



Vorlesung Wissensentdeckung in Datenbanken

Klassifikation und Regression

Katharina Morik, Uwe Ligges

LS 8 Künstliche Intelligenz Fakultät für Informatik
Computergestützte Statistik
Technische Universität Dortmund

7.5.2013



Gliederung

1 Lineare Modelle zur Klassifikation und Regression

- Klassifikation und Regression
- Lineare Modelle

2 Bias-Varianz

- Exkurs: Erwartungswert
- Bias und Varianz bei linearen Modellen

3 kNN zur Klassifikation, Regression

- Bias und Varianz bei kNN



Gliederung

- 1 **Lineare Modelle zur Klassifikation und Regression**
 - Klassifikation und Regression
 - Lineare Modelle
- 2 **Bias-Varianz**
 - Exkurs: Erwartungswert
 - Bias und Varianz bei linearen Modellen
- 3 **kNN zur Klassifikation, Regression**
 - Bias und Varianz bei kNN



Gliederung

- 1 **Lineare Modelle zur Klassifikation und Regression**
 - Klassifikation und Regression
 - Lineare Modelle
- 2 **Bias-Varianz**
 - Exkurs: Erwartungswert
 - Bias und Varianz bei linearen Modellen
- 3 **kNN zur Klassifikation, Regression**
 - Bias und Varianz bei kNN



Grundlagen

Sei $X = \{X_1, \dots, X_p\}$ eine Menge von Zufallsvariablen und $Y \neq \emptyset$ eine Menge.

Ein **Beispiel** (oder *Beobachtung*) \vec{x} ist ein konkreter p -dimensionaler Vektor über diese Zufallsvariablen.

Eine **Menge von n Beispielen** $\mathbf{X} = \{\vec{x}_1, \dots, \vec{x}_N\}$ können wir dann als $(N \times p)$ -Matrix auffassen:

$$\mathbf{X} = \begin{pmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,p} \\ x_{2,1} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ x_{N,1} & x_{N,2} & \dots & x_{N,p} \end{pmatrix}$$

Dabei entspricht jede Zeile \vec{x}_i der Matrix \mathbf{X} einem Beispiel.



Klassifikation und Regression

Beim *überwachten Lernen* (darum geht es hier), ist zusätzlich zu jeder Beobachtung \vec{x} ein *Label (Klasse)* y gegeben, d.h. wir haben Beobachtungen $(\vec{x}, y) \in X \times Y$.

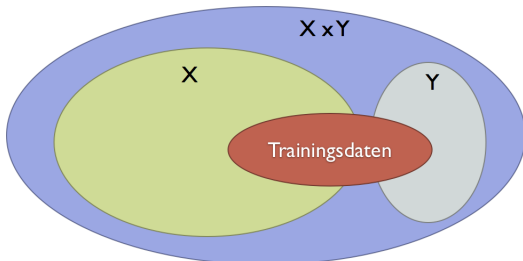
Y kann sowohl eine **qualitative**, als auch eine **quantitative** Beschreibung von \vec{x} sein.

Für den quantitativen Fall ist z.B. $Y = \mathbb{R}$ und wir versuchen für unbekanntes \vec{x} den Wert y vorherzusagen **Regression**.

Im Falle qualitativer Beschreibungen ist Y eine diskrete Menge und wir nutzen f zur **Klassifikation**.

Lernen auf Trainingsdaten

Wovon gehen wir also aus? Was ist unser Ziel?



- Wir suchen *die wahre Funktion* $f : X \rightarrow Y$ mit

$$f(\vec{x}) = y \quad \forall (\vec{x}, y) \in X \times Y$$

- Wir haben jedoch nur eine Teilmenge der Beobachtungen gegeben (Trainingsdaten)



Klassifikation und Regression

Auf Grundlage der Trainingsdaten suchen wir eine möglichst gute Annäherung \hat{f} an die *wahre Funktion* f .

Die Funktion \hat{f} bezeichnen wir auch als das gelernte **Modell**.

Haben wir ein Modell \hat{f} gelernt, so liefert uns dieses Modell mit

$$\hat{y} = \hat{f}(\vec{x})$$

für *neue Daten* $\vec{x} \in X$ eine Vorhersage $\hat{y} \in Y$.



Klassifikation und Regression

Im Falle der *Regression* lässt sich so für zuvor unbekannte $\vec{x} \in X$ der Wert

$$\hat{y} = \hat{f}(\vec{x})$$

mit $\hat{y} \in \mathbb{R}$ vorhersagen.

