Projekt 7

Machine Learning

(Matthias Bethge)

7.1 Overtüre

> *Alles messen, was messbar ist - und messbar machen, was noch nicht messbar ist.* *(Galileo Galilei 1564-1642)*

Ziel der quantitativen Wissenschaft ist es, möglichst präzise Zusammenhänge zwischen messbaren Größen herzustellen. Ein bekanntes Beispiel aus der Physik ist der Zusammenhang zwischen Ort, Geschwindigkeit, Beschleunigung und Masse, der durch die Newtonschen Gesetze und die Gravitationskraft beschrieben ist. Auch wenn wir in der Physik davon ausgehen, dass im Prinzip alle Prozesse auf die Wirkung von vier Grundkräften zurückgeführt werden können, so wissen wir bereits aus der statistischen Mechanik, dass makroskopische Systeme neue Gesetzmäßigkeiten aufweisen können, die von den mikroskopischen Eigenschaften weitgehend unabhänging sind. Ein besonders spannendes Beispiel für eine makroskopische Gesetzmäßigkeit, die wir tagtäglich benutzen, aber deren Grundlagen bisher noch wenig verstanden sind, ist die Frage, wie wir Objekte anhand von gestreutem Licht erkennen können, ohne dabei spezifisches Vorwissen über die Lichtquellen zu besitzen. Anders gefragt[1]: "Wie baut man eigentlich ein Messgerät für eine Kuh?" Obwohl die Optik zur klassischen Ausbildung in der Physik gehört und der Sehsinn von Mensch und Tier ein lebender Beweis dafür sind, dass sich Objekte objektiv und reproduzierbar unter verschiedensten Lichtbedingungen und aus verschiedensten Perspektiven detektieren lassen, wird die Frage der Objekterkennung kaum von Physikern untersucht. Vielmehr geschieht dies im sogenannten "Computer Vision"-Feld, welches sich als Teilgebiet der Informatik entwickelt hat. Methodisch ist diese Zuordnung nachvollziehbar, da Computer eine wesentliche Voraussetzung dafür sind, die komplexen Berechnungen durchführen zu können, die für die Detektion von Objekten nötig sind. Der Computer ist dabei aber nur ein Hilfsmittel. Die Fragestellung hingegen entspricht dem quantitativ-empirischen Ansatz, den man als das Markenzeichen der Physik bezeichnen könnte. Statt den Computer nur zur Simulation von bereits aus der Physik bekannten Gesetzmäßigkeiten zu verwenden, soll

[1] Prof. Manfred Kree wirft diese Frage in seinem Vorlesungsskript auf, was für mich zum ersten Anstoß wurde, genauer über diese Frage nachzudenken.

in diesem Versuch eine Idee davon vermittelt werden, wie sich Computer auch dazu nutzen lassen, ganz neue Gesetztäßigkeiten aus empirischen Daten zu gewinnen. Das Feld des "Maschinellen Lernens" (oder englisch: "Machine Learning") beschäftigt sich mit genau dieser Frage und gehört zu einem der sich am schnellsten entwichkelnden Forschungsgebieten, welches in enger Verbindung mit Internet-Firmen wie Google, Microsoft, Facebook, Baidu, etc dabei ist unsere Welt grundlegend umzugestalten.

7.2 Grundlagen

Die zentrale Fragestellung des maschinellen Lernens besteht darin, wie man aus der Beobachtung von Beispiel-Messungen, auf neue Messungen generalisieren kann. In diesem Projekt werden wir diese Frage anhand des Beispiels der handgeschriebenen Zifferner-kennung kennenlernen. Dazu werden wir einen Datensatz benutzen, welcher ca. 50.000 Beispiele von jeder Ziffer als schwarz-weiss Bilder gespeichert zur Verfügung stellt, wo jedes Bild aus 28×28 Pixeln besteht. Zumächst ist es sinnvoll, sich vor Augen zu führen, wie viele verschiedene schwarz-weiss Bilder mit 28×28 Pixeln insgesamt denkbar sind. Einfache Kombinatorik ergibt, dass es

$$2^{28\times 28} = 2^{10\times 78,4} \approx 1000^{80} = 10^{240} \tag{7.1}$$

verschiedene SW-Bilder geben kann. Diese Zahl ist viel größer noch als die Avogadro-Konstante (10^{23}) und der geschätzten Anzahl Atome (10^{80}) im Universum. Die Zahl der möglichen Bilder, die sich durch Zeichnung eines durchgehenden Striches mit der maximal Länge von L Pixeln generieren lassen, kann man auf ungefähr 10^{20} abschätzen. Die Anzahl aller Bilder mit einer handschriftlichen Ziffer, die wir als solche erkennen würden liegt etwas darunter, beträgt aber sicher mindestens 10^{10}.

