Grundlagen von Support Vector Machines

Jens Pönisch

22. Januar 2019

Eine Support Vector Machine (SVM) ist ein Methode des überwachten maschinellen Lernens zur Klassifikation und Regression. Die n-dimensionalen Datenpunkte sollen durch eine «optimale» Hyperebene in zunächst zwei Klassen eingeteilt werden. Aus bekannten Trainingsdaten, deren Klassenzuordnungen bekannt sind, wird eine «optimale» Hyperebene so konstruiert, dass die Punkte verschiedener Klassen auf unterschiedlichen Seiten der Ebene liegen. Um die Klasse eines neuen Punktes zu bestimmen, muss lediglich ermittelt werden, auf welcher Seite der Ebene er liegt.

Die Hyperebene wird dabei so konstruiert, dass der kleinste Abstand eines Trainingspunktes zu dieser Ebene maximal wird. Damit erhalten wir folgendes konvexes Optimierungsproblem:

$$\begin{aligned} &\arg\min_{\mathbf{w},\,b} \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{s. t.} \qquad &t_i(\langle \mathbf{w},\, \mathbf{x}_i\rangle + b) \geq 1,\, i = 1,\, \ldots,\, M, \end{aligned}$$

Die Parameter $\mathbf{w}$ und b beschreiben die gesuchte Ebene, $\mathbf{x}_i$ die Trainingsdatenpunkte und t_i die Klasse jedes Datenpunktes (-1 oder 1). Die der Ebene am nächsten liegenden Punkte bezeichnet man als *Supportvektoren*.

Durch «Ausreißer» ist eine solche Trennung jedoch nicht immer möglich, deshalb wird durch *Schlupfvariablen* erlaubt, dass sich einzelne Datenpunkte auch dichter oder gar auf der falschen Seite der Trennebene befinden. Ein Regularisierungsparameter C legt fest, wie groß dieser Schlupf werden darf.

Bestimmte Datenmengen lassen sich durch Hyperebenen jedoch gar nicht trennen, man versucht deshalb, die Datenpunkte durch eine nichtlineare Funktion zu transformieren und die transformierten Punkte durch eine Hyperebene zu trennen. Problem ist die massive Zunahme der Dimension der Datenpunkte, was die Lösung des Optimierungsproblems erschwert. Statt dessen geht man mithilfe der *Lagrange*-Funktion zum dualen Problem über.

$$\begin{aligned} &\arg\min_{\mathbf{a}} \left\{ \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{M} a_i a_j t_i t_j k(\mathbf{x}_i,\, \mathbf{x}_j) - \sum_{i=1}^{M} a_i \right\} \\ \text{s. t.} \qquad &0 \leq a_i \leq C, \qquad i = 1,\, \ldots,\, M \\ &\sum_{i=1}^{M} a_i t_i = 0 \end{aligned}$$

In der Lösung sind nur die Koeffizienten a_i der Supportvektoren ungleich Null. Nur noch diese Werte sind notwendig, um neue Datenpunkte zu klassifizieren, die restlichen Daten der Trainingsmenge werden nicht weiter benötigt.

Hier tauchen die Datenpunkte nur noch in Form von Skalarprodukten auf, wir ersetzen diese durch eine geeignete reellwertige *Kernelfunktion* $k(\mathbf{x}_i, \mathbf{x}_j)$. Häufig verwendet man hier Funktionen über den Abstand zweier Datenpunkte, $r = \|\mathbf{x}_i - \mathbf{x}_j\|$, z. B. den *Gauß*-Kernel $e^{-\gamma r^2}$. Dieses Verfahren heißt *Kernel-Trick*.

Da SVM für die Abstandsdefinition die Euklidische Norm benutzen, sollten die Daten in allen Dimensionen gleich skaliert sein. Nominalskalen können mit einer *One-Hot*-Codierung in numerische Werte überführt werden, indem jede Kategorie eine eigene Komponente erhält, die 1 ist, wenn diese Kategorie zutrifft, und 0 im gegenteiligen Fall.

Das Training einer SVM besteht in der Suche nach optimalen Parameterwerten für die Regularisierung C und den oder die Kernelparameter, was mithilfe der *Gittersuche* geschieht. Um ein Overfitting zu vermeiden, verwendet man dabei eine *Kreuzvalidierung*, indem man die Trainingsmenge geeignet in k Gruppen einteilt und davon jeweils $k - 1$ Gruppen für die Lösung des Optimierungsproblems und eine Gruppe für die Validierung verwendet.

Um zwischen mehr als zwei Klassen zu unterscheiden, gibt es zwei grundlegende Ansätze, *one-versus-the-rest* und *one-to-one*. Im ersten Ansatz wird für jede Klasse eine SVM konstruiert, die entscheidet ob ein Punkt zu dieser Klasse gehört oder nicht. Ein neuer Punkt wird von allen diesen SVMs bewertet und der Klasse mit der höchsten Wertung zugeordnet. Im zweiten Ansatz werden SVMs für jedes Paar von zwei Klassen Klassen erzeugt und ein neuer Punkt der Klasse zugeordnet, die von den meisten SVMs ausgewählt wurde.

SVMs sind bei Trainingsdatensätzen mittlerer Größe (einige zehntausend Datenpunkte) mit mittlerer Dimension gut geeignet, da sie relativ kompakte Modelle erzeugen, relativ schnell klassifizieren und weniger als z. B. Entscheidungsbäume zum Overfitting neigen. Eine Erweiterung der Trainingsmenge bedeutet verfahrensbedingt eine komplett neue Modellerstellung, ein vereinfachtes «Nachtrainen» ist nicht möglich.

Literatur

[Are15] Arens et al.: Mathematik. Springer Spektrum. 2015.

[Bis06] Bishop: Pattern Recognition and Machine Learning. Springer. 2006.

[Bit16] Bitterlich: Numerische Verfahren zur Lösung von Support Vector Machines. Masterarbeit, TU Chemnitz. 2016.

[Cor95] Cortes, Vapnik: Support-vector networks. In: Machine Learning, 20. 1995.

[Hsu16] Hsu, Chang, Lin: A Practical Guide to Support Vector Classification. National Taiwan University. 2016. `https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf`