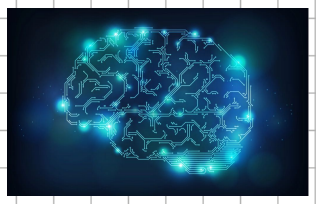
Einführung



2 wichtigste Pfeiler: Daten & Modell

Entwicklung einer ML-Anwendung:
1. Problem modellieren → 2. Lernen → 3. Evaluation (→ evt. iterieren)

no free lunch theorem:
für jedes Modell / jeden Algorithmus gibt es Probleme für welche die Resultate sehr schlecht sind

supervised learning (SL)

basierend auf Eingabewert $\vec{x} \in \mathcal{X}$ möchten wir einen Ausgabewert $\vec{t} \in \mathcal{T}$ voraussagen wobei wir für den Lernprozess eine Daten-Menge $D = \{(\vec{t}_k, \vec{x}_k) \mid 1 \leq k \leq n_D\}$ von beobachtbaren Wertepaaren zur Verfügung haben

$\mathcal{X}$ ist oft $\mathbb{R}^d$, kann aber Menge mit anderen strukturierten Elementen sein

$\mathcal{T}$ kann $\mathbb{R}^d$ sein (Regressionsproblem) oder endliche Menge $\{C_1, \dots, C_m\}$ (Klassifikationsproblem)

Vorgehen: geeignete Funktion $f: \mathcal{X} \to \mathcal{T}$ spezifizieren, die an Trainingsdaten D angepasst wird und deren Funktionswerte $f(\vec{x})$ zur Voraussage von $\vec{t} \in \mathcal{T}$ verwendet werden

als Qualitätsmass benutzt man oft den Verallgemeinerungsfehler

$$\overline{L_{f,T}} = \frac{\sum_{k=1}^{n_T} L(\vec{t}_k, f(\vec{x}_k))}{n_T}$$

über alle Testdaten gemittelter Verlust

wobei L ein für die Anwendung geeignetes Verlustmass und $T = \{(\vec{t}_k, \vec{x}_k) \mid 1 \leq k \leq n_T\}$ unabhängige Testdaten bezeichnen
→ es ist niemals zulässig $\overline{L_{f,D}}$ als Qualitätsmass zu verwenden!

man möchte ein Modell p_m der bedingten Verteilung (Verteilung im Sinne von pmf / pdf) $p(\vec{t} \mid \vec{x})$ erlernen

unter Verwendung eines geeigneten Verlustmasses können wir aus der gelernten bedingten Verteilung $p_m(\vec{t} \mid \vec{x})$ dann eine optimale Voraussagefunktion $f: \mathcal{X} \to \mathcal{T}$ bestimmen

unsupervised learning (UL)

(im Gegensatz zu) überwachtem Lernproblem gibt es keinen ausgezeichneten Ausgabewert und die Lernbasis ist schlicht eine Menge von beobachteten Daten $D = \{\vec{x}_k \mid 1 \leq k \leq n_D\}$

Ziel ist es, die zugrundeliegende Verteilung zu erfassen / repräsentieren und dadurch im weitesten Sinne nutzbar zu machen

Vorteil: kein (aufwändiges) Labeling von Daten nötig

wahrscheinlichkeitstheoretisch kann man unbeaufsichtigtes Lernproblem als Anpassung eines probabilistischen Modells p_m an die den Trainings-Daten D zugrundeliegende Verteilung $p(\vec{x})$ auffassen (density estimation)

häufige Fragestellung:
für Eingabe $\vec{x}$ kompakte, komprimierte sowie für weitere Verarbeitung effiziente und informative Repräsentation $\vec{z}$ (den verborgenen Zustand) finden
indem wir prob. Modell $p_m(\vec{x}, \vec{z})$ an Trainingsdaten D anpassen können wir danach für neue Eingabe $\tilde{\vec{x}}$ zB das Maximum oder Mittelwert $\tilde{\vec{z}}$ der mithilfe trainierten Modells berechneten bedingten Verteilung $p_m(\vec{z}, \tilde{\vec{x}})$ als Repräsentanten für $\tilde{\vec{x}}$

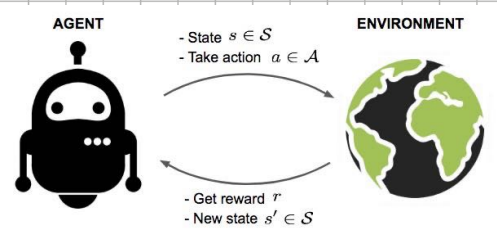
Beispiele:
- Clustering: Unterteilung von Daten in Gruppen von untereinander ähnlichen Daten
- Dimensionsreduktion: (hochdimensionalen) Eingaben $\vec{x}$ niedrigdimensionale Repräsentanten $\vec{z}$ zuordnen so, dass die für die nachfolgende Verarbeitung relevante Information erhalten bleibt und einfacher zugänglich ist
  → der verborgene Zustand wird vielfach als Projektion der Eingabe auf einen Unterraum (im einfachsten Fall ein affiner Unterraum) konstruiert
- Zustandsraummodell

als Qualitätsmass eignet sich die Likelihood:

$$L(T|M) = p_M(T) = \prod_{k=1}^{n_T} p_M(\vec{x}_k)$$

es beschreibt die Wahrscheinlichkeit der Testdaten $T = \{\vec{x}_k \mid 1 \le k \le n_T\}$ unter dem auf den Daten D trainierten Modelles p_M

reinforcement learning (RL)



Agent kann mittels Aktionen mit Umgebung wechselwirken

auf Aktion $a_t \in A$ zum Zeitpunkt t reagiert Umgebung mit mit Veränderung ihres Zustandes $s_t \in S \rightsquigarrow s_{t+1} \in S$ und der Aussendung einer Belohnung r_{t+1}, welche beide vom Agenten beobachtet werden

Agent ist definiert durch seine Handlungsstrategie (policy) $\pi : S \to A$ mit welcher er basierend auf aktuellem Zustand $s \in S$ der Umgebung die Aktion $\pi(s) \in A$ wählt

Ziel ist Finden einer policy $\pi : S \to A$ mit mit welcher die über sehr viele Episoden gemittelte (erwartete) akkumulierte Belohnung $g_t = r_1 + \dots + r_t$ maximal gross wird

Herausforderungen:
- Wirkung einer einzelnen Aktion auf akkumulierte Belohnung nicht unmittelbar erfahrbar
- bei einigen Umgebungen sind Belohnungen bei meisten Schritten ≤ 0 und nur gelegentlich (zB bei Erreichen einiger terminaler Zustände) Aktionen mit positiver Belohnung verbunden sind

als Abgrenzung zu UL gibt es (in Form der Belohnung) Rückmeldung mittels welcher policy als ganzes evaluiert werden muss, im Gegensatz zu SL ist Rückmeldung direkt in Form von Beispielen guter (Ausgabe-, Eingabe-) Paare (a, s)

welche Aktionen in einem Zustand lohnend sind, muss Agent aus seinem akkumulierten Wissen über Umgebung extrahieren => hierzu ist einerseits Erforschung (Exploration) der Wirkung der Aktionen auf akkumulierte Belohnung nötig, aber andererseits sollen auch gute Aktionen gewählt werden (exploitation), weil ja akkumulierte Belohnung gross sein soll

Modellierung des Lernproblems meist als Form von sequentiellen, markovschen Entscheidungsprozessen (MDP). dabei ist Umgebung definiert durch bedingte Verteilungen $p(\tilde{s}, \tilde{r} \mid s, a)$ über den neuen Zustand $\tilde{s}$ der Umgebung und ausgesendete Belohnung $\tilde{r}$, gegeben dass im aktuellen Zustand s die Aktion a gewählt wird. policy ist dabei durch bedingte Verteilung $\pi(a|s)$ über Aktionen a im Zustand s der Umgebung definiert
→ beides zusammen erlaubt die Verteilung über Episoden zu berechnen und bildet das Fundament um Methoden für RL zu entwickeln

zentrale Begriffe

anhand eines Regressionsproblems mit Eingaben $x \in \mathbb{R}$ und Ausgaben $t \in \mathbb{R}$

Daten: zugrundeliegende Funktion $g(x) = \sin(2\pi x)$ überlagert mit normalverteiltem ($\mu=0, \sigma=0.3$) Rauschen r: $t = g(x) + r$

Datensatz generieren mit and $[0,1]$ gleichm. vert. Eingaben Ausgabe gemäss obiger Formel ergibt $D = \{(t_k, x_k) \mid 1 \leq k \leq n_D\}$ und $T = \{(t_k, x_k) \mid 1 \leq k \leq n_T\}$

Verlustmass: $L(t, p) = (t-p)^2$

Verallgemeinerungsfehler: $\overline{L_{f,T}} = \frac{\sum_{k=1}^{n_T} (t_k - f(x_k))^2}{n_T}$ MSE oder $\sqrt{\frac{\sum_{k=1}^{n_T} (t_k - f(x_k))^2}{n_T}}$ RMSE

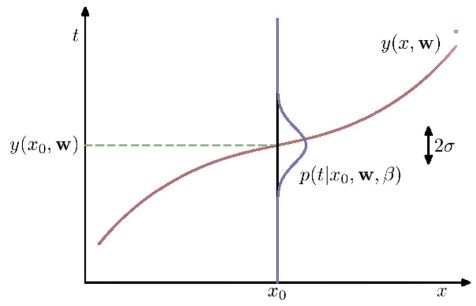
für ein genügend grosses n_T erwarten wir für die wahre Funktion g einen RMSE in der Grössenordnung der Standardabweichung des Rauschens also $E_{RMS}(g, T) \simeq \sigma$

Modell für zugrundeliegende Regularität:

$$y(x, \vec{w}) = \sum_{j=0}^{M} w_j x^j = w_0 + w_1 x + \cdots + w_M x^M$$

Modell für Abweichung: Normalverteilung

insgesamt erhalten wir das wahrscheinlichkeitsth. Modell



→ Präzision (inverse Varianz)

$$p(t \mid x, \vec{w}, \beta) = \mathcal{N}(t \mid y(x, \vec{w}), \beta^{-1})$$ polynomiale Regression mit Gaussschem Rauschen

mit adaptiven Parametern $\vec{\theta} = (\vec{w}, \beta)$ mit $\vec{w} \in \mathbb{R}^{M+1}$, $\beta > 0$ wobei $\beta = \nu^{-1} = \sigma^{-2}$

es folgen 3 allgemeine Methoden für die Anpassung von parabolischen Modellen an Daten

Maximum-Likelihood-Methode Kenle 1

→ pmf (diskrete Daten) oder pdf (kont. Daten)

Likelihood der Daten $D = L(\vec{\theta}|D) = p(D|\vec{\theta}) = \prod_{k=1}^{n_D} p(t_k|\vec{\theta})$

Ziel: Parameter $\boxed{\vec{\theta}_{ML} = \text{argmax}_{\vec{\theta}}\, p(D|\vec{\theta})}$ finden welcher die Likelihood maximiert

mithilfe der relativen Entropie (Kullback-Leibler Distanz resp KL divergence) kann ML-Methode auch als die im Sinne der Entropie beste Näherung an die empirische Verteilung der Daten D erkannt werden

Optimierungsproblem, welches nur manchmal analytisch (geschlossen) lösbar ist, in vielen Fällen jedoch numerisch angegangen werden muss

für das polynomiale Regressionsproblem erhalten wir eine geschlossene Lösung:

$$L(\vec{\theta}|D) = \prod_{k=1}^{n_D} \mathcal{N}(t_k|y(x_k,\vec{\omega}),\beta^{-1}) = \prod_{k=1}^{n_D}\left((2\pi)^{-\frac{1}{2}}\beta^{\frac{1}{2}}\exp\left(-\frac{1}{2\beta^{-1}}(t_k - y(x_k,\vec{\omega}))^2\right)\right) \quad (1)$$

$$\ln L(\vec{\theta}|D) = -\frac{\beta}{2}\sum_{k=1}^{n_D}\left(t_k - y(x_k,\vec{\omega})\right)^2 + \frac{n_D}{2}\ln(\beta) - \frac{n_D}{2}\ln(2\pi) \quad (2)$$

$$\frac{1}{2}\sum_{k=1}^{n_D}\left(t_k - y(x_k,\vec{\omega})\right)^2 = \frac{1}{2}\sum_{k=1}^{n_D}\left(t_k - \sum_{j=0}^{M}\omega_j x^j\right)^2 = \frac{1}{2}\left\|\vec{t} - \underline{X}\vec{\omega}\right\|^2_{\mathbb{R}^{n_D}} \text{ mit } X_{k,j} = x_k^{j-1} \quad (3)$$

Maximimierung von (1) bez. $\vec{\omega}$ ist äquivalent zu Minimierung von (3) bez. $\vec{\omega}$

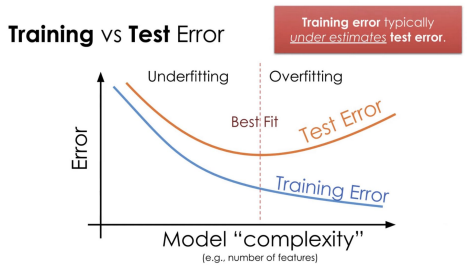
ML-Methode liefert für das polynomiale Regressionsproblem die Punktschätzungen

$$\boxed{\vec{\omega}_{ML} = (\underline{X}^T\underline{X})^{-1}\underline{X}^T\vec{t} \quad \text{und} \quad \beta_{ML}^{-1} = \frac{1}{n_D}\left\|\vec{t} - \underline{X}\vec{\omega}_{ML}\right\|^2_{\mathbb{R}^{n_D}}}$$

mit Systemmatrix $\underline{X} \in \mathbb{R}^{n_D \times (M+1)}$ und Daten $\vec{t}, \vec{x} \in \mathbb{R}^{n_D}$

bei Implementierung $\vec{\omega}_{ML}$ nicht gemäss Formel sondern direkt als Lösung des Kleinstfehler-Quadrat-Problems berechnen (mit QR- oder Singulärwertzerlegung)

Evaluation der Qualität



beim polynomialen Regressionsmodell tritt im Falle $M \approx n_D$ das Phänomen overfitting auf, da dann die Datenpunkte exakt wiedergegeben werden können resp. $Erms(y, D) = 0$, die Verallgemeinerungsqualität jedoch sehr schlecht ist

ML-Methode tendiert dazu ein Modell sehr fein an die Daten (inklusive Rauschen) anzupassen, falls das Modell in Bezug auf die Menge an verfügbaren Daten zu flexibel ist

bei konstant gehaltener Menge Trainingsdaten nimmt bei ML-Methode Testdatenfehler in Abhängigkeit von zunehmender Flexibilität typischerweise U-Form an während Trainingsdatenfehler monoton abfällt

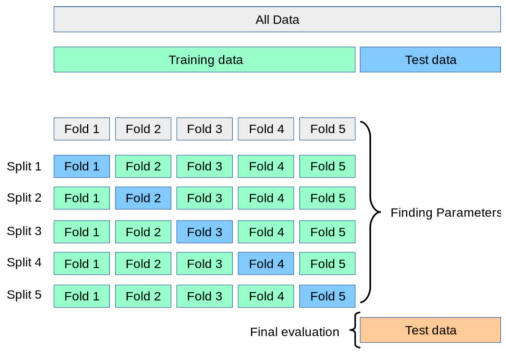
i.A. reduziert sich Überanpassungsproblem mit zunehmender Trainingsdatenmenge bei gleichbleibender Modellflexibilität