Dieses Modell \hat{f} lässt sich auch für die Klassifikation nutzen, bei der z.B. $\hat{y} \in \{-1, +1\}$ vorhergesagt werden sollen:

$$\hat{y} = \begin{cases} +1, & \text{falls } \hat{f}(\vec{x}) \geq \theta \\ -1, & \text{sonst} \end{cases}$$

Hier ist θ ein vorgegebener Schwellwert.



Beispiel

Gegeben seien Gewicht (X_1) und Größe (X_2) einiger Personen und ein Label $y \in \{m, w\}$:

	X_1	X_2	Y
x_1	91	190	m
x_2	60	170	w
x_3	41	160	w
\vdots	\vdots	\vdots	\vdots

Die Tabelle enthält die zur Verfügung stehenden Trainingsdaten, also

$$\mathbf{X} = \begin{pmatrix} 91 & 190 \\ 60 & 170 \\ 41 & 160 \\ \vdots & \vdots \end{pmatrix}$$



Beispiel

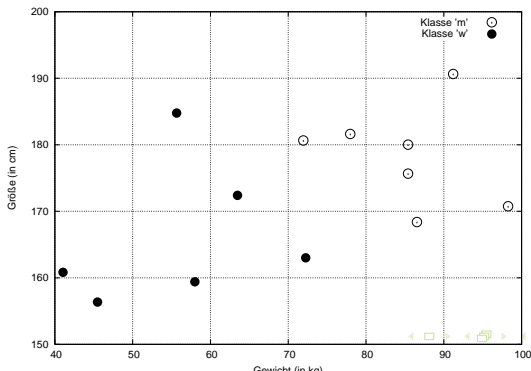
Es wird nun eine Funktion \hat{f} gesucht, die für neue Daten \vec{x} das Attribut Y (Geschlecht) voraussagt, also

$$\hat{y} = \begin{cases} m, & \text{falls } \hat{f}(x) > \theta \\ w, & \text{sonst} \end{cases}$$

Beispiel

Es wird nun eine Funktion \hat{f} gesucht, die für neue Daten \vec{x} das Attribut Y (Geschlecht) voraussagt, also

$$\hat{y} = \begin{cases} m, & \text{falls } \hat{f}(x) > \theta \\ w, & \text{sonst} \end{cases}$$





Lineare Modelle

Welche Art von Funktionen sind denkbar?

Lineare Funktionen als einfachste Funktionenklasse:

$$y = f(x) = mx + b \quad \text{Gerade im } \mathbb{R}^2$$

Allerdings betrachten wir als Beispielraum den \mathbb{R}^p , d.h. wir brauchen eine verallgemeinerte Form:

$$y = f(\vec{x}) = \sum_{i=1}^p \beta_i x_i + \beta_0 \quad \text{mit } \beta_0 \in \mathbb{R}, \vec{x}, \vec{\beta} \in \mathbb{R}^p \quad (1)$$

Die Funktion f wird also durch $\vec{\beta}$ und β_0 festgelegt und sagt uns für ein gegebenes \vec{x} das entsprechende y voraus



Notation, Vereinbarungen

Bei genauerer Betrachtung von Formel (1) lässt sich $\sum_{i=1}^p \beta_i x_i$ als Matrizenmultiplikation oder Skalarprodukt schreiben, also

$$y = \sum_{i=1}^p \beta_i x_i + \beta_0 = \vec{x}^T \vec{\beta} + \beta_0 = \langle \vec{x}, \vec{\beta} \rangle + \beta_0$$

Zur einfacheren Darstellung von f , wird β_0 in den Vektor $\vec{\beta}$ codiert, indem jedes Beispiel $x = (x_1, \dots, x_p)$ aufgefasst wird als $(p + 1)$ -dimensionaler Vektor

$$(x_1, \dots, x_p) \mapsto (1, x_1, \dots, x_p)$$

Dies ermöglicht die Darstellung von f als:

$$y = f(\vec{x}) = \sum_{i=0}^p \beta_i x_i = \vec{x}^T \vec{\beta} = \langle \vec{x}, \vec{\beta} \rangle$$



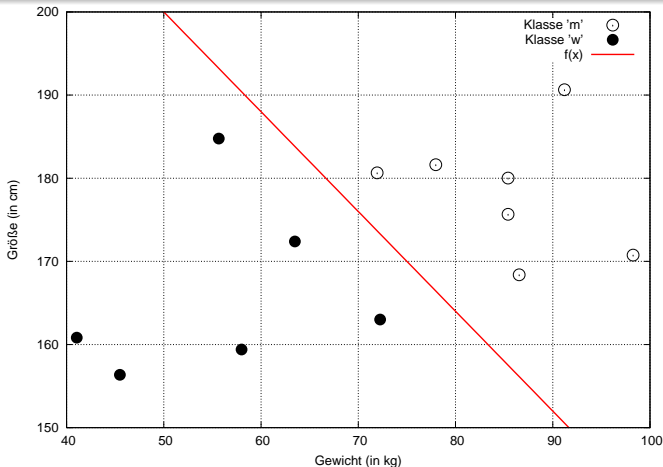
Was haben wir nun gemacht?

Wir haben (bei der Beschränkung auf lineare Modelle) nun eine Darstellung für das, was wir *lernen* wollen:

$$y = \hat{f}(\vec{x}) = \vec{x}^T \vec{\beta}$$

Wir haben die Zielfunktion \hat{f} in Abhängigkeit von $\vec{\beta}$ geschrieben und müssen *nur noch* das passende $\vec{\beta}$ finden.

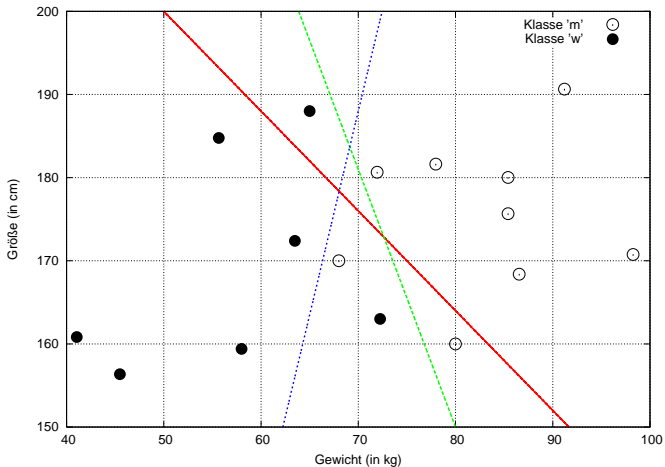
Beispiel: Ein mögliches $\vec{\beta}$



$$f(\vec{x}) = \vec{x}^T \hat{\vec{\beta}} \quad \text{mit} \quad \hat{\vec{\beta}} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix} = \begin{pmatrix} 260 \\ 1 \\ 1.2 \end{pmatrix} \theta = 550$$



Es ist nicht garantiert, dass $\vec{\beta}$ immer passt!





Modell-Anpassung

Unsere linearen Modelle sind durch $\vec{\beta}$ parametrisiert, das Lernen eines Modells haben wir also auf die Wahl eines $\vec{\beta}$ abgewälzt.

Das wirft eine Reihe von Fragen auf:

- Was ist ein gutes $\vec{\beta}$?
- Gibt es ein optimales $\vec{\beta}$?
- Welche Möglichkeiten haben wir, unser Modell zu beurteilen?



Modell-Anpassung

Unsere linearen Modelle sind durch $\vec{\beta}$ parametrisiert, das Lernen eines Modells haben wir also auf die Wahl eines $\vec{\beta}$ abgewälzt.

Das wirft eine Reihe von Fragen auf:

- Was ist ein gutes $\vec{\beta}$?
- Gibt es ein optimales $\vec{\beta}$?
- Welche Möglichkeiten haben wir, unser Modell zu beurteilen?

Eine Möglichkeit: Berechne den *Trainingsfehler*

$$Err(\vec{\beta}) = \sum_{i=1}^N |y_i - \hat{f}(\vec{x}_i)| = \sum_{i=1}^N |y_i - x_i^T \vec{\beta}|$$



Modell-Anpassung

Häufig wird als Fehlerfunktion die *quadratische Fehlersumme* (RSS) verwendet:

$$\begin{aligned}RSS(\vec{\beta}) &= \sum_{i=1}^N (y_i - \vec{x}_i^T \vec{\beta})^2 \\ &= (\vec{y} - \mathbf{X}\vec{\beta})^T (\vec{y} - \mathbf{X}\vec{\beta})\end{aligned}$$

Wir wählen jetzt $\vec{\beta}$ derart, dass der Fehler minimiert wird:

$$\min_{\vec{\beta} \in \mathbb{R}^p} RSS(\vec{\beta}) \quad (2)$$

⇒ Konvexes Minimierungsproblem!



Minimierung von $RSS(\vec{\beta})$

Um $RSS(\vec{\beta})$ zu minimieren, bilden wir die partielle Ableitung nach $\vec{\beta}$:

$$\frac{\partial RSS(\vec{\beta})}{\partial \beta} = \mathbf{X}^T (\mathbf{y} - \mathbf{X}\vec{\beta})$$

Notwendige Bedingung für die Existenz eines (lokalen) Minimums von RSS ist

$$\frac{\partial RSS(\vec{\beta})}{\partial \beta} = \mathbf{X}^T (\mathbf{y} - \mathbf{X}\vec{\beta}) = 0$$

Ist $\mathbf{X}^T \mathbf{X}$ regulär, so erhalten wir

$$\hat{\vec{\beta}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (3)$$



Reguläre Matrix

Wenn es zu einer quadratischen Matrix \mathbf{X} eine Matrix \mathbf{X}^{-1} gibt mit

$$\mathbf{X}\mathbf{X}^{-1} = \mathbf{X}^{-1}\mathbf{X} = \mathbf{I}$$

Einheitsmatrix

$$I = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \cdot & \cdot & \dots & 0 \\ \cdot & \cdot & \dots & 0 \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

dann ist die Matrix \mathbf{X} invertierbar oder *regulär*, sonst *singulär*.

Optimales $\hat{\vec{\beta}}$?

Mit Hilfe der Minimierung der (quadratischen) Fehlerfunktion RSS auf unseren Trainingsdaten haben wir ein (bzgl. RSS) optimales $\hat{\vec{\beta}}$ gefunden.

Bei einem konvexen Problem ist das lokale auch das globale Minimum.

Damit liefert unser Modell Voraussagen \hat{y} für $\vec{x} \in X$:

$$\hat{y} = \hat{f}(\vec{x}) = \vec{x}^T \hat{\vec{\beta}}$$



Sind wir schon fertig?

- Schön wär's!
- Aber drei Gründe sprechen für weitere Arbeit:
 - 1 Es ist nicht immer so einfach, z.B. dann nicht, wenn wir viele Dimensionen haben (Fluch der hohen Dimension).
 - 2 Vielleicht lassen sich die Beispiele nicht linear trennen!
 - 3 Nur den Fehler zu minimieren reicht nicht aus, wir suchen noch nach weiteren Beschränkungen, die zu besseren Lösungen führen.
- Also schauen wir uns den Fehler noch einmal genauer an, stoßen auf Bias und Varianz und merken, dass wir noch keine perfekte Lösung haben.



Fehler

- Bisher haben wir mit RSS die Fehler einfach summiert.
- Wir wollen aber einbeziehen, wie wahrscheinlich der Fehler ist – vielleicht ist er ja ganz unwahrscheinlich! Das machen wir über den **Erwartungswert**.
- Wir können sehr unterschiedliche Stichproben als Beispielmengen haben. Der Fehler soll sich auf **alle möglichen Trainingsmengen** beziehen – nicht nur eine, zufällig günstige!



Zur Erinnerung: Erwartungswert

Erwartungswert

Sei X eine **diskrete Zufallsvariable**, mit Werten x_1, \dots, x_n und p_i die Wahrscheinlichkeit für x_i . Der Erwartungswert von X ist

$$E(X) = \sum_i x_i p_i = \sum_i x_i P(X = x_i)$$

Ist X eine **stetige Zufallsvariable** und f die zugehörige Wahrscheinlichkeitsdichtefunktion, so ist der Erwartungswert von X

$$E(X) = \int_{-\infty}^{\infty} x f(x) dx$$



Erwartungswert (Eigenschaften)

Eigenschaften

Seien X, Y und X_1, \dots, X_n Zufallsvariablen, dann gilt:

- Der Erwartungswert ist additiv, d.h. es gilt

$$E\left(\sum_{i=1}^n X_i\right) = \sum_{i=1}^n E(X_i) \quad (4)$$

- Ist $Y = kX + d$, so gilt für den Erwartungswert

$$E(Y) = E(kX + d) = kE(X) + d \quad (5)$$

- Sind die Zufallsvariablen X_i **stochastisch unabhängig**, gilt

$$E\left(\prod_{i=1}^n X_i\right) = \prod_{i=1}^n E(X_i)$$



Varianz und Standardabweichung

Über den Erwartungswert einer Zufallsvariablen X sind mehrere Eigenschaften von X definiert, die helfen, X zu charakterisieren:

Varianz

Sei X eine Zufallsvariable mit $\mu = E(X)$. Die **Varianz** $Var(X)$ ist definiert als

$$Var(X) := E((X - \mu)^2).$$

Die Varianz wird häufig auch mit σ^2 bezeichnet.

