Materialien zur Vorlesung Lernen
Lerntheorie und Support-Vektor-Maschinen

Ralf Der

January 15, 2007

Allgemeine Bemerkungen zum Text: Die mit *** gekennzeichneten Kapitel sind weiterführend und stellen keinen Prüfungsstoff dar!

1 Statistische Lerntheorie

Die statistische Lerntheorie macht allgemeine Aussagen über die Leistungsfähigkeit von Lernmaschinen. Einige ganz wenige Ergebnisse sollen hier vorgestellt werden. Weiterführende Literatur siehe [1].

1.1 Lernen allgemein:

- Gegeben: Lerner als parametrisierte Funktion $f(x; w)$ mit $x \in \Omega^{in} \subseteq R^n$ Inputvektor und $w \in R^p$ Parametervektor. Beispiel Neuron $f(x; w) = g(w \cdot x)$. $p \cdot q$ bezeichnet das Skalarprodukt zwischen den Vektoren $p, q \in R^n$, d.h. $p \cdot q = \sum_{i=1}^{n} p_i q_i$.
- Gegeben Menge $\Omega = \{(x_i, y_i) | x_i \in \Omega^{in} \subseteq R^n\}$ von Trainingsbeispielen, y_i ist der Sollwert zu x_i. Fehler:

$$E(w) = \frac{1}{m} \sum_{i=1}^{m} D(f(x_i; w), y_i) \tag{1}$$

wobei $m = \#\Omega$ die Zahl der Datenpunkte ist und $D(p,q)$ ist der quadratische Abstand zwischen zwei Vektoren p und q

$$D(p,q) = \|p-q\|^2 = \sum_{\alpha=1}^{n} (p_\alpha - q_\alpha)^2$$

- Gesucht w^{opt} mit $E = Min$, d.h.

$$w^{opt} = \arg\min_w E(w)$$

1.2 Einige Ergebnisse der Lerntheorie:

- Spezialfall Klassifikation: $f : R^n \to \{+1,-1\}$ und $y(x) \in \{+1,-1\}$. $f(x)$ heißt dann auch Entscheidungsfunktion. Beispiel bipolares Neuron $f(x;w) = sign(w \cdot x)$.

- Statt E verwende mittleres empirisches Risiko der Fehlklassifikation ermittlet aus m Datenpunkten (empirical risk)

$$R_m[f] = \frac{1}{m} \sum_{i=1}^{m} |f(x_i;w) - y_i| \tag{2}$$

  Das exakte Risiko ist

$$R[f] = \sum_{i=1}^{m} |f(x_i;w) - y_i| \; P(x_i,y_i)$$

  mit $P(x,y)$ die Wahrscheinlichkeit, dass die Trainigsinstanz (x,y) gezogen wird.

- Für eine gegebene Trainingsmenge Ω muss die Kapazität von f passend gewählt werden.

- Maß der Kapazität: **Vapnik-Chervonenkis-Dimension**. Betrachte eine Menge $M = \{x_1,...,x_m | x_i \in R^n \; \forall i\}$ von Datenpunkten mit Klassenlabeln $y(x_i) = y_i \in \{+1,-1\}$. Es gibt 2^m verschiedene Möglichkeiten der Zuordnung der x_i zu einer Klasse.

- Begriff des Shatterns (Zerschlagen): Eine Maschine $f(x;w)$ kann eine Menge von Trainingsbeispielen $t_i = (x_i, y(x)), i = 1,...,m$ zerschlagen, wenn es ein w gibt, so dass das Risiko $R_m[f] = 0$, vgl. 1.

- Die Vapnik-Chervonenkis-Dimension einer Entscheidungsfunktion $f(x;w)$ ist die größte Zahl m von Datenpunkten in allgemeiner Lage die $f(x;w)$ zerschlagen kann. Anders: Für jeden Satz aus m Datenpunkten (bzw Lernbeispielen) gibt es ein w so dass das Risiko R_m gleich Null ist. Oder: Die VC-Dimension einer Menge von Hypothesen H ist die maximale Anzahl von Beispielen m, die von H zerschlagen wird.

- Beispiel: Wie ist die VC - Dim. der Maschine (mit $w \in R^1$und $x \in R^2$)

$$f(x;w) = sign(x \cdot x - w) = sign\left(x_1^2 + x_2^2 - w\right)$$

  Antwort: VCD = 1. Die VCD der Maschine

$$f(x;w) = sign(w_1 x \cdot x - w_2)$$

  ist gleich 2.

1.3 Hyperebenen-Klassifikator (line machines)

Vorbetrachtung: Hessesche Normalform einer Geraden bzw. Hyperebene G: Ist gegeben durch einen Vektor $\hat{w}$ und eine Zahl d. Die Hyperebenen ist die Menge aller Punkte $x \in R^n$ für die

$$\hat{w} \cdot x = d$$

mit $b \in R^1$ Geometrisch ist $\hat{w}$ der Normalenvektor, d.h. ein Einheitsvektor senkecht zur Hyperebene und d ist der Abstand der Geraden zum Ursprung des Koordinatensystems. Mit einem beliebigen Vektor w schreiben wir

$$w \cdot x + b = 0$$

wobei jetzt der Abstand [1]

$$d = -\frac{b}{\|w\|}$$

[1]Wie bisher ist die Länge des Vektors w

$$\|w\| = \sqrt{\sum_{i=1}^{n} w_i^2}$$

ist. Für den Abstand A eines beliebigen Punktes $x \in R^n$ zur Ebene gilt

$$A(x) = \frac{w \cdot x + b}{\|w\|} \tag{3}$$

mit $A > 0$ ($A < 0$) wenn der Punkt über (unter) der Hyperebene G liegt (oben ist wo w hinzeigt).

Die **Entscheidungsfunktion**

$$f(x; w, b) = sign(w \cdot x + b)$$

gibt an ob ein Punkt x ober- oder unterhalb von G liegt. Für die VCD der Maschine $f(x; w, b)$ gilt:

Für $2-$dimensionale Systeme ist $VCD = 3$. (für drei Punkte findet man immer eine Trenngerade, für vier kann man leicht ein Gegenbeispiel angeben.)

Für $n-$dimensionale Systeme ist

$$VCD = n + 1$$

(Nichtrivial)

1.4 Anwendungen der VCD***

Man kann z.B . Aussagen über den zu erwartenden Fehler auf der Testmenge machen: Sei $m > VCD$ (mehr Datenpunkte als zur Festlegung der Maschine nötig) dann gilt mit Wahrscheinlichkeit $1 - \delta$

$$R[f] < R_m[f] + \phi(h, m, \delta)$$

mit $h = VCD$ und

$$\phi(h, m, \delta) = \sqrt{\frac{\left(\ln \frac{2m}{h} + 1\right) h + \ln \frac{4}{\delta}}{m}}$$

Für sehr große m gilt

$$\phi(h, m, \delta) = O\left(\sqrt{\frac{\ln m}{m}}\right)$$

2 Support-Vektor-Maschinen (SVM)

Die SVM sind eine der beliebtesten Lernmaschinen für typische Aufgabenstellung aus Mustererkennnung, data mining, Spracherkennung u.a. dar.

2.1 Lineare SVM

Aufgabe ist es, die Trennebene (Hperebene) G zu finden, die für gegebene Punkte $x_i,\ i \in \{1, ..., m\}$ den größten Rand (breitester Trennstreifen) zwischen den Gebieten der beiden Klassen hat. Seien zunächst zwei Punkte $x_1 \in R^n$ und $x_2 \in R^n$ mit Klassenlabel $x_1 \longmapsto y_1 = 1$ und $x_2 \longmapsto y_2 = -1$ betrachtet. Sei $w \cdot x + b$ eine Kandidaten-Trennebene. Bei geeigneter Skalierung von w und b können wir die Forderung dass die beiden Punkte den gleichen Abstand zu G haben aber über bzw. unter dieser liegen durch die Bedingung $w \cdot x_1 + b = y_1$ und $w \cdot x_2 + b = y_2$ oder

$$(w \cdot x_i + b)\, y_i - 1 = 0 \tag{4}$$

formulieren. Mit Gleichung 3 findet man, dass die Breite des Streifens parallel zu G zwischen den Punkten gerade $2/\left\|w\right\|$ ist. Diese Breite gilt es zu maximieren, was bedeutet $\|w\|$ zu minimieren. Wegen der Bedingungen (4) heißt das, dass w gedreht und gestaucht bzw. gestreckt wird.

Allgemein können mehr als zwei Punkte die Bedingung (4) gleichzeitig erfüllen. Alle anderen Punkte sind dann weiter weg, für sie gilt $(w \cdot x_i + b)\, y_i - 1 > 0$. Allgemein heißen damit unsere Nebenbedingung für die Minimierung von $\|w\|$

$$(w \cdot x_i + b)\, y_i - 1 \geq 0 \ \ \forall i = 1, \ldots, m \tag{5}$$

Wir haben deshalb das folgende

Optimierungsproblem: Minimiere $\|w\|$ mit Nebenbedingungen (5). Lösung durch Methode der Lagrangeschen Multiplikatoren. Wir definieren die Lagrangefunktion des Problems

$$L\left(w, b, \alpha\right) = \frac{1}{2}\left\|w\right\|^2 - \sum_{i=1}^{m} \alpha_i \left(\left(w \cdot x_i + b\right) y_i - 1\right) \tag{6}$$

wobei die α_i die Lagrangeschen Mulitplikatoren sind, die den Einfluss der Nebenbedingungen auf die Minimierung von $\left\|w\right\|^2$ haben. Wegen (5) muss

$$\alpha_i \geq 0$$

gelten. Das heißt minimiere L bezüglich w und b und maximiere bezüglich der α_i. Die gesuchten Minima von L finden wir durch

$$\frac{\partial L}{\partial w} = 0,\ \frac{\partial L}{\partial b} = 0$$

mit Lösungen

$$\sum_{i=1}^{m} \alpha_i y_i = 0,\ w = \sum_{i=1}^{m} \alpha_i y_i x_i \tag{7}$$

Die Maximierung bezüglich α liefert[2]

$$\alpha_i = 0\ \forall i \text{ für die } (w \cdot x_i + b)\, y_i - 1 > 0$$

Es "überlebt" also nur die im allgemeinen kleine Anzahl von Vektoren für die

$$(w \cdot x_i + b)\, y_i - 1 = 0$$

gilt. Nur diese gehen in w ein (vgl. (7)), d. h. nur diese "stützen" die Hyperebene G. Sie werden deshalb die Support-Vektoren von G genannt. Geometrisch liegen sie auf dem Rand des Trennstreifens (engl. margin) zwischen den Klassen. In einer physikalischen Interpretation (über die Lagrangefunktion) summieren sich die durch die Support-Vektoren auf G ausgeübten Kräfte $F_i = y_i \alpha_i \hat{w} = y_i \alpha_i w / \|w\|$ ebenso wie die Drehmomente[3] zu Null, siehe (7).

Die **Entscheidungsfunktion** (gibt für einen beliebigen Datenvektor x die Klasse aus) ist damit gefunden:

$$f(x) = sign\left(\sum_{i \in SV} y_i \alpha_i\ x \cdot x_i + b\right) \tag{8}$$

[2]Klar, da für alle Vektoren x_i mit $(w \cdot x_i + b)\, y_i - 1 > 0$ wegen der Forderung $\alpha_i \geq 0$ der Beitrag $-\alpha_i \left((w \cdot x_i + b)\, y_i - 1\right)$ am größten (nämlich gleich 0) ist wenn $\alpha_i = 0$.