Modellselektion

Ziel ist unter verschiedenen Modellen mit gewöhnlich variierender Flexibilität das Modell mit der besten Verallgemeinerungsqualität auszuwählen. daher müsste man einen weiteren Teil der zur Verfügung stehenden Daten für die Selektion des Modells reservieren (zusätzlich zu den Testdaten für die finale Evaluation)

die Varianz der Schätzung des Verallgemeinerungsfehler ist umso kleiner je mehr Daten wir für das Evaluieren verwenden (Gesetz der grossen Zahlen). andererseits möchten wir einen möglichst grossen Anteil der Daten zum Trainieren benützen, um möglichst für alle Modelle die Überanpassung zu vermeiden

mithilfe der Kreuzvalidierung versucht man diesen Zielkonflikt zu entschärfen und die verfügbaren Daten effizienter zu nutzen, muss mit Rechenleistung bezahlt werden muss, weil alle Modelle dafür mehrmals trainiert werden müssen



bei der k-fach Kreuz-Validierung werden verfügbare Daten D in k Teile $D_1, \ldots, D_k$ zerlegt. anschliessend werden für alle Teile $j \in \{1, \ldots, k\}$ alle Modelle $M_1, \ldots, M_m$ auf $D \setminus D_j$ trainiert und dann auf D_j der Fehler $E(M_i, D_j)$ evaluiert. danach wird für jedes Modell der gemittelte Fehler

$$E^{k\text{-cv}}(M_i, D) = \frac{1}{k}\left(\sum_{j=1}^{k} E(M_i, D_j)\right)$$

als Verallgemeinerungsfehler verwendet. kann eingesetzt werden um 1 einzelnes Modell zu evaluieren oder um unter verschiedenen Modellen zu selektieren. für die Verwendung zur Selektion muss für die die Evaluation des schliesslich selektierten Modells ein n für die KV verwendetes Testdatenset abgewendet werden, wobei das CV-selektierte Modell auf gesamten Trainingsdaten neu trainiert wird, bevor die Qualität auf den Testdaten evaluiert wird

Parameter, welche die Flexibilität/Komplexität kontrollieren, können nicht mit ML-Methode auf Trainingsdaten alleine adaptiert werden, da die ML-M. immer die flexibelsten Modelle bevorzugt und damit Überanpassung provoziert wird

Maximum-a-posteriori-Methode Keule 2

bezieht Parameter ein, sich mithilfe einer Prior-Verteilung – auch: Prior-Verteilung $p(\vec{\theta})$ repräsentiert unser aktuelles Wissen / unsere Annahmen über Parameter des Modells, bevor wir (weitere) Daten beobachten

bei polynomialen Regression können wir sagen dass grosse Koeffizienten a priori weniger wahrscheinlich sind als kleine (da diese zu starken Oszillationen führen) indem eine Verteilung mit um $\vec{0} \in \mathbb{R}^{M+1}$ konzentrierter Wahrscheinlichkeitsmasse, zur Vereinfachung von späteren Berechnungen in Form einer Normalverteilung

$$p(\vec{\omega}|\alpha) = \mathcal{N}(\vec{\omega}|\vec{0}, \alpha^{-1}\underline{E}_{M+1}) = \left(\frac{\alpha}{2\pi}\right)^{\frac{M+1}{2}} \exp\left(\frac{-\alpha}{2}\vec{\omega}^T\vec{\omega}\right)$$

mit Mittelwert $\vec{0}$ und Kovarianz $\alpha^{-1}\underline{E}_{M+1}$ wählen

um nun sowohl die Prior-Verteilung $p(\vec{\theta})$ als auch Daten $D = \{(t_k, x_k) | 1 \leq k \leq n_D\}$ miteinzubeziehen können wir mit der Bayes-Formel die Posterior-Verteilung der Parameter $\vec{\theta}$ berechnen:

$$p(\vec{\theta}|D) = \frac{p(D|\theta)p(\theta)}{p(D)}$$

posterior $\propto$ likelihood $\times$ prior

bei MAP-Methode berechnet man

$$\vec{\theta}_{MAP} = \text{argmax}_{\vec{\theta}}\, p(\vec{\theta}|D) = \text{argmax}_{\vec{\theta}}\, p(D|\vec{\theta})p(\vec{\theta}) = \text{argmax}_{\vec{\theta}} \prod_{k=1}^{n_D} p(t_k|\vec{\theta})\, p(\vec{\theta})$$

welcher Posterior-Verteilung $p(\vec{\theta}|D)$ des Parameters $\vec{\theta}$ gegeben die Daten D maximiert

Evidenz oder marginale Likelihood $p(D) = \int p(D|\vec{\theta})\, p(\vec{\theta})\, d\vec{\theta}$, welche analytisch meistens nicht geschlossen berechenbar und deren numerische Bestimmung nicht immer einfach und manchmal sehr CPU-intensiv ist, muss für MAP-Methode nicht berechnet werden

für das polynomiale Regressionsproblem erhalten wir:

$$p(\vec{t}|\vec{x}, \vec{\omega}, \beta)\, p(\vec{\omega}|\alpha) = \prod_{k=1}^{n_D} \mathcal{N}(t_k|y(x_k, \vec{\omega}), \beta^{-1})\, \mathcal{N}(\vec{\omega}|\vec{0}, \alpha^{-1}\underline{E}_{M+1}) \quad (1)$$

$$\frac{1}{2}\sum_{k=1}^{n_D}\left(t_k - y(x_k, \vec{\omega})\right)^2 + \frac{\alpha}{\beta}\frac{\vec{\omega}^T\vec{\omega}}{2} = \frac{1}{2}\left\| \begin{pmatrix} \vec{t} \\ \vec{0}_{M+1} \end{pmatrix} - \begin{pmatrix} \underline{X} \\ \sqrt{\frac{\alpha}{\beta}}\underline{E}_{M+1} \end{pmatrix}\vec{\omega} \right\|^2 \quad (2)$$

neg, log, etc

(2) wird augmentierter (regularisierter) Summe-der-Quadrate-Fehler genannt

Maximimierung von (1) bez. $\vec{\omega}$ ist äquivalent zu Minimierung von (2) bez. $\vec{\omega}$

MAP-Methode liefert die Punktschätzung

$$\vec{\omega}_{MAP} = (\underline{X}^T\underline{X} + \lambda\underline{E}_{M+1})^{-1}\underline{X}^T\vec{t} \quad \text{mit } X_{k,j} = x_k^{j-1}$$

wobei $\lambda = \frac{\alpha}{\beta}$, $\underline{X} \in \mathbb{R}^{n_D \times (M+1)}$, $\underline{E}_{M+1} \in \mathbb{R}^{(M+1)\times(M+1)}$

wir erhalten eine durch $\alpha > 0$ parametrisierte Familie von Modellen. α (oder äquivalent λ) kontrolliert die Flexibilität des Modells wobei kleinere Werte von λ zu flexibleren Modellen führt. wir können α (oder auch Grad d. Polyn. M) allerdings noch immer nicht auf Basis der Trainingsdaten alleine bestimmen!