Standardabweichung

Die **Standardabweichung** σ einer Zufallsvariable X ist definiert als

$$\sigma := \sqrt{Var(X)}$$



Varianz und Standardabweichung

Verschiebungssatz

Sei X eine Zufallsvariable, für die Varianz gilt

$$\text{Var}(X) = E(X - E(X))^2 = E(X^2) - (E(X))^2$$



Bias

Eine weitere Charakteristik, die häufig zur Beschreibung von erwarteten Fehlern verwendet wird, ist die Verzerrung:

Verzerrung (Bias)

Sei Y eine Zufallsvariable, dann ist die Verzerrung definiert als der erwartete Schätzfehler für Y
also wie im Durchschnitt die Schätzungen vom wahren Mittelwert abweichen

$$\text{Bias}(\hat{y}) = E(Y - \hat{y}) = E(Y) - \hat{y}$$



Fehler der Regression

- Fehlerfunktion $L(y, \hat{y})$ für gelernte Modelle \hat{f}
 - absolut $\sum (y_i - \hat{y}_i)$
 - quadratisch $\sum (y_i - \hat{y}_i)^2$
 - 0,1-Fehler $\sum \delta_i$, $\delta = 1$, falls $y \neq \hat{y}$, sonst 0.
- Es geht um Y . Wir unterscheiden
 - das wahre y ,
 - das in der Beispielmenge genannte y ,
 - das vom Modell vorhergesagte \hat{y}
- Wir wollen den Erwartungswert des Fehlers minimieren.
- Wir mitteln über alle möglichen Beispielmengen \mathcal{T} .



Erwartungswert des Fehlers einer Regression minimieren!

Erwarteter quadratischer Vorhersagefehler: Gelernte Funktion $\hat{f} : X \rightarrow Y$, der Erwartungswert ihres Fehlers ist:

$$EPE(f) = E(Y - \hat{f}(X))^2 \quad (6)$$

Optimierungsproblem: Wähle \hat{f} so, dass der erwartete Fehler minimiert wird!

$$\hat{f}(x) = \operatorname{argmin}_c E_{Y|X}((Y - c)^2 | X = x) \quad (7)$$

Lösung (Regressionsfunktion): $\hat{f}(x) = E(Y | X = x)$



Bias und Varianz

Zwei Aspekte machen den erwarteten Fehler aus, die Verzerrung (Bias) und die Varianz. Wir wollen den Fehler an einem Testpunkt $x_0 = 0$ angeben und mitteln über allen Trainingsmengen \mathcal{T} .

- Wir gehen davon aus, dass die Angaben in den Daten nicht immer ganz stimmen, so dass es einen Messfehler ϵ gibt, dessen Erwartungswert aber 0 ist.
- Der Bias ist unabhängig vom Beispielsatz und 0 bei einem perfekten Lerner.
- Die Varianz ist unabhängig vom wahren Wert y und 0 bei einem Lerner, der bei allen Beispielsätzen dasselbe ausgibt.



MSE Dekomposition in Bias und Varianz

Wir nehmen für unser Modell an,
dass $Y = f(x) + \epsilon$ und $E(\epsilon) = 0$.

$$\begin{aligned}
 EPE(x_0) &= E_{Y,\mathcal{T}}((Y - \hat{y}_0)^2|x_0) \\
 &= E_Y((Y - f(x_0))^2|x_0) + \sigma^2 \text{Rauschen} \\
 &\quad E_{\mathcal{T}}((f(x_0) - E_{\mathcal{T}}(\hat{y}_0))^2|x_0) + \text{Bias}^2 \\
 &\quad E_{\mathcal{T}}((E_{\mathcal{T}}(\hat{y}_0) - \hat{y}_0)^2|x_0) \quad \text{Varianz}
 \end{aligned}$$

Wie das?!

Haupttrick: kreatives Einfügen von Termen, $+a - a$, die nichts ändern, aber Umformungen erlauben.



Herleitung der Dekomposition - 1

$$\begin{aligned}
 EPE(x_0) &= E_{Y,\mathcal{T}}((Y - \hat{y}_0)^2|x_0) \\
 &= E_{Y,\mathcal{T}}(((Y - f(x_0)) + (f(x_0) - \hat{y}_0))^2|x_0) && \textit{kreativ} \\
 &= E_{Y,\mathcal{T}}((Y - f(x_0))^2 + (f(x_0) - \hat{y}_0)^2 + && \textit{binomisch} \\
 &\quad 2(Y - f(x_0))(f(x_0) - \hat{y}_0)|x_0) \\
 &= E_Y((Y - f(x_0))^2|x_0) + && \textit{herausziehen} \\
 &\quad E_{Y,\mathcal{T}}((f(x_0) - \hat{y}_0)^2|x_0) + && \textit{s.Formel(4)} \\
 &\quad 2E_Y(Y - f(x_0)|x_0)E_{Y,\mathcal{T}}(f(x_0) - \hat{y}_0|x_0) && E(2) = 2
 \end{aligned}$$

Jetzt $E_Y(Y - f(x_0)|x_0) = E_Y(Y|x_0) - f(x_0) = 0$ wegen der Modellannahme $E_Y(Y|x_0) = f(x_0)$.



