Maschinelles Lernen und Datenanalyse

In der Mess- und Prüftechnik

PD Stefan Bosse

Universität Bremen - FB Mathematik und Informatik

Regression und Support Vector Machines

Wie funktioniert Regression?

SVM gehören zu den Regressionsverfahren

SVM nutzen aber bei der Parameteranpassung (Training) eine andere Fehlerfunktion (Loss) als bei anderen gängigen Regressionsverfahren (z.B. Least-Square Minimierung)

SVM können primär kategorische und weniger numerische Zielvariablen abbilden

SVM sind aber (zunächst) lineare Klassifikatoren!

Regressionsverfahren

- Klassifikationsprobleme sind durch Booleschen Ausgabevariablen gekennzeichnet (Klasse c_i={true,false})

- Bei Regressionproblemen findet hingegen eine Ausgabe mit kontinuierlichen Variablen statt, idealerweise $y \in [0,1]$

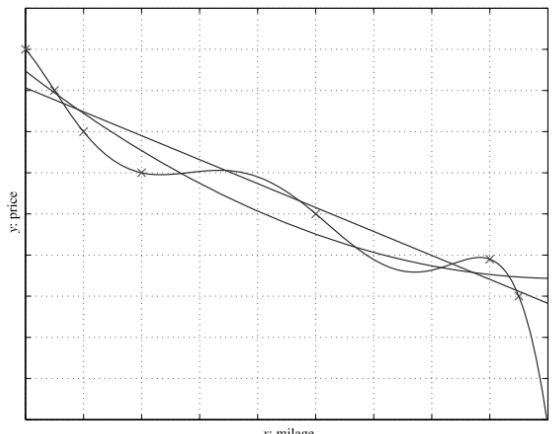
- D.h. es gibt Trainingsdaten mit:

$$D = X^t = \left\{ \vec{x}^t, r^t \right\}_{t=1}^{N}$$

wobei $r \in \mathbb{R}$. Wenn Rauschen vernachlässigt wird handelt es sich um ein reines Interpolationsproblem.

- Das Ziel ist es nun, eine Funktion $f(\vec{x})$ zu finden, die die Trainingsdaten optimal repräsentiert (also die beste Hypothese g von f finden)

Beispiel



[19] Abb. 1. Lineare Gerade, Polynome zweiter Ordnung und sechster Ordnung werden an denselben Satz von Punkten angepasst. Die höchste Ordnung ergibt eine perfekte Passform, aber angesichts dieser vielen Daten ist es sehr unwahrscheinlich, dass die reale Kurve so geformt ist. Die zweite Ordnung scheint besser zu sein als die lineare Anpassung bei der Erfassung des Trends in den Trainingsdaten (Extrapolation).

Es wird immer einen Fehler ε geben:

$$r^t = f\left(\vec{x}^t\right) + \epsilon$$

- Ziel der Regression ist es diesen Fehler über eine Verlustfunktion zu minimieren in dem Parameter $\vec{\Theta}$ der Funktion *f* angepasst werden:

$$\arg \min_{\theta} \ \epsilon \rightarrow E(g|X) = \frac{1}{N} \sum \left(r^t - g\left(\vec{x}^t\right)\right)^2$$

Lineare Multivariate Regression

- Es wird angenommen dass die Funktion $f(\vec{x})$ durch eine lineare Funktion über $\vec{x}$ abgebildet werden kann.
- Dann gilt:

$$g(\vec{x}) = w_1 x_1 + w_2 x_2 + .. + w_d x_d + w_0 = \sum_{j=1}^{d} w_d x_d + w_0$$

Schon dieses Problem kann unterbestimmt sein, d.h., es kann unendlich viele Hypothesen *g* von der unbekannten Funktion *f* geben!

- Bei nichtlinearen Zusammenhängen wird die Regressionsfunktion noch komplexer!