Plug-In Modell: $\vec{\theta}_{PI} = \int p(\vec{\theta}|D)\, \vec{\theta}\, d\vec{\theta}$ Mittelwert anstatt Maximum

Bayes-Methode Kerle 3

führt auf prinzipiellen Ansatz zur Selektion der Flexibilität von Modellen (basierend auf Trainingsdaten, ohne Kreuzvalidierung)

Ursache der Überanpassung bei ML und MAP: Vernachlässigung der nach dem Lernen verbleibende Unsicherheit in den Parametern $\vec{\theta}$

BM bestimmt Posterior-Voraussage-Verteilung mit Marginalisierung über Modellparameter $\vec{\theta}$

$$p(t|D) = \int p(t|\vec{\theta})\, p(\vec{\theta}|D)\, d\vec{\theta}$$

unter Einbezug des aktuellen Wissens (in Form der Posterior-Verteilung)

«Mischung der Modelle $p(t|\vec{\theta})$ zu allen möglichen Parametern wobei jeder Parameter $\vec{\theta}$ entsprechend der Posterior-Verteilung gewichtet wird»

$p(t|D)$ berücksichtigt auch die nach Beobachtung der Daten verbleibende Unsicherheit über die Parameter $\vec{\theta}$

für das polynomiale Regressionsproblem erhalten wir:

$$p(t|x, \vec{t}, \vec{x}, \alpha, \beta) = \int p(t|x, \vec{w}, \beta)\, p(\vec{w}|\vec{t}, \vec{x}, \alpha, \beta)\, d\vec{w}$$

wobei $p(t|x, \vec{w}, \beta) = \mathcal{N}(t|y(x, \vec{w}), \beta^{-1})$, $p(\vec{w}|\vec{t}, \vec{x}, \alpha, \beta) = \frac{p(\vec{t}|\vec{x}, \vec{w}, \beta)\, p(\vec{w}|\alpha)}{\int p(\vec{t}|\vec{x}, \vec{w}, \beta)\, p(\vec{w}|\alpha)\, d\vec{w}}$, $p(\vec{w}|\alpha) = \mathcal{N}(\vec{w}|\vec{0}, \alpha^{-1}\underline{E})$

für dieses einfaches Modell sind die Integrale analytisch geschlossen berechenbar und wir erhalten:

$$p(t|x, \vec{t}, \vec{x}, \alpha, \beta) = \mathcal{N}(t|m(x), v(x))$$

mit $m(x) = \vec{b}(x)^T \vec{m}$, $v(x) = \beta^{-1} + \vec{b}(x)^T \underline{S}\, \vec{b}(x)$, $\vec{b}(x) = (1, x, x^2, \ldots, x^M)^T$, $\underline{S}^{-1} = \alpha \underline{E} + \beta \underline{X}^T \underline{X}$, $\vec{m} = \beta \underline{S} \underline{X}^T \vec{t}$

diese Voraussage-Verteilung hängt noch immer von den Hyper-Parametern α, β, M ab. um unter dieser Familie das beste zu wählen gibt es im Rahmen der Bayes-Methode den Ansatz die Unsicherheit in den (Hyper-)Parametern wieder explizit mittels einer Prior-Verteilung zu benennen.

bspw. kann bei einer Familie von parametrischen Modellen $p(t|\vec{\theta}, M)$ mit dazugehörigen Prior-Verteilungen $p(\vec{\theta}|M)$ die auf den Trainingsdaten $D = \{t_k | 1 \le k \le n_D\}$ berechnete Evidenz oder marginale Likelihood

$$p(D|M) = \int p(D|\vec{\theta}, M)\, p(\vec{\theta}|M)\, d\vec{\theta} = \int \prod_{k=1}^{n_D} p(t_k|\vec{\theta}, M)\, p(\vec{\theta}|M)\, d\vec{\theta}$$

zur Selektion von Modellen verwendet werden (nur mithilfe der Trainingsdaten alleine!)

Modelle für diskrete Daten und die Bayes-Methode

im Fall von diskreten Daten ist Bayes-Methode berechenbar

Endliche Familien diskreter Verteilungen

Annahme: echte θ^* von endlich vielen möglichen Hypothesen $\theta \in \{1,\dots,r\}$ generiert diskrete Daten $x \in \mathcal{X}$ mit Verteilung $p(x|\theta^*)$ und wir möchten basierend auf Testdaten $D=\{x_j \mid 1 \le j \le n_D\}$ eine von zukünftigen Werten abhängende Grösse voraussagen
→ gelingt ausbesten falls wir $p(x|\theta^*)$ rekonstruieren können

falls Werte nur endlich viele Werte $|\mathcal{X}|=m$ auftreten können wir diese nummerieren $\mathcal{X}=\{1,\dots,m\}$ und Verteilungen aller Hypothesen in einer $m \times r$-Matrix $M=p(x|\theta)$ mit den Einträgen $M_{k,l}=p(x=k|\theta=l)$ kodieren

Likelihood und ML-Methode

für die Likelihood der Daten D unter der Hypothese θ gilt $p(D|\theta)=\prod_{j=1}^{n_D} p(x_j|\theta)=\prod_{k=1}^{m} p(k|\theta)^{n_k}$

wobei $n_k=|\{j \in \{1,\dots,n_D\} \mid x_j=k\}|$ die Anzahl der Datenpunkte mit Wert $k \in \{1,\dots,m\}$ bezeichnet

die Anzahlen $\vec{n}=(n_1,\dots,n_m)$ werden erschöpfende Statistik genannt

mit der ML-Methode können die Hypothese θ_{ML} wählen welche das Produkt maximiert wobei wie immer bei der ML-Methode das Risiko einer Überanpassung besteht

Prior/Posterior-Verteilung und MAP- und Bayes-Methode

falls Vorwissen über Hypothesen $\theta=\{1,\dots,r\}$ vorhanden ist, können wir dies auch hier in Form einer diskreten Prior-Verteilung $p(\theta)=(\pi_1,\dots,\pi_r)$ über die Parameter θ des parametrisierten Modells $p(x|\theta)$ verwerten und die Voraussage-Verteilung mittels Marginalisierung berechnen:

Voraussage-Verteilung: $p(x)=\sum_\theta p(x|\theta)p(\theta)$

falls Prior-V. $p(\theta)$ und Daten D vorliegen kann man Posterior-V. berechnen:

Posterior-Verteilung: $p(\theta|D)=\frac{p(D|\theta)p(\theta)}{p(D)}=\frac{\prod_{k=1}^m p(k|\theta)^{n_k} p(\theta)}{\sum_{l=1}^r (\prod_{k=1}^m p(k|l)^{n_k} p(l))}$

MAP-Methode verwendet Maximum θ_{MAP} der Posterior-V. als Basis für Voraussagen: $p(x|\theta_{MAP})$

Bayes-Methode benutzt induzierte Posterior-Voraussage-Verteilung: $p(x|D)=\sum_\theta p(x|\theta)p(\theta|D)$

Sequentielles Lernen

mittels Induktion kann man zeigen, dass man bei der Bayes-Methode auf das gleiche Ergebnis kommt wenn man folgendermassen vorgeht: D in n Pakete $D_1,\dots,D_n$ zerlegen, mit Prior-Verteilung $p(\theta)$ und D_1 Posterior-Verteilung $p(\theta|D_1)$ berechnen, mit $p(\theta|D_1)$ als neuer Prior-V. und D_2 wieder dazugehörige Posterior-Verteilung berechnen welche genau $p(\theta|D_1 \cup D_2)$ entspricht, usw

im Grenzwert von unendlich viel Daten liegt die Posteriorverteilung auf den MAP-Parameter konzentriert, welcher dann dasmeisten mit dem ML-Parameter zusammenfällt, weil der Einfluss der Prior-Verteilung vom Gewicht der Daten (über Likelihood) vollständig überstrahlt wird
→ bei wenigen Daten kann Unterschied der 3 Lernmethoden jedoch sehr gross sein!