Herleitung der Dekomposition - 2

$$\begin{aligned}
 EPE(x_0) &= E_Y((Y - f(x_0))^2|x_0) + Var_\epsilon \\
 &= E_{Y,\mathcal{T}}((f(x_0) - \hat{y}_0)^2|x_0) \\
 &= Var_\epsilon + E_{Y,\mathcal{T}}((f(x_0) - \hat{y}_0)^2|x_0) \\
 &= Var_\epsilon + E_{Y,\mathcal{T}}(((f(x_0) - E_{Y,\mathcal{T}}(\hat{y}_0)) + \\
 &\quad (E_{Y,\mathcal{T}}(\hat{y}_0) - \hat{y}_0))^2|x_0) \quad \textit{kreativ} \\
 &= Var_\epsilon + E_{Y,\mathcal{T}}((f(x_0) - E_{Y,\mathcal{T}}(\hat{y}_0))^2 + \\
 &\quad (E_{Y,\mathcal{T}}(\hat{y}_0) - \hat{y}_0)^2 + \\
 &\quad 2(f(x_0) - E_{Y,\mathcal{T}}(\hat{y}_0))(E_{Y,\mathcal{T}}(\hat{y}_0) - \hat{y}_0|x_0)) \\
 &= Var_\epsilon + E_{Y,\mathcal{T}}((f(x_0) - E_{Y,\mathcal{T}}(\hat{y}_0))^2|x_0) + \textit{herausziehen} \\
 &\quad E_{Y,\mathcal{T}}((E_{Y,\mathcal{T}}(\hat{y}_0) - \hat{y}_0)^2|x_0) + (4) \\
 &\quad 2E_{Y,\mathcal{T}}((f(x_0) - E_{Y,\mathcal{T}}(\hat{y}_0)) \\
 &\quad (E_{Y,\mathcal{T}}(\hat{y}_0) - \hat{y}_0)|x_0)
 \end{aligned}$$



Bias und Varianz bei linearen Modellen

Das lineare Modell wird an die Daten angepasst durch

$$\hat{f}_p(\vec{x}) = \hat{\beta}^T \vec{x}$$

Der Fehler ist dann für ein beliebiges \vec{x} :

$$Errr(\vec{x}_0) = E[(Y - \hat{f}_p(\vec{x}_0))^2 | X = \vec{x}_0] \quad (8)$$

$$= \sigma_\epsilon^2 + Var(\hat{f}_p(\vec{x}_0)) + \left[f(\vec{x}_0) - E\hat{f}_p(\vec{x}_0) \right]^2 \quad (9)$$

Die Anpassung des linearen Modells geht über alle N Beispiele und gewichtet alle p Merkmale (s. (3)).

Diese Varianz ist von x_i zu x_i verschieden. Im Mittel über allen \vec{x}_i ist $Var(\hat{f}_p) = (p/N)\sigma_\epsilon^2$.



Zusammenhang zwischen Anzahl der Beispiele, der Attribute und erwartetem Fehler

Modellkomplexität (p, N) und Varianz der Schätzungen bei unterschiedlichen Trainingsmengen hängen bei linearen Modellen direkt zusammen.

Gemittelt über alle x_i ist der Trainingsfehler linearer Modelle:

$$\frac{1}{N} \sum_{i=1}^N Errr(x_i) = \sigma_\epsilon^2 + \frac{p}{N} \sigma_\epsilon^2 + \frac{1}{N} \sum_{i=1}^N \left[f(\vec{x}_i) - E\hat{f}(\vec{x}_i) \right]^2 \quad (10)$$

Wir haben also wieder das Rauschen, die Varianz, die die Schwankungen der Schätzungen angibt, und den Bias, der sich auf die Differenz von Schätzung und Wahrheit bezieht (in-sample error).



Fluch der hohen Dimension bei linearen Modellen

- Leider mussten wir annehmen, dass das Modell genau passt, um den erwarteten Fehler klein zu halten.
- Wir wissen aber nicht, welche Art von Funktion gut zu unseren Daten passt! **Modellselektion** ist schwierig!
- Das Modell muss immer komplizierter werden, je mehr Dimensionen es gibt.
- Bei linearen Modellen entspricht die Komplexität des Modells direkt p , denn β hat so viele Komponenten wie p bzw. $p + 1$.

Lineare Modelle

Die grünen und roten Datenpunkte werden durch eine Ebene getrennt.

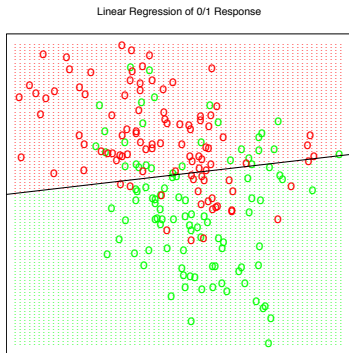


Figure 2.1: A classification example in two dimensions. The classes are coded as a binary variable—**GREEN** = 0, **RED** = 1—and then fit by linear regression. The line is the decision boundary defined by $x^T \hat{\beta} = 0.5$.