[3]Betrachte die Kraft

$$\vec{F}_i = \alpha_i y_i \hat{w}$$

(mit $\hat{w} = \vec{w} / \|\vec{w}\|$), die auf einen Ortsvektor $\vec{x}_i$ wirkt. Das Drehmoment ist

$$\vec{m}_i = \vec{x}_i \times \vec{F}_i$$

(Kreuzprodukt der Vektoren). Die Summe der Drehmomente ist

$$\sum_{i=1}^{m} \vec{x}_i \times \alpha_i y_i \hat{w} = \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \times \hat{w} = w \times \hat{w} = 0$$

Die Summe läuft also nur über die Support-Vektoren ($\alpha_i > 0$), die zusammen mit b und den α_i über das Optimierungsproblem gefunden werden.

2.2 Das duale Problem

Durch Umtransformation kann man mittels (7) das Optimierungsproblem auch so formulieren

$$\underset{\alpha}{Maximiere}\, W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j x_i \cdot x_j \tag{9}$$

mit Nebenbedingungen

$$\alpha_i \geq 0 \; \forall i \text{ und } \sum_{i=1}^{m} \alpha_i y_i = 0 \tag{10}$$

Dieses Problem bildet meist die Grundlage für die praktischen Anwendungen

2.3 Nichtlineare SVM

Bisher nur linear separable Probleme lösbar. Idee für nichtlineare Probleme ist, in einem ersten Schritt, die Eingabedaten x_i durch eine nichtlineare Transformation $\phi : R^n \rightarrow R^s$ so umzuwandlen, dass das nichtlineare Problem in ein linear trennbares überführt wird, für das dann die Supportvektoren gefunden werden können. Im allgemeinen ist $s > n$, d.h. die Transformation

$$x \rightarrow \phi(x) \tag{11}$$

ist mit einer Dimensionserhöhung verbunden. Die Vektoren $\phi(x) \in R^s$ heißen Merkmals-Vektoren (*feature vectors*), der von ihnen aufgespannte Raum der Merkmalsraum. Formal ergibt sich der Übergang zum nichtlinearen Problem durch Ersetzen der Vektoren x bzw. x_i durch ihre Merkmalsvektoren.

Man baut auf der Formulierung durch das duale Problem auf, d.h. auf den Gleichungen (8), (9) und (10) und bildet die Skalarprodukte mit den transformierten Vektoren. Es ergibt sich das Optimierungsproblem

$$\underset{\alpha}{Maximiere}\, W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \phi(x_i) \cdot \phi(x_j) \tag{12}$$

mit Nebenbedingungen (10). Die Entscheidungsfunktion ist entsprechend

$$f(x) = sign\left(\sum_{i \in SV} y_i \alpha_i \ \phi(x) \cdot \phi(x_i) + b\right) \tag{13}$$

Support-Vektoren sind durch $\alpha_i > 0$ ausgezeichnet.

2.4 Kernel-Maschinen

Das Auffinden einer geeigneten nichtlinearen Transformation ist in der Praxis schwierig, vor allem sind die Merkmalsräume oft sehr hochdimensional. Man umgeht die gesamte Problematik durch den Trick, ein Skalarprodukt zweier Vektoren $\phi(p)$ und $\phi(q)$ als einen Kern (engl. kernel) zu interpretieren

$$\phi(p) \cdot \phi(q) = k(p, q) \tag{14}$$

und diesen direkt vorzugeben, ohne den Umweg über ϕ zu nehmen. Das Optimierungsproblem lautet nun

$$\underset{\alpha}{Maximiere}\ W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \tag{15}$$

mit Nebenbedingungen (10). Die Entscheidungsfunktion ist

$$f(x) = sign\left(\sum_{i \in SV} y_i \alpha_i \ k(x, x_i) + b\right) \tag{16}$$

Support-Vektoren wie bisher durch $\alpha_i > 0$ ausgezeichnet.