Bernoulli-Modelle

→ Basisfedn für flexiblere & komplizierte Modelle

Bernoullie-Verteilung: $p(x|\theta) = \text{Bern}_\theta(x) = \theta^x(1-\theta)^{1-x}$, $x \in \{0,1\}$, $\theta \in [0,1]$

erschöpfende Statistik der Datenmenge $D = \{x_1, \dots, x_n\}$: $n_1 = |\{x \in D \mid x=1\}|$, $n_0 = |\{x \in D \mid x=0\}|$

$L(\theta|D) = p(D|\theta) = \theta^{n_1}(1-\theta)^{n_0} \Rightarrow \ln(\cdot) \Rightarrow \text{maximieren} \Rightarrow \theta_{ML} = \frac{n_1}{n_0+n_1}$

Posterior-V.: $p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{\theta^{n_1}(1-\theta)^{n_2}p(\theta)}{\int_0^1 \theta^{n_1}(1-\theta)^{n_2}p(\theta)d\theta}$

Definition konjugierte Prior-Verteilung:

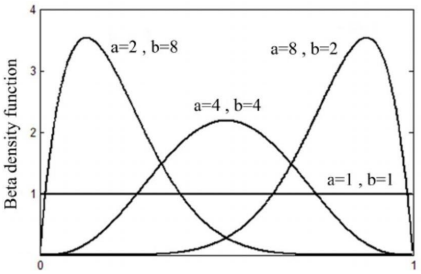
> Familie $\mathcal{K}$ von Verteilungen $p(\vec{\theta})$ über die Parameter $\vec{\theta}$ eines prob. Modells $p(\vec{x}|\vec{\theta})$ heisst **konjugiert** zu $p(D|\vec{\theta})$ falls für alle Daten D und alle Verteilungen $p \in \mathcal{K}$ auch induzierte Posterior-Verteilung $p(\vec{\theta}|D)$ eine Verteilung der Familie ist, also $p(\vec{\theta}|D) \in \mathcal{K}$

Likelihood der Bern.-V.:

> Beta-Prior-V $p(\theta) = \text{Beta}_{a,b}(\theta)$ ist konjugiert zur Likelihood der Bernoulli-V. $p(x|\theta) = \text{Bern}_\theta(x)$ und für Daten D mit Statistik (n_1, n_0) erhalten wir die Posterior-V. $p(\theta|D) = \text{Beta}_{a+n_1, b+n_2}(\theta)$

Beta-Verteilung:

$$\text{Beta}_{a,b}(\theta) = \frac{\theta^{a-1}(1-\theta)^{b-1}}{B(a,b)} \quad \text{wobei} \quad \theta \in [0,1], \quad a \,\&\, b > 0$$



Normalisierung im Nenner heisst Beta-Funktion:

$$B(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)} \quad \text{mit} \quad \Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt$$

Kenngrössen:

$$\text{mean} = \frac{a}{a+b}, \quad \text{variance} = \frac{\text{mean}(1-\text{mean})}{a+b+1}, \quad \text{mode} = \frac{a-1}{a+b-2} \ (\text{existiert nur falls } a>1, b>1), \quad \text{Stärke} = a+b$$

mit $p(\theta) = \text{Beta}_{a,b}(\theta)$ als Prior-Verteilung erhält man für die Posterior-V.:

$$p(\theta|D) = \frac{1}{p(D)}p(D|\theta)p(\theta) = \frac{1}{p(D)}\theta^{n_1}(1-\theta)^{n_0}\frac{1}{B(a,b)}\theta^{a-1}(1-\theta)^{b-1} \propto \text{Beta}_{a+n_1, b+n_2}$$

zwei Wahrscheinlichkeitsdichten die proportional zueinander sind müssen jedoch notwendigerweise identisch sein (wegen Integral $\doteq 1$) ■

prior $\xrightarrow{\text{data}}$ posterior
$\text{Beta}_{a,b} \xrightarrow{n_1,n_0} \text{Beta}_{a+n_1,b+n_0}$

dieser Übergang legt es nahe die Parameter a, b als Pseudo-Beobachtungen von 1-Werten respektive 0-Werten zu interpretieren

Stärke $a+b$ der Prior-V. ist Mass für relativen Einfluss der Prior-V. auf Posterior-V., je stärker der Prior (bei gleich viel Daten) desto mehr gleicht sich Posterior-V. der Prior-V. an

Varianz nimmt bei Übergang prior $\xrightarrow{D}$ posterior i.A. ab, was als Zunahme des Wissens über Parameter θ interpretiert werden kann

Voraussage-Verteilung liefert $p(x|a,b) = \int \text{Bern}(x|\theta)\text{Beta}(\theta|a,b)d\theta = \text{Bern}(x|\theta_{\text{prior}})$ wobei:

$$\theta_{\text{prior}} = p(x=1|a,b) = \int \text{Bern}(x=1|\theta)\text{Beta}(\theta|a,b)d\theta = \int \theta\,\text{Beta}(\theta|a,b)d\theta = \text{Mean}(\text{Beta}_{a,b}) = \frac{a}{a+b}$$

MAP-Methode wählt $\theta_{MAP} = \text{argmax}_\theta\, p(x|D) = \text{Mode}(\text{Beta}_{a+n_1,b+n_0}) = \frac{a+n_1-1}{a+n_1+b+n_0-2}$

Bayessche Voraussage-Verteilung liefert $\text{Bern}(x|\theta_{\text{mean}})$ mit $\theta_{\text{mean}} = \frac{a+n_1}{a+n_1+b+n_0}$

Bayessche Methode interpoliert zwischen Prior-Voraussage (nur Prior, keine Daten) und der ML-Voraussage (kein Prior, nur Daten):

$$\theta_{mean} = \lambda\,\theta_{prior} + (1-\lambda)\,\theta_{ML} \quad \text{mit Mischungsparameter } \lambda = \frac{1}{1+\frac{n_1+n_0}{a+b}}$$

Voraussage ohner Sequenz

falls mehrere, m Werte vorausgesagt werden sollen → Binomialverteilung

- θ_{ML} und θ_{MAP} können direkt in $Bin_{m,\theta}(y) = \binom{m}{y}\theta^y(1-\theta)^{m-y}$ eingesetzt werden
- mit Bayes ergibt sich $\int Bin(y|m,\theta)\,Beta(\theta|c,d)\,d\theta = \binom{m}{y}\frac{B(y+c,\,m-y+d)}{B(c,d)} =: BetaBin(y|c,d,m)$

Multitask Bernoulli

Ziel: r Parameter θ_k von r ähnlich strukturierten, nicht notwendigerweise gleich grossen Populationen mit Datensätzen D_k mit Statistiken $(n_{k,1}, n_{k,0})$ bestimmen

$$p(D_k, \theta_k | a, b) = p(D_k|\theta_k)\,p(\theta_k|a,b) = \theta_k^{n_{k,1}}(1-\theta_k)^{n_{k,0}}\,Beta(\theta_k|a,b) \quad \text{Modell}$$

$$p(\theta_k|D_k,a,b) = Beta(\theta_k|a+n_{k,1},\,b+n_{k,0}) \quad \text{Posterior-V.}$$

hierarchischen Ansatz: (Prior, Hyperprior: Prior über Prior)

$$p(D,\vec{\theta},a,b) = p(D|\vec{\theta})\,p(\vec{\theta}|a,b)\,p(a,b) = \left(\prod_{k=1}^{r}\theta_k^{n_{k,1}}(1-\theta_k)^{n_{k,0}}\,Beta(\theta_k|a,b)\right)p(a,b)$$

gemeinsame Verteilung

mit $D = \bigcup_{k=1}^{r} D_k$, $\vec{\theta} = (\theta_1,\dots,\theta_r)$

nach Bayes-Methode können wir mittels Marginalisierung die Posterior-V. $p(\theta_k|D)$ berechnen, welche analytisch leider schwierig zu berechnen ist

Vorteil ist, dass statistische Stärke von Populationen mit vielen Daten auf Populationen mit wenig Daten transferiert werden kann, da die θ_k nun abhängig sind

Multinoulli-Modelle

Kategorische / Multinoulli-Verteilung: $p(k|\vec{\theta}) = Cat_{\vec{\theta}}(k) = \theta_k$ — Verallgemeinerung Bernoulli-V.

mit $k \in \{1, \dots, K\}$, $\vec{\theta} = (\theta_1, \dots, \theta_K) \in \mathbb{R}_+^K$ (welcher $\sum \theta_k = 1$ erfüllt)

erschöpfende Statistik der Datenmenge $D = \{x_1, \dots, x_n\}$: $\vec{n} = (n_1, \dots, n_K)$
mit $n_k = |\{x \in D \mid x = k\}|$

$L(\vec{\theta}|D) = p(D|\vec{\theta}) = \prod_{k=1}^{K} \theta_k^{n_k} \Rightarrow \ln(\cdot) \Rightarrow$ maximieren mit Lagrange-Multiplikatoren (→ Nebenbedingung: $\sum \theta_k = 1$)