7.3 1. Übung: Zeitreihenvorhersage

Wir betrachten folgende Klasse von dynamischen Systemen:

$$x_{t+1} = ux_t^2 + vx_tx_{t-1} + wx_{t-1}^2 + ax_t + bx_{t-1} + c \tag{7.2}$$

In Bild 7.1 sind zwei Zeitreihen dargestellt, die mit einer jeweils anderen Wahl für den Parameter-Vektor (u, v, w, a, b, c) generiert wurden. Der Zeitverlauf lässt ist intuitiv nicht gut zugänglich. Nichtsdestotrotz handelt es sich um ein sehr einfaches Vorhersageproblem, welches auf dem Computer mit wenig Aufwand gelöst werden kann.

7.3.1 Lösungsverfahren

Bei den Zeitreihen handelt es sich um deterministische Prozesse, die eindeutig durch die Anfangswerte x_1, x_2 und die Abbildungsvorschrift $x_{t+1} = f(x_t, x_{t-1})$ aus Gleichung (7.2) festgelegt sind. Wenn wir eine beobachtete Zeitreihe haben, stellt sich die Frage, inwieweit es möglich ist, einen funktionalen Zusammenhang wie z.B. den in Gleichung

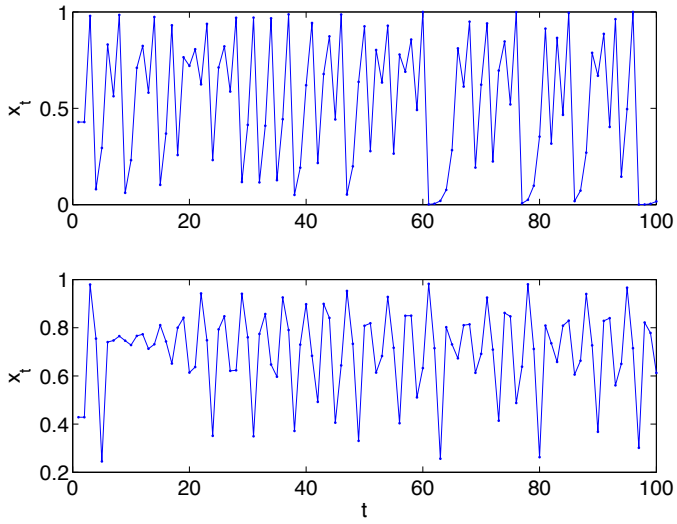

Abbildung 7.1: Die beiden Zeitreihen wurden mit einer jeweils anderen Wahl für den Parameter-Vektor (u, v, w, a, b, c) generiert. Können Sie vorhersagen, wie die beiden Zeitreihen in Zukunft weiterverlaufen werden?

(7.2), aus den Daten zu ermitteln. Als eine formal präzise Formulierung dieses Problems werden wir zumächst die folgende betrachten:

$$\text{Gegeben: } (x_{1t}, \ldots, x_{mt}, y_t)_{t=1}^N \qquad \text{Gesucht: } f \in \mathcal{F}, (x_1, \ldots, x_m) \mapsto f(x_1, \ldots, x_m),$$
$$\text{so dass } E[\epsilon(y, f(x_1, \ldots, x_m))] \text{ ein Minimum annimmt.} \tag{7.3}$$

wobei $\mathcal{F}$ irgendein ausgewählter Funktionenraum ist und $\epsilon(y, f)$ eine nichtnegative Fehlerfunktion darstellt, die für $y = f$ den Wert Null annimmt. Optimiert man alle denkbaren Funktionen, so gibt es unendlich viele Funktionen, die genau die beobachteten Werte $y_1, \ldots, y_N$ annehmen und die somit alle perfekt die Fehlerfunktion minimieren. Für neue, bisher unbeobachtete Werte der Zeitreihe lässt sich dann aber keine Vorhersage ableiten, da es für alle denkbaren Werte ebenfalls wieder unendlich viele Funktionen gibt, die exakt die beobachteten Werte annehmen würden. Die wesentliche Herausforderung des Maschinellen Lernens besteht also in der Frage, wie die Auswahl der Funktion nicht nur durch die beobachteten Daten, sondern durch zusätzliche Regularisierungsannahmen (d.h. durch Einschränkungen des Funktionenraums $\mathcal{F}$) so bestimmt werden kann, dass der zu erwartende Fehler minimiert wird.

Eine weit verbreitete Regularisierungsannahme ist die Einschränkung auf lineare Funktionen $y = f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$. Versucht man, eine lineare Funktion an gegebene Daten anzupassen, so spricht man von einer Regression. Entsprechend werden auch Zeitreihen, bei denen der funktionale Zusammenhang zwischen dem aktuellen und den vorangehenden Werten als linear angenommen wird, als autoregressive Prozesse bezeichnet. Wählt man eine quadratische Fehlerfunktion $\epsilon(y, f) = \epsilon_q(y, f) = (y - f)^2$ so kann man den Gewichtsvektor $\hat{\mathbf{w}}$, für den der empirische mittlere quadratische Fehler

$$\hat{E}[\epsilon_q(y, f)^2] = \frac{1}{N} \sum_{t=1}^{N} (y_t - f(\mathbf{x}_t))^2 \tag{7.4}$$

minimal wird, in geschlossener Form berechnen:

$$\hat{\mathbf{w}}^\top = \sum_{t=1}^{N} y_t \mathbf{x}_t^\top \left(\sum_{t=1}^{N} \mathbf{x}_t \mathbf{x}_t^\top \right)^{-1} . \tag{7.5}$$

Durch einen einfachen Trick kann man die Methode der linearen Regression auch auf viele nichtlineare Funktionsklassen erweitern. Dazu definiert man einen beliebigen *Merkmalsraum* (korrekter Weise müßte es eigentlich Merkmalsfunktion heißen):

$$g : R^m \to R^s, \quad \mathbf{x} \mapsto \mathbf{z} = g(\mathbf{x}) \tag{7.6}$$

und führt dann eine Regression von $\mathbf{z}$ anstelle von $\mathbf{x}$ auf y durch. Bekannte Merkmalsäume sind z.B. die Polynome oder radiale Basisfunktionen. Das Beispiel aus Gleichung (7.2) ist als "quadratischer Merkmalsraum" bekannt, der ein Spezialfall der Polynom-Merkmalsräume darstellt. Dabei ist die Merkmalsfunktion g wie folgt definiert:

$$g : R^2 \to R^6, \quad \begin{pmatrix} x_t \\ x_{t-1} \end{pmatrix} \mapsto \begin{pmatrix} x_t^2 \\ x_t x_{t-1} \\ x_{t-1}^2 \\ x_t \\ x_{t-1} \\ 1 \end{pmatrix} \tag{7.7}$$

und der Gewichtsvektor $\mathbf{w}^\top := (u, v, w, a, b, c)$. Somit lassen sich viele Lernprobleme auf ein verallgemeinertes Regressionsproblem $y = \mathbf{w}^\top g(\mathbf{x})$ zurückführen, bei dem die Funktion g nicht an die Daten angepasst wird, sondern nur der Gewichtsvektor $\mathbf{w}$ der dann nach wie vor in geschlossener Form berechnet werden kann:

$$\hat{\mathbf{w}}^\top = \sum_{t=1}^{N} y_t \mathbf{z}_t^\top \left(\sum_{t=1}^{N} \mathbf{z}_t \mathbf{z}_t^\top \right)^{-1} . \tag{7.8}$$

Konzeptionell kann es manchmal hilfreich sein, statt einem linearen etwas allgemeiner einen affinen Zusammenhang $y = \mathbf{w}^\top \mathbf{z} + d$ zwischen den Prediktoren $\mathbf{z}$ und den abhängigen Variablen y anzunehmen. Führt man eine Variablen-Substitution $\tilde{\mathbf{z}} := \mathbf{z} - E[\mathbf{z}]$ und $\tilde{y} = y - E[y]$ durch, so wird der letzte Eintrag im Merkmalsvektor in Gleichung (7.7) identisch Null und hat keinerlei Einfluss mehr auf die Vorhersage der abhängigen Variable y und kann daher komplett entfernt werden. Der Merkmalsvektor enthält dann nur Komponenten, die sich in irgendeiner Weise mit $\mathbf{z}$ ändern. Diese Aufspaltung lässt sich so interpretieren, dass der Mittelwert $E[y]$ die bestmögliche *a priori* Vorhersage von y ist, d.h. eine Vorhersage, die man machen kann ohne die Werte von $\mathbf{z}$ zu kennen. Die lineare Vorhersage bezieht sich dann ausschliesslich auf die Abweichung von diesem Mittelwert:

$$\hat{y} = E[y] + \tilde{y} = E[y] + \tilde{\mathbf{w}}^\top \tilde{\mathbf{z}} = E[y] + \tilde{\mathbf{w}}^\top (\mathbf{z} - E[\mathbf{z}]) \tag{7.9}$$

Aufgabe 1: Quadratische Regression

a) Reproduzieren Sie die beiden Zeitreihen aus Abbildung 1 durch Simulation des dynamischen Systems aus Gleichung (7.2) mit folgenden Parametervektoren: $\mathbf{w}_1^\top = (-4, 0, 0, 4, 0, 0), \mathbf{w}_2^\top = (-2, 0, -2, 2, 2, 0)$ und der Anfangsbedingung $x_1 = x_2 = \frac{3}{7}$.

b) Versuchen Sie durch Regression die Gewichtsvektoren aus den Zeitreihen zu bestimmen (mit Hilfe von Gleichung (7.8)) und vergleichen Sie die Vorhersage $\hat{x}_{t+1} = \mathbf{w}^\top \mathbf{z}_t$ mit den tatsächlichen Werten von x_t.

c) Führen sie "zentrierte" Variablen $\tilde{x}_t := x_t - \langle x_t \rangle$ und $\tilde{\mathbf{z}}_t := \mathbf{z}_t - \langle \mathbf{z}_t \rangle$ ein und benutzen Sie die Gleichung (7.9) um x_{t+1} vorherzusagen. Überprüfen Sie, ob die Vorhersage mit der aus b) übereinstimmt.

d) Ändere Gleichung (7.2) in das folgende stochastische System:

$$x_{t+1} = \Big(u x_t^2 + v x_t x_{t-1} + w x_{t-1}^2 + a x_t + b x_{t-1} + c + \epsilon \eta \Big) \bmod 1 \tag{7.10}$$

wobei $\eta \sim \mathcal{N}(0, 1)$ eine Zufallsvariable mit Standard-Normalverteilung ist und generiere für dieselben zwei Parametersätze wie in a) im Fall $\epsilon = 0.01$ je eine Zeitreihe. Schätze anschliessend aus den neuen Zeitreihen wieder den jeweiligen Parametervektor mittels Regression wie zuvor in b) und c).

e) Fehler finden.

7.4 2. Übung: Lineare und quadratische Klassifikation von handschriftlichen Ziffern

Wir betrachten im folgenden eine Diskriminationsaufgabe, in der Bilder mit der Ziffer 0 von solchen mit der Ziffer 1 unterschieden werden sollen. Die SW-Bilder bestehen aus $28 \times 28 = 784$ Pixeln und werden im folgenden als multivariate reelle Zufallsvektoren modelliert $\mathbf{x} \in R^{784}$ (d.h. die quadratische Anordnung der Pixel wird ignoriert). Wir benutzen einen Bayesianischen Ansatz, um das Klassifikationsproblem zu lösen. Die Wahrscheinlichkeit, dass ein Bild $\mathbf{x}$ zur Klasse c gehört, kann allgemein wie folgt berechnet werden:

$$p(c|\mathbf{x}) = \frac{p(\mathbf{x}, c)}{p(\mathbf{x})} = \frac{p(\mathbf{x}, c)}{\sum_c p(\mathbf{x}, c)} = \frac{p(\mathbf{x}|c)p(c)}{\sum_c p(\mathbf{x}|c)p(c)} \tag{7.11}$$

Im Fall von zwei Klassen von Bildern $c \in \{0, 1\}$ genügt es das Verhältnis

$$\frac{p(c=1|\mathbf{x})}{p(c=0|\mathbf{x})} = \frac{p(\mathbf{x}|c=1)p(c=1)}{p(\mathbf{x}|c=0)p(c=0)} = \exp\left(\log \frac{p(\mathbf{x}|c=1)}{p(\mathbf{x}|c=0)} + \log \frac{p(c=1)}{p(c=0)} \right) \tag{7.12}$$

zu berechnen. Im folgenden nehmen wir nun weiter an, dass die bedingten Wahrscheinlichkeiten der beiden Klassen durch eine Normalverteilung beschrieben werden kann:

$$p(\mathbf{x}|c) = \mathcal{N}(\mathbf{x}|\mu_c, \Sigma_c) = \frac{1}{\sqrt{(2\pi)^n |\Sigma_c|}} \exp\left(-\frac{1}{2} (\mathbf{x} - \mu_c)^\top \Sigma_c^{-1} (\mathbf{x} - \mu_c) \right) \tag{7.13}$$

Damit ergibt sich

$$\begin{aligned} -2\log\frac{p(\mathbf{x}|c=1)}{p(\mathbf{x}|c=0)} &= (\mathbf{x}-\mu_1)^\top\Sigma_1^{-1}(\mathbf{x}-\mu_1) - (\mathbf{x}-\mu_0)^\top\Sigma_0^{-1}(\mathbf{x}-\mu_0) + \log\frac{|\Sigma_1|}{|\Sigma_0|} \\ &= \mathbf{x}^\top\left(\Sigma_1^{-1}-\Sigma_0^{-1}\right)\mathbf{x} - 2\left(\mu_1^\top\Sigma_1^{-1}-\mu_0^\top\Sigma_0^{-1}\right)\mathbf{x} \\ &\quad +\mu_1^\top\Sigma_1^{-1}\mu_1 - \mu_0^\top\Sigma_0^{-1}\mu_0 + \log\frac{|\Sigma_1|}{|\Sigma_0|}. \\ &= \mathbf{x}^\top A\mathbf{x} + \mathbf{b}^\top\mathbf{x} + c \end{aligned} \tag{7.14}$$

Wenn wir die 0-1-Fehlerfunktion annehmen (d.h. $\epsilon(c,\hat{c}) = 0$ falls $c = \hat{c}$ und $\epsilon(c,\hat{c}) = 1$ falls $c \neq \hat{c}$) dann ist der *Maximum a poseriori* (MAP) Schätzer optimal:

$$\hat{c}_{MAP} = \begin{cases} 1 & , \quad p(\mathbf{x}|c=1) > p(\mathbf{x}|c=0) \\ 0 & , \quad p(\mathbf{x}|c=1) < p(\mathbf{x}|c=0) \end{cases} \tag{7.15}$$

welcher sich mit Hilfe von Gleichung (7.14) in geschlossener Form berechnen lässt, wobei die Klassen-Mittelwerte μ_0, μ_1 und die Kovarianzen Σ_0, Σ_1 Schätzwerte sind, die mit Hilfe der Trainingsdaten bestimmt werden müssen. Ein weit verbreiteter Kandidat sind der Sample-Mittelwert:

$$\mu_{sample} = \frac{1}{N}\sum_{k=1}^{N}\mathbf{x}_k \tag{7.16}$$

und der Sample-Kovarianz-Schätzer:

$$\Sigma_{sample} = \frac{1}{N}\sum_{k=1}^{N}(\mathbf{x}_k-\mu)(\mathbf{x}_k-\mu)^\top \tag{7.17}$$

die die "Maximum Likelihood"-Schätzer unter der Annahme einer Normalverteilung sind. Bei einer begrenzten Menge an Trainingsdaten ist die Verwendung des Sample-Kovarianz-Schätzers nicht ideal, da sich dieser Schätzer zu sehr von zufälligen Schwankungen in den Daten beeinflussen lässt, welche zu einer suboptimalen Klassifikationsleistung führen. Wenn man annimmt, dass alle Komponenten des Zufallsvektors unkorreliert sind, ergibt sich folgender Kovarianz-Schätzer:

$$\Sigma_{diag} = \mathrm{diag}(\mathrm{diag}(\Sigma_{sample})) \tag{7.18}$$

Kreuzvalidierungsverfahren sind ein wichtiges Werkzeug um den Vorhersage-Fehler zu minimieren. Die Funktionsweise von Kreuzvalidierungsverfahren soll hier anhand von einem Beispiel demonstriert werden.

