird $\phi(\vec{x})$ nur für das Skalarprodukt genutzt.

Wir definieren daher einen so genannten Kernel, welcher das gleiche ergebnis wie das Skalarprodukt der transformierten Vektoren aufweist, aber einfacher zu berechnen ist.

$$K(\boldsymbol{v}, \boldsymbol{w}) = \phi(\boldsymbol{v})^T \phi(\boldsymbol{w})$$
$$K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$$

2.3 Verschiedene kernel

In der Praxis wird der lineare, der polynomielle und der Gauß'sche Kernel am meisten genutzt.

2.3.1 Linear

Der Lineare Kernel ist equivalent zu einer SVM ohne Kernel.
Er ist in der Lage linear trennbare daten zu klassifizieren 2.
$K(\boldsymbol{v}, \boldsymbol{w}) = \boldsymbol{v}^T \boldsymbol{w}$

2.3.2 Polynomiell

Der polynomielle Kernel korespondiert zu einem ϕ welches die Daten auf einer polynomiellen Fläche verteilt.

2.3.3 Gauß

Der Gauß'sche Kernel nutz die Geometrische distanz zu anderen Datenpunkten zur Klassifizierung. Um den maximalen Abstand der zu vergleichenden Punkte festzulegen kann der Parameter ϕ angepasst werden 5.
Er ermöglicht die daten in verschiedenen Kluster zu trennen 4.
$K(\boldsymbol{v}, \boldsymbol{w}) = \exp(-\frac{||\boldsymbol{v}-\boldsymbol{w}||^2}{2\sigma^2})$

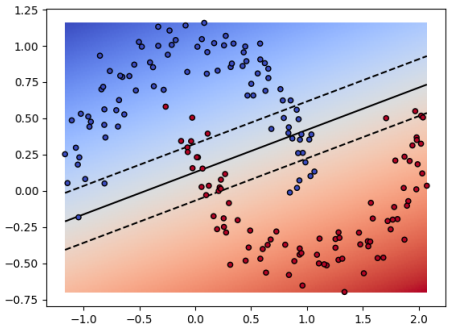

Abbildung 2: SVM mit linearem Kernel

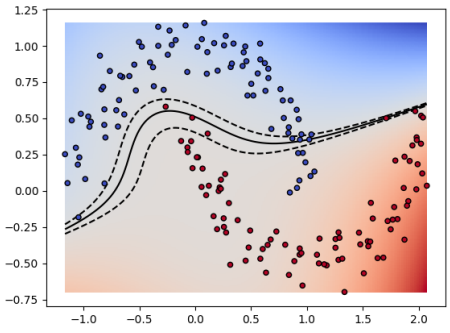

Abbildung 3: SVM mit Polynomiellen Kernel

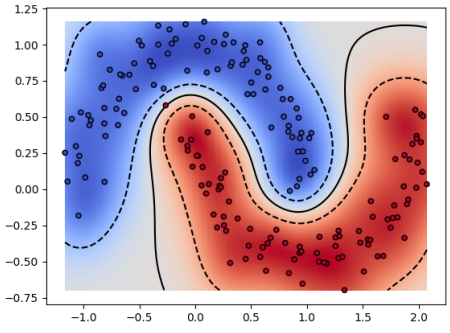

Abbildung 4: SVM mit Gauß'schem Kernel

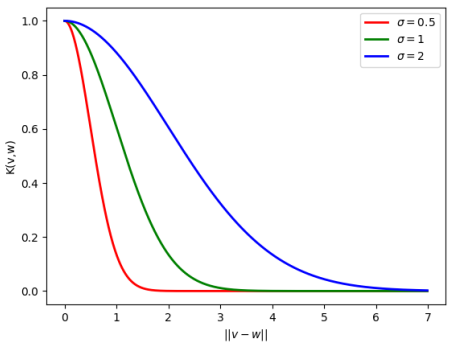

Abbildung 5: Verlauf des Gauß'schen Kernels

3 Tipps und Tricks zu Machine Learning und SVMs

Sämtlicher Code, Erläuterungen und das Präsentations-Skript sind online verfügbar: `https://github.com/NIPE-SYSTEMS/support-vector-machine`

Unten auf der Repository-Seite (siehe Link) befindet sich eine Beschreibung zu allen wichtigen Dateien im Repository.