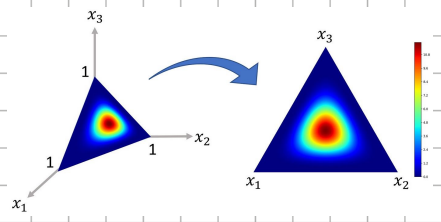
$\Rightarrow \vec{\theta}_{ML} = \frac{\vec{n}}{\|\vec{n}\|_1} = \frac{(n_1, \dots, n_K)}{n_1 + \dots + n_K}$ relative Häufigkeit

Dirichlet-Verteilung:

$$Dir_{\vec{\alpha}}(\vec{\theta}) = \frac{\prod_{k=1}^{K} \theta_k^{\alpha_k - 1}}{B(\vec{\alpha})} = \frac{\prod_{k=1}^{K-1} \theta_k^{\alpha_k - 1} \left(1 - \sum_{k=1}^{K-1} \theta_k\right)^{\alpha_K - 1}}{B(\vec{\alpha})}$$

Verallgemeinerung Beta-Verteilung

mit $B(\vec{\alpha}) = \prod_{k=1}^{K} \Gamma(\alpha_k) / \Gamma(\sum_{k=1}^{K} \alpha_k)$ Verallgemeinerung Beta-Funktion



Kenngrössen:

$$mean_k = \frac{\alpha_k}{\|\vec{\alpha}\|_1}, \quad variance_k = \frac{mean_k(1 - mean_k)}{\|\vec{\alpha}\|_1 + 1}, \quad mode_k = \frac{\alpha_k - 1}{\|\vec{\alpha}\|_1 - K} \text{ (existiert nur falls } \alpha_k > 1 \; \forall k), \quad Stärke = \|\vec{\alpha}\|_1$$

Dirichlet-Prior-V. $p(\vec{\theta}) = Dir_{\vec{\alpha}}(\vec{\theta})$ ist konjugiert zur Likelihood der Multinoulli-V. $p(x|\vec{\theta}) = Cat_{\vec{\theta}}(x)$ und für Daten D mit Statistik $\vec{n} = (n_1, \dots, n_K)$ erhalten wir die Posterior-V. $p(\vec{\theta}|D) = Dir_{\vec{\alpha}+\vec{n}}(\vec{\theta})$

$$\vec{\theta}_{MAP} = \frac{\vec{\alpha} + \vec{n} + \vec{1}_K}{\|\vec{\alpha} + \vec{n}\|_1 - K}$$

Voraussage-Verteilung: $Cat(x|\vec{\theta}_{Prior})$ wobei $\vec{\theta}_{Prior} = \frac{\vec{\alpha}}{\|\vec{\alpha}\|_1}$

$$\vec{\theta}_{mean} = \frac{\vec{\alpha} + \vec{n}}{\|\vec{\alpha} + \vec{n}\|_1}$$

Naive-Bayes-Klassifikator

Annahmen: $t \in \{1,\dots,C\}$, $\vec{x} \in \mathcal{X} = \{1,\dots,K_1\} \times \dots \times \{1,\dots,K_M\}$, $D = \{(t_k, \vec{x}_k) \mid 1 \le k \le n_D\}$

Anzahl verschiedener möglicher $\vec{x}$-Werte: $|\mathcal{X}| = \prod_{m=1}^{M} K_m$

Klassen-Prior-Verteilung: $p(t \mid \vec{\theta}_0) = \mathrm{Cat}_{\vec{\theta}_0}(t)$ — modelliert Wissen über Verteilung der Klassenlabels

auf Klasse t konditionierte kategorische Verteilung: $p(\vec{x} \mid t, \vec{\theta}_t)$

gemeinsame Verteilung der (Ausgabe, Eingabe)-Paare: $p(t, \vec{x} \mid \vec{\theta} = (\vec{\theta}_0, \vec{\theta}_1, \dots, \vec{\theta}_C)) = p(\vec{x} \mid t, \vec{\theta}_t)\, p(t \mid \vec{\theta}_0)$

trainiertes Modell: $p(t \mid \vec{x}_n) = \frac{p(t, \vec{x}_n \mid D)}{p(\vec{x}_n \mid D)} = \frac{p(t, \vec{x}_n \mid D)}{\sum_{t \in C} p(t, \vec{x}_n \mid D)}$

für die $p(\vec{x} \mid t, \vec{\theta}_t)$ mit $t = 1,\dots,C$ bräuchten wir $C \cdot (\prod_{m=1}^{M} K_m - 1)$ modellabhängige Komponenten in $[0,1]$
→ dies übersteigt schon bei einfachen Problemen die Eddington-Number von $\approx 10^{80}$ ($\#p^+$ im Universum)

daher behandelt man die einzelnen Komponenten x_m als unabhängig („naiv"):

$$p(\vec{x} \mid t, \vec{\theta}_t) = \prod_{m=1}^{M} p(x_m \mid t, \vec{\theta}_{t,m}) = \prod_{m=1}^{M} \mathrm{Cat}_{\vec{\theta}_{t,m}}(x_m)$$

Anzahl $\sum_{m=1}^{M} (K_m - 1)$ unabh. Parametern $\vec{\theta}_t = (\vec{\theta}_{t,1}, \dots, \vec{\theta}_{t,M})$ ist linear in Dimension M des Eingabevektors

Naive-Bayes-Klassifikator:

$$p(t, \vec{x} \mid \vec{\theta}) = p(\vec{x} \mid t, \vec{\theta}_t)\, p(t \mid \vec{\theta}_0) = \prod_{m=1}^{M} p(x_m \mid t, \vec{\theta}_{t,m})\, p(t \mid \vec{\theta}_0) = \prod_{m=1}^{M} \mathrm{Cat}_{\vec{\theta}_{t,m}}(x_m) \cdot \mathrm{Cat}_{\vec{\theta}_0}(t)$$

mit Parametern $\vec{\theta} = (\vec{\theta}_0, \vec{\theta}_1, \dots, \vec{\theta}_C) = (\vec{\theta}_0, \vec{\theta}_{1,1}, \dots, \vec{\theta}_{1,M}, \dots, \vec{\theta}_{C,1}, \dots, \vec{\theta}_{C,M})$

ML-Methode:

$$p(D \mid \vec{\theta}) = \prod_{(t,\vec{x}) \in D} \left(p(\vec{x} \mid t, \vec{\theta}_t)\, p(t \mid \vec{\theta}_0) \right) = \prod_{(t,\vec{x}) \in D} p(\vec{x} \mid t, \vec{\theta}_t) \prod_{(t,\vec{x}) \in D} p(t \mid \vec{\theta}_0) = \prod_{t=1}^{C} \prod_{m=1}^{M} \prod_{\vec{x} \in D_t} \mathrm{Cat}_{\vec{\theta}_{t,m}}(x_m) \cdot \prod_{(t,\vec{x}) \in D} \mathrm{Cat}_{\vec{\theta}_0}(t)$$

$$\Rightarrow \vec{\theta}_{ML,0} = \vec{n}_0 / \|\vec{n}_0\|_1, \quad \vec{\theta}_{ML,t,m} = \vec{n}_{t,m} / \|\vec{n}_{t,m}\|_1$$

Bayessche Methode:

Prior $p(\vec{\theta} \mid \vec{\alpha}) = \prod_{t=1}^{C} \prod_{m=1}^{M} \mathrm{Dir}_{\vec{\alpha}_{t,m}}(\vec{\theta}_{t,m}) \cdot \mathrm{Dir}_{\vec{\alpha}_0}(\vec{\theta}_0)$ ist konjugiert zu Likelihood d. N-B-K

$\Rightarrow$ Posterior: $p(\vec{\theta} \mid \vec{\alpha} + \vec{n}) = \prod_{t=1}^{C} \prod_{m=1}^{M} \mathrm{Dir}_{\vec{\alpha}_{t,m} + \vec{n}_{t,m}}(\vec{\theta}_{t,m}) \cdot \mathrm{Dir}_{\vec{\alpha}_0 + \vec{n}_0}(\vec{\theta}_0)$