Nichtlineare Univariate Regression

- Es gibt nur eine Eingabevariable und die Hypothesenfunktion wird durch ein Polynom *k*-ter Ordnung approximiert:

$$g(x) = w_0 + w_1 x + w_2 x^2 + .. + w_k x^k = \sum_{j=1}^{k} w_j x^j + w_0$$

- Wird von einem Polynom ersten Grades ausgegangen (Gerade) dann gilt es folgende Gleichung zu bestimmen und das Minimierungsproblem zu lösen:

$$g(x) = w_1 x + w_0$$

$$E\left(w_1, w_0 | X^t\right) = \frac{1}{N} \sum_{t=1}^{N} \left(r^t - \left(w_1 x^t + w_0\right)\right)^2$$

- Den Minimumspunkt dieser Fehlergleichung findet man durch Gradientenbildung der Parameter, das bedeutet dann:

$$w_1 = \frac{\sum_t x^t r^t - \overline{xr} N}{\sum_t \left(x^t\right)^2 - N \overline{x}^2}$$

$$w_0 = \overline{r} - w_1 \overline{x}$$

$$\overline{x} = \sum_t \frac{x^t}{N}, \overline{r} = \sum_t \frac{r^t}{N}$$

Nichtlineare Multivariate Regression

- Sehr hochdimensionales Problem! Und i.A. völlig unterbestimmt (d.h. kein eindeutigen Zusammenhänge zwischen $\vec{x}$ und y)

$$g(\vec{x}) = \sum_{i=1}^{d}\sum_{j=1}^{k} w_{i,j} x_i^j + w_0$$

- Es können auch exponentielle, logarithmische, und sinusoidale Terme hinzukommen!

- Ein numerisches Lösen ist meist nicht mehr möglich; daher Verwendung nichtlinearer Randbedingungslöser sowie statistische Verfahren wie randomiserte Monte Carlo Simulation und Simmuliertes Abkühlen (Evolutionäre Algorithmen?)

Warum können hochdimensionale Polynome nicht mehr mit gradientenbasierten Verfahren numerisch auf einem Computer (gut oder überhaupt) lösbar sein? Hinweis: Wie entwickeln sich Gradienten bei Polynomen sehr hoher Ordnung oder gar Exponentialterme wie b^n?

???

Direkte Lösungsverfahren

Single Value Decomposition (SVD)

- Erweiterung der Eigenwertanalyse und verwendet Matrixalgebra und Inversionsmethoden [4]
- Ansatz: Dekomposition einer (nichtlinearen) Funktion $f(\vec{x},\vec{\Theta})$ mit b Basisfunktionen ϕ, z.B. *sin* oder ähnlich, mit Parametersatz Θ:

$$f_\Theta(\vec{x}) = \sum_{j=1}^{b} \Theta_j \phi_j(x)$$

$$f_\Theta(\vec{x}) = \vec{\Theta}^T \vec{\phi}(x)$$

- Z.B. $\vec{\phi}(x) = (1,x,x^2,..,x^{b-1})^T$, oder $\vec{\phi}(x) = (1,\sin(x),\cos(x),\sin(2x),...)^T$

- Die gesamte Trainingstabelle wird in Matrixform repräsentiert, und somit erhält man eine Parametermatrix $\hat{\Phi}$ mit den Basisfunktionstermen für die anzupassenden Funktion $f(\vec{x})_\Theta$ (Design Matrix):

$$\hat{\Phi} = \begin{pmatrix} \phi_1(\vec{x}_1) & .. & \phi_b(\vec{x}_1) \\ \cdots & \cdots & \cdots \\ \phi_1(\vec{x}_n) & .. & \phi_b(\vec{x}_n) \end{pmatrix}$$

Die Größe der Design Matrix als Ausgangspunkt für SVD/LS Verfahren wächst quadratisch mit der Anzahl der Trainingsdateninstanzen!

- Die Lösung (der Funktionsparameter Θ) geschieht dann doch Matrixinversion der Designmatrix Φ:

$$\hat{\Theta}_{LS} = \left(\Phi^T \Phi\right)^{-1} \Phi^T y$$
$$\hat{\Theta}_{LS} = \Phi^G y$$

mit Φ^G als generalisierte Inverse der Matrix Φ

Die generalisierte Inverse wird dann mit dem SVD Verfahren mit sogenannten links- und rechtssingulären Vektoren bestimmt.

Vertiefung: M. Sugiyama, Introduction to Statistical Machine Learning. 2016, Kapitel 22.2

Support Vector Machines (SVM)

Obwohl die SVM zu den linearen (oder nichtlinearen) Regressionsverfahren gehören, wird die SVM primär für die binäre Klassifikation eingesetzt!

- SVM Verfahren gehören zu den "Maximum Margin" Methoden

Binärer Klassifikator

- Ein binärer Klassifikator soll durch eine **lineare Funktion** repräsentiert werden (y={-1,+1}):

$$f(\vec{x}) : \vec{x} \rightarrow y = \vec{w}^T \vec{x} + \gamma$$

Dabei sind ***w*** und γ die Parameter des Modells die durch das Training an das Problem angepasst werden müssen.