Beliebte kernel-Funktionen sind

$$\begin{aligned}
k(p,q) &= (p \cdot q)^d \text{ Polynomiale Klassifkation (homogen)} \\
k(p,q) &= (p \cdot q + 1)^d \text{ Polynomiale Klassifkation (inhomogen)} \\
k(p,q) &= \exp\left(\|p - q\|^2 / 2\sigma^2\right) \text{ Gaussche radiale Basisfunktionen} \\
k(p,q) &= \tanh(\gamma p \cdot q + \theta) \text{ Neuronale Netze}
\end{aligned}$$

2.5 Nicht trennbare Mengen

Falls die Klassen nicht trennbar sind (Verrauschung der Klassen oder zu viele Datenvektoren) kann man das folgende verallgemeinerte Problem durch Einführen sog. Schlupfvariabler

$$\xi_i \geq 0 \tag{17}$$

betrachten. Die Randbedingungen (5) werden dann relaxiert zu

$$(w \cdot x_i + b)\, y_i - 1 \geq -\xi_i \;\; \forall i = 1, \ldots, m \tag{18}$$

Die Zielfunktion für das Optimierungsproblem ist dann

$$\tau(w, \xi) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{m} \xi_i$$

mit den Nebenbedingungen (17) und (18). Die Konstante C wird von Hand gewählt und bestimmt wie sehr die ursprünglichen Randbedingungen (5) an die Minimierung von $\|w\|^2$ (das die Breite des Trennstreifens festlegt) "aufgeweicht" bzw. relaxiert werden. Wir erlauben so, dass schlecht erfassbare Punkte auch fehlklassifiziert werden können. Die Konstante C legt damit die Kosten für eine Fehlklassifikation bzw. die Weichheit des Trennstreifens (margin) fest. Wir erhalten so eine *soft margin hyperplane.*

Die Zielfunktion für eine Kernel-Maschine in der dualen Fomulierung lautet wieder (vgl. (15))

$$\underset{\alpha}{Maximiere}\, W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \tag{19}$$

mit Nebenbedingungen

$$0 \leq \alpha_i \leq C \; \forall i \text{ und } \sum_{i=1}^{m} \alpha_i y_i = 0 \tag{20}$$

Der einzige Unterschied liegt also in der Einschränkung an die Lagrangeschen Multiplikatoren α_i.

2.6 Bemerkungen

Vorteile der Kernel-Maschinen:

- Systematischen Verfahren zur Lösung nichtlinearer Klassifikationsprobleme druch nichtlineare Abbildung in einen Merkmalsraum.
- Durch die Kernelmethode bleibt die Abbildung $x \rightarrow \phi(x)$ implizit, der unterliegende Merkmalsraum kann ∞-dimensional sein, ohne dass sich das in der Rechenkomplexität äußert.
- Auffinden der SVen ist quadratisches Optimierungsproblem, konvex (keine lokalen Minima), vgl. (15).
- Lösung in polynomialer Zeit möglich. Kein "Fluch der Dimension".
- Sparsamkeitsprinzip: Nur einige wenige Datenvektoren zur Definition der Trennebene nötig (Kuhn-Tucker-Theorem: nur die $\alpha_i \neq 0$ die nahe an der Hyperebene liegen (Abstand 1)

2.7 Mathematische Ergänzungen*** (kein Prüfungsstoff)

Mathematisch dient ein Kern (kernel) der Transformation zwischen Funktionen. Sei $g : R^n \rightarrow R^1$ dann kann man diese transformieren

$$q(x) = \int K(x,s)\, g(s)\ ds$$

Das kann als die Anwendung eines durch K definierten Operators auf die Funktion g betrachtet werden. Dieser habe Eigenwerte λ_l und Eigenfunktionen $e_l(x)$ so dass

$$\lambda_l e_l(x) = \int K(x,s)\, e_l(s)$$

Ist K positiv semidefinit[4] so gilt Mercer´s Theorem:

$$K(x,s) = \sum_l \lambda_l e_l(x)\, e_l(s) \tag{21}$$

[4]Das heißt

$$\int g(x)\, K(x,s)\, g(s)\ dx\ ds \geq 0$$

für beliebige Funktionen $g(s)$. Für die Eigenswerte gilt dass diese nicht negativ sein können.