Modellierung beaufsichtigter Lernprobleme

Verlustmasse und end-zu-end Lernen

MSE nicht robust gegen Ausreisser resp. nicht optimal für bedingte Ausgabeverteilungen mit heavy tails
→ falls man grosse Abweichungen weniger stark gewichten möchte, kann man auch mittlere absolute Abweichung als Verallgemeinerungsfehler wählen

end-zu-end: direkt Voraussagefunktion lernen (nicht Verteilung der Ausgabe gegeben Eingabe)

Verallgemeinerungsproblem

Konstruktion eines prob. Modell $p_m(\vec{t}|\vec{x},\vec{\theta})$ für die den Daten zugrundeliegende Verteilung $p(\vec{t}|\vec{x})$

Unterschiede zum end-zu-end-Ansatz:
- i.A. einfacher vorhandenes Wissen / Hypothesen über Problem kontrolliert und dosiert ins Modell einfliessen zu lassen
- erlaubt speziell entwickelte Algorithmen für Dichte-Schätzung anstelle von allg. Optimierung ins Spiel zu bringen
- Verlustmass L geht nicht ins Verallgemeinerungsproblem ein

Entscheidungsproblem

basierend auf gelerntem prob. Modell p_m die bez. Verlustmass beste Voraussage-Funktion finden:

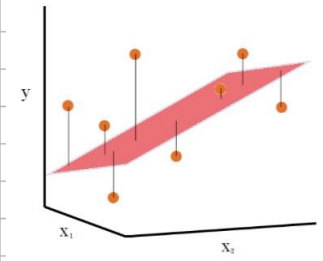
$$f_{m,opt}(\vec{x}) = \text{argmin}_{\vec{p}} \int L(\vec{t},\vec{p})\, p_m(\vec{t}|\vec{x})\, d\vec{t}$$

unter Annahme, dass Modell p_m exakt ist, ist dies die beste Voraussage-Funktion

weitere Konsequenzen der Aufteilung

Flexibilität, Modularität

Lineare Modelle für Regressionsprobleme



häufiger Bestandteil komplizierterer Modelle → „Arbeitspferd des m.L."

Familie von diskriminativen, parametrischen, probabilistischen Modellen

Lineares Regressionsmodell:

(→ (hier) als unabhängig von $\vec{x}$ angenommen (homoskedastische Regression))

$$p(t|\vec{x},\vec{w},\beta) = \mathcal{N}(t|\vec{\phi}(\vec{x})^T\vec{w},\beta^{-1}) \quad \text{mit} \quad \vec{\phi}(\vec{x})^T\vec{w} = \sum_{k=0}^{M} w_k \phi_k(\vec{x})$$

mit adaptiven Parametern $\vec{w} \in \mathbb{R}^{M+1}$, $\beta > 0$

vielfach verwendete generische Familien von Basisfunktionen: Monome, Spline-Funktionen, Gaussche Funktionen (weniger exakt lokalisiert wie Splines), Fourier-Basen (lokalisiert im Frequenzbereich), Wavelet-Basen (co-lokalisiert in Zeit- und Frequenzbereich)

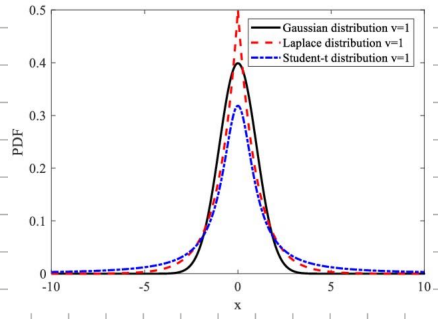
klassische Regression: $\phi_0(\vec{x}) = 1$, $\phi_k(\vec{x}) = x_k$ $\Rightarrow$ $\vec{\phi}(\vec{x})^T\vec{w} = \sum_{k=1}^{M} w_k x_k + w_0$ Ausgabe hängt affin von Eingabe ab

eine Familie von Basisfunktionen induziert eine Transformation $\vec{x} \to \vec{\phi}(\vec{x}) = [\phi_0(\vec{x}), \phi_1(\vec{x}), \dots, \phi_M(\vec{x})]^T$ des Eingaberaums $\Rightarrow$ jedes lineare Modell kann als klassische Regression auf dem Raum der transformierten Eingaben $\vec{\phi}(\vec{x})$ aufgefasst werden resp. Transformation gefolgt von klassischer Regression

transformierte Eingabegrössen $\vec{\phi}(\vec{x})$ nennt man auch **Features**

Ziel der **Feature Extraktion** ist lineare Berechnung der Ausgabe aus Features zu ermöglichen/vereinfachen

alternative Ausgabeverteilungen



viele der geschlossenen Formeln hängen von Annahme der Normalverteilung als Ausgabeverteilung ab

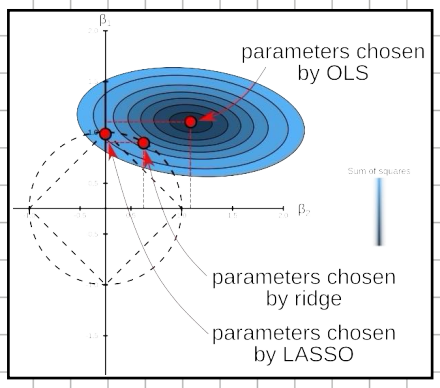
zur Rechtfertigung kann man eine Form des zentralen Grenzwertsatzes oder auch Prinzip der Entropie-Maximierung unter Vorgabe der ersten zwei Momente heranziehen

N ist jedoch nicht robust gegen Ausreisser, in solchen Fällen sollte mit Vorteil eine andere Verteilung mit under Wahrscheinlichkeitsmasse in Enden, z.B. Laplace- oder Student-t-Verteilung, gewählt werden

Standardisierung der Daten

Zentrierung der Ausgabe und Features in Null zusammen mit Skalierung der Varianz der Features auf Eins
→ begünstigt Vergleichbarkeit der Sensitivität der Parameter $\vec{w}$
→ ermöglicht Wahl einer uniformen Stärke der Prior-Verteilung in allen Dimensionen

Maximum Likelihood Lernen



$$\text{Log-Likelihood} = \ln p(\vec{t} \mid \underline{X}, \vec{w}, \beta) = \sum_{j=1}^{N} \ln \mathcal{N}(t_j \mid \vec{\phi}(\vec{x}_j)^T \vec{w}, \beta^{-1})$$

$$= \frac{N}{2}\ln(\beta) - \frac{N}{2}\ln(2\pi) - \frac{\beta}{2}\sum_{j=1}^{N}(t_j - \vec{\phi}(\vec{x}_j)^T \vec{w})^2$$

Maximierung der Log-Likelihood ergibt

$$\vec{w}_{ML} = \left(\underline{\Phi}(\underline{X})^T \underline{\Phi}(\underline{X})\right)^{-1} \underline{\Phi}(\underline{X})^T \vec{t}$$

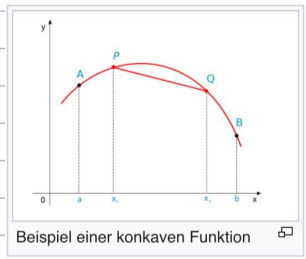
$$\beta^{-1} = \frac{1}{N}\left\| \vec{t} - \underline{\Phi}(\underline{X})\vec{w}_{ML} \right\|^2$$

mittlere quadratische Abweichung der Lösung des Least-Squares-Problems

wobei für die Systemmatrix (Designmatrix) $\Phi_{j,k}(\underline{X}) = \phi_{k-1}(\vec{x}_j)$ gilt, also

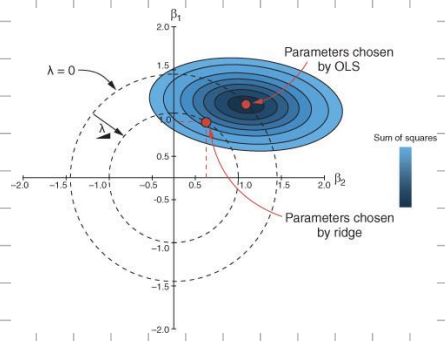
$$\underline{\Phi}(\underline{X}) = \begin{bmatrix} \phi_0(\vec{x}_1) & \phi_1(\vec{x}_1) & \cdots & \phi_m(\vec{x}_1) \\ \vdots & \vdots & & \vdots \\ \phi_0(\vec{x}_N) & \phi_1(\vec{x}_N) & \cdots & \phi_m(\vec{x}_N) \end{bmatrix} \in \mathbb{R}^{N \times (m+1)}$$

Likelihood bei linearen Modellen konkav
→ Vorteil gegenüber komplexeren Modellen!