- ***w*** ist ein Normalenvektor der bei einem binären Klassifikationsproblem die beiden Instanzklassen trennt

[4]

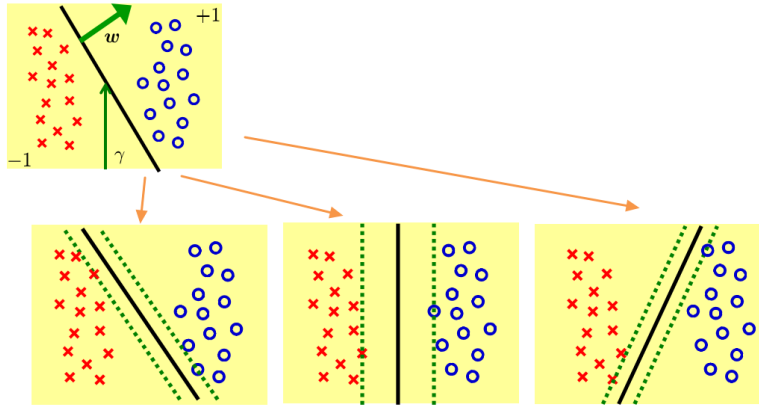


Abb. 2. (Oben) $\boldsymbol{w}$ ist der Normalenvektor und γ die Verschiebung der Trennungsgrenze für zwei Klasseninstanzen (Unten) Verschiedene $\boldsymbol{w}/\gamma$ Varianten der Trennungsgrenze mit unterschiedlichen Rändern (Sicherheitsbereichen)

Training

- Das Lernen von $\boldsymbol{w}$ und γ erfordert die Berechnung des Abstandes von allen Dateninstanzen $(\boldsymbol{x},y)_i$ von der Trenngrenze. Die Abstände müssen positiv sein:

$$f(\vec{x}_i)y_i = (\vec{w}^T\vec{x}_i + \gamma)y_i > 0, \forall i$$

für alle Dateninstanzen $\boldsymbol{D}=\{(\boldsymbol{x}_i,y_i)\}^n$.

- Da $\boldsymbol{w}$ und γ beliebig gewählt werden können, kann die Randbedingung auch mit $(..)y_i \geq 1$ gewählt werden.

- Weiterhin kann es sinnvoll sein alle Dateninstanzen um den Ursprung des Koordinatensystems zu **zentrieren**

Wichtig: Die Werte für *y* liegen im Intervall [-1,1]!

- Alle Probleme die (..)$y_i \geq 1$ erfüllen mit einem (***w***,γ) sind linear separierbar.
- Es gibt unendlich viele Lösungen (also Entscheidungsgrenzen)
- Man wählt das (***w***,γ) aus bei der alle Dateninstanzen die größte Trennung besitzen (breitester Trennbereich, siehe Abb.)
- Der Abstand der Dateninstanzen ***D*** ist definiert als das Minimum des normalisierten Abstandes:

$$m_i = (\vec{w}^T \vec{x}_i + \gamma) y_i / ||\vec{w}||$$

- D.h. SVM zu trainieren ist das Minimierungsproblem zu lösen:

$$\min_i \frac{(\vec{w}^T \vec{x}_i + \gamma) y_i}{||w||} = \frac{1}{||w||}$$

Vertiefung: M. Sugiyama, Introduction to Statistical Machine Learning. 2016., Kapitel 27

- Jede Dateninstanz die nicht in den Trennbereich "eindringt" ist ein Supportvektor!

Harter Trennungsbereich (Hard SVM)

- Bei einer "harten" Trennung einer SVM gilt dann:

$$\min_i 1/2||w||^2, (\vec{w}^T\vec{x}_i + \gamma)y_i \geqslant 1, \forall i$$

Hier wird aber keine Lösung für ***w*** und γ gefunden wenn das Problem nicht strikt linear separierbar ist (also keine einzige Gerade die Klassen trennen kann)

Weicher Trennungsbereich (Soft SVM)

- Die SVM mit harten Trennungsbereich erfordert lineare Separierbarkeit der Dateninstanzen
  - Ist in der Realität aber nicht immer oder eher selten gegeben
- Die weiche Trennung durch eine SVM führt einen Fehlerparametervektor $\xi=\{\xi_i\}^n$ für die Bestimmung des Trennbereichs ein:

$$\min_{\forall i: w,\xi,\gamma} \left[1/2||w||^2 + C\sum_i \xi_i \right],$$
$$(\vec{w}^T\vec{x}_i + \gamma)y_i \geqslant 1 - \xi_i, \xi_i \geqslant 0, \forall i$$

- Sie lässt einzelne nicht (linear) separierbare Datenpunkte zu.