Was wissen Sie jetzt?

- Sie haben theoretisch lineare Modelle für Klassifikation und Regression kennengelernt.
- Sie kennen das **Optimierungsproblem** der kleinsten Quadrate RSS (Gleichung 2) für lineare Modelle (Gleichung 3).
- Sie kennen den erwarteten Fehler EPE bei linearen Modellen (Gleichung 6).
- Sie kennen den **Fluch der hohen Dimension** bei linearen Modellen: Komplexität und Varianz hängen an der Dimension! Der Bias kann sehr hoch sein, wenn die Beispiele tatsächlich nicht linear separierbar sind.



Bis zum nächsten Mal...

- Gehen Sie alle Folien noch einmal in Ruhe durch.
- Rechnen Sie mal ein Beispiel durch mit Gleichung zur Optimierung linearer Modelle (3), der Minimierung des Trainingsfehlers (10)...
- Diskutieren Sie, warum Bias und Varianz so wichtig sind!
- Probieren Sie lineare Regression in RapidMiner aus!



Globale und lokale Modelle

- Lineare Modelle finden eine trennende Hyperebene.
- Die durch $\vec{\beta}$ angegebene Hyperebene wurde durch **alle** Beispiele bestimmt.
- Deshalb sind lineare Modelle **globale Modelle**.
- Klassifiziert man ein Beispiel nur anhand der Beispiele seiner **Umgebung**, spricht man von einem **lokalen Modell**.
- Nächste Nachbarn sind ein lokales Modell.



Nächste Nachbarn

- Das k NN-Modell betrachtet nur noch die k nächsten Nachbarn eines Beispiel \vec{x} :

$$\hat{f}(\vec{x}) = \frac{1}{k} \sum_{\vec{x}_i \in N_k(\vec{x})} y_i \quad (11)$$

- Die Nachbarschaft $N_k(\vec{x})$ wird durch ein Abstandsmaß, z.B. den Euklidischen Abstand bestimmt.
- Es gibt maximal $\frac{N}{k}$ Nachbarschaften und in jeder bestimmen wir den Durchschnitt (11).



Regression und Klassifikation

Gleichung (11) gibt als Regressionsfunktion den Mittelwert der y_i zurück.

$$\hat{f}(\vec{x}) = \frac{1}{k} \sum_{\vec{x}_i \in N_k(\vec{x})} y_i$$

Wir können durch einen Schwellwert aus der Regression eine Klassifikation machen:

$$\hat{y} = \begin{cases} 1, & \text{falls } \hat{f}(\vec{x}) \geq 0,5 \\ 0, & \text{sonst} \end{cases}$$



Die grünen und roten Datenpunkte werden in Nachbarschaften gruppiert

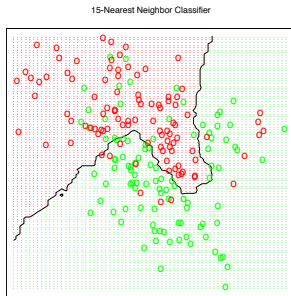


Figure 2.2: *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable (GREEN = 0, RED = 1) and then fit by 15-nearest-neighbor averaging as in (2.8). The predicted class is hence chosen by majority vote amongst the 15-nearest neighbors.*



Bei $k=1$ wird nur auswendig gelernt.

- Falls $\vec{x} = \vec{x}' \rightarrow y = y'$, gibt es bei $k = 1$ keinen Trainingsfehler.
- Wenn allein der Trainingsfehler das Optimierungskriterium ist, würden wir stets $k = 1$ nehmen und nur auswendig lernen.
- Vermutlich ergibt das auf den Testdaten einen großen Fehler!



Overfitting

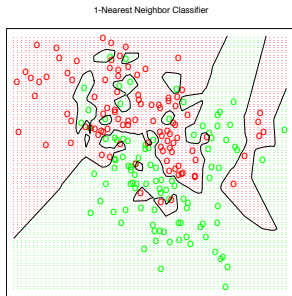


Figure 2.3: *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable (GREEN = 0, RED = 1), and then predicted by 1-nearest-neighbor classification.*

Training- und Testfehler bei verschiedenen k

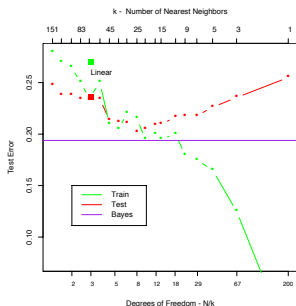


Figure 2.4: *Misclassification curves for the simulation example used in Figures 2.1, 2.2 and 2.3. A single training sample of size 200 was used, and a test sample of size 10,000. The red curves are test and the green are training error for k -nearest-neighbor classification. The results for linear regression are the bigger green and red dots at three degrees of freedom. The purple line is the optimal Bayes Error Rate.*