Für quadratischen Funktionen wächst die Anzahl der zu lernenden Parameter quadratisch mit der Anzahl der Dimensionen. Daher kann eine Dimensionsreduktion sehr effektiv dazu beitragen, die Schätzung der Funktion robust gegen zuälligen Fluktuationen in den Daten zu machen. Ein nützliches Verfahren ist die sogenannte "oriented Principal Component Analysis" (oriented PCA), welche eine Basis bestimmt, in der die Basisvektoren entsprechend eines Signal-Rausch-Verhältnisses sortiert sind. Dieses

lässt sich mittels eines generalisierten Eigenwertproblems bestimmen. Dazu betrachten wir zunächst eine Zerlegung der Gesamt-Kovarianz

$$\begin{aligned}\Sigma_{total} &= E[(\mathbf{x}-E[\mathbf{x}])(\mathbf{x}-E[\mathbf{x}])^\top] \\ &= \underbrace{p(c=1)p(c=0)(\mu_1-\mu_0))(\mu_1-\mu_0))^\top}_{Cov[E[\mathbf{x}|c]]} + \underbrace{p(c=1)\Sigma_1+p(c=0)\Sigma_0}_{E[Cov[\mathbf{x}|c]]}\end{aligned}$$

in den Signalanteil $\Sigma_{signal} = Cov[E[\mathbf{x}|c]]$ und den mittleren Rauschanteil $\Sigma_{avg\,noise} = E[Cov[\mathbf{x}|c]]$. Wir definieren als erstes eine neue Repräsentation der Daten $\mathbf{y} = \Sigma_{avg\,noise}^{-1/2}$ in welcher der Effekt der zufälligen Streuungen (das "Rauschen") isotrop ist (also nicht von der Richtung im gewählten Koordinatensystem abhängt. Die Kovarianz-Matrix von $\mathbf{y}$ ergibt sich aus der ursprünglichen wie folgt:

$$\Sigma_y = E[C^{-1/2}(\mathbf{x}-E[\mathbf{x}])(\mathbf{x}-E[\mathbf{x}])^\top C^{-1/2\top}] = C^{-1/2}\Sigma_x C^{-1/2} = C^{-1/2}\Sigma_{signal}C^{-1/2} + I$$

Durch Eigenwertzerlegung der Kovarianzmatrix $\Sigma_y = UDU^\top$ mit $D_{11} \geq \ldots \geq D_{dd} \geq 0$ lässt sich eine Dimensionsreduktion definieren, welche das Signal-Rausch-Verhältnis bei gegebener Anzahl an Dimension maximiert:

$$W_m = U_m^\top C^{-1/2}$$

wobei U_m aus den ersten m Spaltenvektoren $U_m = (\mathbf{u}_1, \ldots, \mathbf{u}_m)$ der Eigenvektor-Matrix U besteht. Entsprechend erhalten wir

$$\begin{aligned}-2\log\frac{p(W_m\mathbf{x}|c=1)}{p(W_m\mathbf{x}|c=0)} &= \mathbf{x}^\top W_m^\top\left(\tilde{\Sigma}_1^{-1}-\tilde{\Sigma}_0^{-1}\right)W_m\mathbf{x} - 2\left(\mu_1^\top\tilde{\Sigma}_1^{-1}-\mu_0^\top\tilde{\Sigma}_0^{-1}\right)W_m\mathbf{x} \\ &\quad +\mu_1^\top W_m^\top\tilde{\Sigma}_1^{-1}W_m\mu_1 - \mu_0^\top W_m^\top\tilde{\Sigma}_0^{-1}W_m\mu_0 + \log\frac{|W_m\tilde{\Sigma}_1W_m^\top|}{|W_m\tilde{\Sigma}_0W_m^\top|}.\end{aligned}$$