[4]

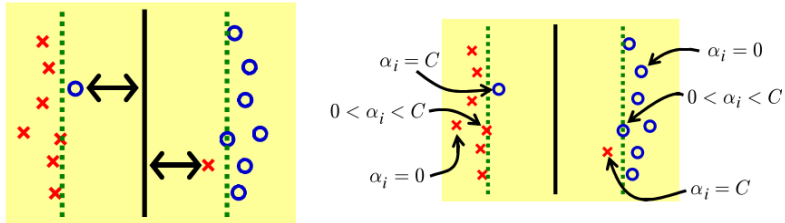


Abb. 3. Weicher Trennbereich einer SVM (Soft margin SVM). Durch ξ werden kleine Klassifikationsfehlerbereiche erlaubt.

Die Ausreißer können durch Rauschen und Messunsicherheit (random. und systematischer Fehler) aber auch aufgrund eines nichtlinear separierbaren Problems entstehen!

Dabei ist C ein einstellbarer Parameter der den Fehler steuert und für den gilt:

$$C = \alpha_i + \beta_i$$

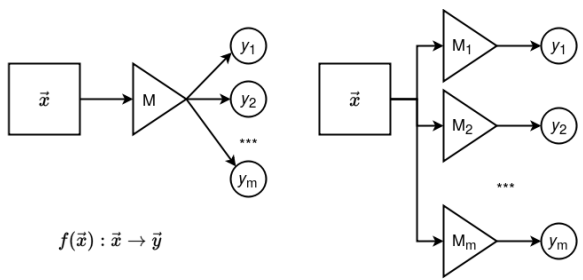
Größere C Werte machen den Abstandsfehler kleiner und für große C geht die weiche in eine harte SVM über

- $\alpha_i = 0$ impliziert $m_i \geq 1$: die i-te Trainingsinstanz x_i ist auf der Margengrenze oder innerhalb der Marge und ist korrekt klassifiziert.
- $\alpha_i = C$ impliziert $m_i \leq 1$: x_i ist auf der Margengrenze oder außerhalb der Marge. Wenn $\xi_i > 1$, $m_i < 0$ dann ist x_i falsch klassifiziert.
- $0 < \alpha_i < C$ impliziert $m_i = 1$: x_i ist auf der Margengrenze und ist korrekt klassifiziert.
- $m_i > 1$ impliziert $\alpha_i = 0$: wenn x_i innerhalb der Marge ist, $\alpha_i = 0$.
- $m_i < 1$ impliziert $\alpha_i = C$: wenn x_i außerhalb der Marge ist, $\alpha_i = C$.

Multiklassen SVM

Jedes Multiklassenproblem mit *m* verschiedenen (diskreten) Klassenwerten kann auf *m* binäre Klassifikationsprobleme transformiert werden

- Anders als bei ANN ist bei SVMs aber nur eine One-hot Kodierung möglich (ggfs. mit Softmaxfunktion).



ML Framework API für SVM

```
ML.learn({
    algorithm:ML.ML.SVM,
    x:number [][],
    y:number [],
    // threshold function on output? highest value of multi-svms is winner
    threshold:boolean,
    // default : 1.0. C in SVM.
    C : number,
    // default : 1e-4. Higher tolerance --> Higher precision
    tol : number,
    // default : 20. Higher max_passes --> Higher precision
    max_passes : number,
    // default : 1e-5. Higher alpha_tolerance --> Higher precision
    alpha_tol : number,
    kernel : { type: 'rbf', sigma: 0.5 }
    // { type: "polynomial", c: 1, d: 5}
}) → model
```

Beispiel

Web WorkShell Live

Zusammenfassung

[Sugiyama, ItSML, pp 303]

- Regressionsverfahren werden auf kontinuierliche Zielvariablen angewendet (der Hypothesenraum kann bei nichtlinearen Problemen sehr groß werden)
- Eine SVM wird als binärer Klassifikator verwendet und wird i.A. durch eine lineare Funktion (Kernel) repräsentiert
  - Das Problem sollte dann linear separierbar sein!
- Multiklassenprobleme werden auf Multi-SVMs zurückgeführt
  - Verwendung einer Softmax Funktion für eindeutige Klassentrennung
- Das Training einer SVM ist ein Minimierungsproblem dass den Trennbereich maximiert und den Fehler minimiert