Definiere $\phi : R^n \to R^s$ oder $\phi(x) = (\phi_1(x), \ldots, \phi_s(x))^T$ mit $\phi_l = \sqrt{\lambda_l} e_l(x)$ dann wird (**??**)

$$K(a, b) = \phi(a) \cdot \phi(b) \tag{22}$$

Der Merkmalsraum wird also durch die Eigenvektoren von K aufgespannt – die Eigenfunktionen sind die Merkmale in dieser Darstellung. Umgekehrt kann man auch zeigen, dass K in (22) für jede beliebige Funktion ϕ positiv semidefinit ist.

3 Beispiel: Von RBF-Netzwerken to SVM

SVM haben eine gewisse Ähnlichkeit zu den *radial basis function* (RBF) Netzwerken aus der Neuroinformatik. Diese bestehen aus einer Inputschicht (Eingabe Vektor $x \in R^n$), einer Zwischenschicht und einer Ausgabeschicht aus linearen Neuronen. Mathematisch ist das Netzwerk durch die Funktion (betrachten nur ein Ausgabeneuron, d.h. $f : R^n \to R^1$)

$$f(x; w, b) = \sum_{i=1}^{m} a_i \rho(\|x - c_i\|) + b \tag{23}$$

(w fasst alle Paramter außer b zusammen) bzw. für einen Klassifikator (Entscheidungsfunkion)

$$f(x; w, b) = sign\left(\sum_{i=1}^{m} a_i \rho(\|x - c_i\|) + b\right) \tag{24}$$

Die Funktion $\rho : R^1 \to R^1$ heißt radiale Basisfunktion, Beispiel

$$\rho(u) = \exp\left(-\frac{u^2}{2\sigma^2}\right)$$

die $c_i \in R^n$ sind die Zentren der Basisfunktionen. In der Praxis hat es sich als günstig erwiesen, mit normierten Basisfunktionen zu arbeiten, statt (23) setzt man

$$f(x; w) = \sum_{i=1}^{m} a_i r(\|x - c_i\|) + b \tag{25}$$

mit den normierten RBF

$$r(\|x - c_i\|) = \frac{\rho(\|x - c_i\|)}{\sum_{i=1}^{m} \rho(\|x - c_i\|)}$$

Der Unterschied zwischen (13) und (24) liegt bei Verwendung eines Gauss-Kernels nur in der Wahl der Zentren c_i und des Bias b, die bei den RBF Netzen "von Hand" eingerichtet bzw. gelernt werden müssen. Die Parameter a_i und b können leicht durch einen Gradientenabstieg auf dem Trainingsfehler gelernt werden. Für die c_i kann man geeignete Eingabevektoren x_i auswählen, was die Nähe zu den SVM besonders nahe legt, mit dem Unterschied, dass die Support-Vektoren sich aus dem Optimierungsproblem automatisch ergeben.

Mit dem Fehler

$$E\left(x; w, b\right) = \frac{1}{2}\left(y^{soll} - f\left(x\right)\right)^2$$

folgt sofort für die linearen Gewichte

$$\begin{aligned} \Delta a_i &= \varepsilon\left(y^{soll} - f\left(x\right)\right)\rho\left(\|x - c_i\|\right) \\ \Delta b &= \varepsilon\left(y^{soll} - f\left(x\right)\right) \end{aligned}$$

Im Prinzip können die Zentren ebenso wie die Breiten σ auch gelernt werden (Gradientenabstieg). Vorteil ist die Tatsachen, dass das on-line geht. Nachteil die vielen Nebenminima. In der Praxis verwendet man deshalb entweder eine Auswahl von als wichtig empfundener Eingabevektoren x_i oder man nutzt *unsupervised learning* Verfahren wie Vektor-Quantisierer oder selbstorganisierende Karten (siehe Script Neuroinformatik).

References

[1] V. N. Vapnik. *The nature of statistical learning theory.* Springer, 1995.