Es gibt viele verschiedene Kreuzvalidierungs-Methoden. Die wesentliche Idee lässt sich besonders leicht anhand der sogenannten $twofold$ Kreuzvalidierungsmethode erklären. Dazu teilen wir die Trainingsdaten zufällig in zwei gleichgroße disjunkte Teilmengen M_1 und M_2 die abwechselnd als Trainingsdaten und als Testdaten dienen. Zuerst benutzen wir die Daten in M_1 um den Gewichtsvektor zu bestimmen und die ungenutzten Testdaten in M_2 um den Generalisierungsfehler abzuschätzen. Anschliessend benutzen wir umgekehrt die Daten in M_2 um den Gewichtsvektor zu bestimmen und die Daten in M_1 um den Generalisierungsfehler abzuschätzen. Auf diese Weise wird gewährleistet, dass jeder Datenpunkt einmal zum Trainieren und einmal zum Testen benutzt wird (also alle Datenpunkte gleichstark gewichtet werden). Diese Prozedur kann man mehrere Male mit neuen zufälligen Aufspaltungen der Daten wiederholen, um die zufälligen Schwankungen in der Schätzung des Generalisierungsfehler zu reduzieren.

Durch die Kreuzvalidierung bekommen wir eine verläßliche Schätzung für den Generalisierungsfehler. Das können wir benutzen, um den optimalen Parameter m (i.e. die optimale Anzahl der Dimensionen) zu finden, bei dem der Generalisierungsfehler minimal wird.

Aufgabe 2: MNIST-Klassifizierung

a) Gehen Sie zu der Web-Seite `http://yann.lecun.com/exdb/mnist/` und laden Sie die MNIST-Daten herunter. Benutzen Sie die gesamten Trainingsdaten, um den Sample-Mittelwert und die Sample-Kovarianzmatrix jeweils für die Klasse der Nullen und für die Klasse der Einsen zu schätzen. Fertigen Sie eine Abbildung an, die die beiden Mittelwerte als Bilder darstellt. Führen Sie eine Eigenwertzerlegung der Kovarianzmatrix durch und plotten Sie die Eigenbilder zu den drei größten Eigenwerten jeweils für beide Klassen.

b) Benutzen Sie den Sample-Mittelwert und Kovarianzmatrizen um die Parameter A und $\mathbf{b}$ der quadratischen Regression (Gleichung 7.8) zu bestimmen. Fertigen Sie eine Abbildung an, die die Verteilung der Diskriminante $d(\mathbf{x}) := \mathbf{x}^\top A\mathbf{x} + \mathbf{b}^\top\mathbf{x}$ für die Daten in den beiden Klassen zeigt (beide Histogramme in eine Abbildung aber mit unterschiedlichen Farben für die beiden Klassen).

c) Fertigen Sie eine Abbildung an, wie sich der Trainingsfehler als Funktion der Schwelle c ändert, wenn man als Klassifikator

$$\hat{c}(\mathbf{x}) = \begin{cases} 1 & d(\mathbf{x}) > c \\ 0 & d(\mathbf{x}) < c \end{cases}$$

benutzt.

d) Bestimmen Sie den Diskriminationsfehler auf den Testdaten, wenn Sie die in c) bestimmte optimale Schwelle c benutzen.

e) Bestimmen Sie mit Hilfe von oriented PCA und twofold Kreuzvalidierung eine optimale Dimensionreduktion und bestimmen Sie dafür den Diskriminationsfehler auf den Testdaten.

f) Man kann die Anzahl der Parameter auch dadurch einschränken, dass man die funktionale Abhängigkeit vereinfacht. Wenn man bereit ist, die Annahme zu machen, dass die Kovarianzen für die beiden Klassen identisch sind, als $\Sigma_1 = \Sigma_0$ gilt, dann reduziert sich das Log-Likelihood-Verhältnis auf eine lineare Form:

$$\log \frac{p(\mathbf{x}|c=1)}{p(\mathbf{x}|c=0)} = (\mu_1 - \mu_0)^\top (\Sigma_0 + \Sigma_1)^{-1} \mathbf{x} + \tilde{c} \tag{7.19}$$

Bestimmen Sie wie in e) mit Hilfe von oriented PCA und twofold Kreuzvalidierung eine optimale Dimensionreduktion für den linearen Klassifikator und ermitteln Sie dafür den Diskriminationsfehler auf den Testdaten.

g) Fehler finden.

7.5 Literatur

1) Christopher Bishop: Pattern Recognition and Machine Learning (`http://www.amazon.de/dp/8132209060/ref=rdr_ext_tmb`)

2) David Barber: Bayesian Reasoning and Machine Learning (`http://www.amazon.de/gp/offer-listing/B00OL3I4JG`)