Erwartungswert von Y bei k Nächsten Nachbarn

- Der **Erwartungswert** von Y , $E(Y) = \sum_{i=1}^N y_i p_i$, geht bei linearen Modellen in den Fehler ein:
$$EPE(\hat{f}) = E(Y - \hat{f}(X))^2.$$
- k NN verwendet den Erwartungswert von Y direkt zur Vorhersage, allerdings beschränkt auf die Nachbarschaft
$$E(Y) = \frac{1}{k} \sum_{\vec{x}_i \in N_k(\vec{x})} y_i.$$
- Für die Vorhersage sind wir an bedingten Wahrscheinlichkeiten interessiert $P(Y|X = \vec{x})$.
- Bei k NN wird die bedingte Wahrscheinlichkeit auf die Nachbarschaft begrenzt $E(Y|\vec{x}_i \in N_k(\vec{x}))$.
- Gerechnet wird dies mit Hilfe Gleichung (11).



Asymptotisches Ergebnis zu k NN

- Wenn k/N gegen 0 und N, k gegen ∞ konvergieren, konvergiert auch $\hat{f}(x)$ gegen $E(Y|X = x)$. (Hastie/etal/2001, S. 19)
- Haben wir also schon (wieder) den perfekten Lernalgorithmus gefunden?



Fluch der hohen Dimension bei k NN

- Die Dichte der Beispiele ist proportional zu $N^{\frac{1}{p}}$.
- Schon bei $p = 10$ brauchen wir 80% der möglichen Werte jedes Attributs X_i , um wenigstens 10% der Daten in einer Nachbarschaft gesehen zu haben!
- Die Dichte der Datenpunkte in der Nachbarschaft ist bei hoher Dimension furchtbar spärlich.
 - $N^{\frac{1}{p}}$ ist bei 100 Beispielen und $p = 10$ nur $100^{1/10} = \sqrt[5]{10}$.
 - Wenn 100 Beispiele bei $p = 1$ einen dichten Raum ergeben, muss man für die selbe Dichte bei $p = 10$ schon 100^{10} Beispiele sammeln: $100^{1/1} = 100$, $100^{10 \cdot \frac{1}{10}} = 100$



Bias und Varianz bei k NN

- Wenn man die richtige, dicht besetzte Nachbarschaft hat, verzerrt k NN die Vorhersage nicht (**kleiner Bias**).
- Wenn - wie bei hohen Dimensionen - die Nachbarschaft wild variiert, schwankt auch die Güte der Vorhersage (**große Varianz**).

Bias und Varianz – bildlich

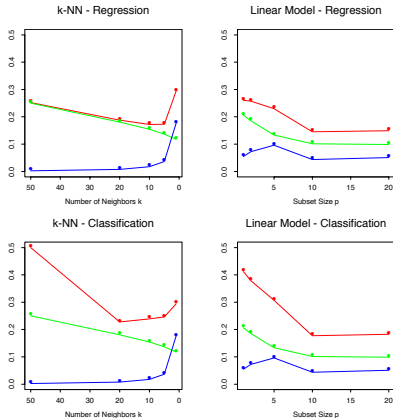


Figure 7.3: Prediction error (red), squared bias (green) and variance (blue) for a simulated example. The top row is regression with squared error loss; the bottom row is classification with 0–1 loss. The models are k-nearest neighbors (left) and best subset regression of size p (right). The variance and bias curves are the same in regression and classi-

Bias, Varianz und Modellkomplexität – bildlich

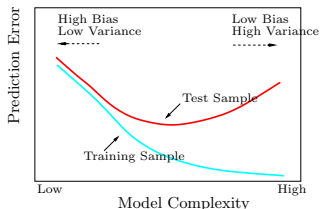


Figure 7.1: Behavior of test sample and training sample error as the model complexity is varied.

- Dieser Zusammenhang von Training-/Testfehler und Komplexität bestimmt alle Lernverfahren.
- Kreuzvalidierung lässt abschätzen, wie gut das Modell zu den Daten passt (Modellselektion).



Was wissen Sie jetzt?

- Sie kennen den Fluch der hohen Dimension bei kNN:
kleiner Bias, aber hohe Varianz.
- Bei linearen Modellen war es umgekehrt: kleine Varianz,
aber hoher Bias (falls die Annahme des linearen
Zusammenhangs von X, Y nicht stimmt).