Beispiel einer konkaven Funktion

Maximum Posterior Lernen



wir betrachten β als externen Parameter und wählen motiviert durch Konjugiertheit

$$p(\vec{w}|\vec{\alpha}) = \mathcal{N}(\vec{w}|\vec{0}, \underline{D}_{\vec{\alpha}}^{-1}) = \prod_{k=0}^{M} \mathcal{N}(w_k|0, \alpha_k^{-1})$$ Prior

als Prior-Verteilung auf dem Raum der Gewichte $\vec{w} \in \mathbb{R}^{M+1}$

Kovarianzmatrix wird diagonal gewählt, weil Gewichte der einzelnen Basisfunktionen nicht a priori korreliert sein sollen und der Präzisions-Hyperparameter $\vec{\alpha} \in \mathbb{R}_+^{M+1}$ kontrolliert Stärke, mit welcher Gewichte in Nähe von $\vec{0}$ gezwungen werden

$$\text{Log von Likelihood} \times \text{Prior} = \ln\left(p(\vec{t}|\underline{X}, \vec{w}, \beta)\, p(\vec{w}|\vec{\alpha})\right) = C - \frac{\beta}{2}\sum_{j=1}^{N}\left(t_j - \vec{\phi}(\vec{x}_j)^T\vec{w}\right)^2 - \frac{1}{2}\sum_{k=0}^{M}\alpha_k w_k^2$$

Maximierung davon in $\vec{w}$ liefert

$$\vec{w}_{MAP} = \left(\underline{\Phi}(\underline{X})^T\underline{\Phi}(\underline{X}) + \underline{D}_{\vec{\lambda}}\right)^{-1}\underline{\Phi}(\underline{X})^T\vec{t}$$

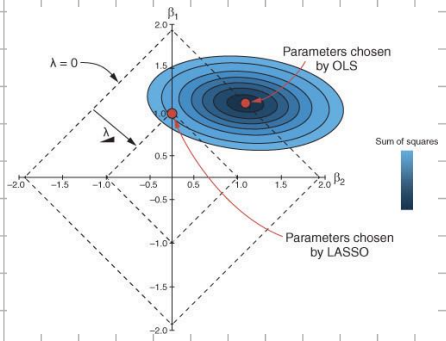
mit $\vec{\lambda} = \vec{\alpha}/\beta$

klassische Ridge-Regression: MAP-Methode mit einparametrigen Prior-V. mit Kovarianz $\alpha^{-1}\underline{E}_{M+1}$

bei gegebener Präzision β kontrolliert der Hyper-Parameter $\vec{\alpha}$ (oder äquivalent $\vec{\lambda}$) die Flexibilität des Modells und dieser wird oft numerisch optimiert (z.B. mit Kreuz-Validation)

für $\vec{\alpha} \to \vec{0}$ konvergiert $\vec{w}_{MAP} \to \vec{w}_{ML}$

Lasso-Regression: faktorisierte Laplace-V. als Prior
→ besser als Ridge falls aus grosser Familie von Basisfunktionen die besten ausgewählt werden sollen, da sie die Eigenschaft hat dass das damit gelernte Modell viele Koeffizienten exakt bei Null hat (sparsity enforcing)
→ man kann den die Stärke der Prior-V. kontrollierenden Faktor λ als Lagrange-Multiplikator eines Minimierungsproblems mit der Nebenbedingung $\sum_{k=0}^{M}|w_k| = C$ auffassen und dann aufgrund der Geometrie der Nebenbedingung argumentieren, dass in der Lösung $\vec{w}$ viele Nulleinträge vorkommen sollten



falls man Gauss-Prior zur Selektion von Features verwenden möchte, wird die allgemeine Form mit gleich vielen Hyperparametern $\vec{\alpha}$ wie Gewichten $\vec{w}$ und nicht die Ridge-Regression benötigt
→ bedingt, dass auch die Hyperparameter $\vec{\alpha}$ adaptiert werden
→ automatische-Relevanz-Bestimmung (ARD)

Bayessches Lernen → für sequentielles Lernen sowie permanente Adaptivität geeignet

Likelihoodhood: $p(\vec{t}\,|\,\vec{w}, \vec{X}, \beta) = \mathcal{N}(\vec{t}\,|\,\underline{\Phi}(X)\vec{w}, \beta^{-1}\underline{E}) = \prod_{j=1}^{N} \mathcal{N}(t_k\,|\,\vec{\phi}(\vec{x}_j)^T\vec{w}, \beta^{-1})$

↳ hier extern, Grossmutter fragen

Prior: $p(\vec{w}\,|\,\vec{m}_0, \underline{S}_0) = \mathcal{N}(\vec{w}\,|\,\vec{m}_0, \underline{S}_0)$

Posterior:

$$p(\vec{w}\,|\,\vec{t}, \underline{X}, \eta) = \mathcal{N}(\vec{w}\,|\,\vec{m}_N, \underline{S}_N)$$

mit $\vec{m}_N = \underline{S}_N(\beta\,\underline{\Phi}(X)^T\vec{t} + \underline{S}_0^{-1}\vec{m}_0)$, $\underline{S}_N = (\underline{S}_0^{-1} + \beta\,\underline{\Phi}(X)^T\underline{\Phi}(X))^{-1}$

marginale Likelihood:

$$p(\vec{t}\,|\,\underline{X}, \beta, \vec{m}_0, \underline{S}_0) = \mathcal{N}\left(\vec{t}\,|\,\underline{\Phi}(X)\vec{m}_0, \beta^{-1}\underline{E} + \underline{\Phi}(X)\underline{S}_0\underline{\Phi}(X)^T\right)$$

Posterior-Voraussage-Verteilung für $p(t\,|\,\vec{x}, \vec{w}, \beta) = \mathcal{N}(t\,|\,\vec{\phi}(\vec{x})^T\vec{w}, \beta^{-1})$ und $p(\vec{w}\,|\,\vec{m}, \underline{S}) = \mathcal{N}(\vec{w}\,|\,\vec{m}, \underline{S})$:

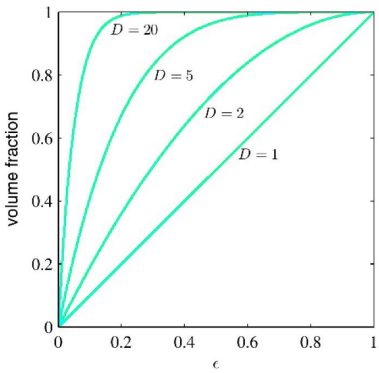
$$p(t\,|\,\vec{x}, \beta, \vec{m}, \underline{S}) = \int p(t\,|\,\vec{x}, \vec{w}, \beta)\,p(\vec{w}\,|\,\vec{m}, \underline{S})\,d\vec{w} = \mathcal{N}(t\,|\,m(\vec{x}), s(\vec{x}))$$

mit $m(\vec{x}) = \vec{\phi}(\vec{x})^T\vec{m}$ und $s(\vec{x}) = \beta^{-1} + \vec{\phi}(\vec{x})^T\,\underline{S}\,\vec{\phi}(\vec{x})$

die Bayessche Voraussage-Varianz $s(\vec{x}) = \beta^{-1} + \vec{\phi}(\vec{x})^T\,\underline{S}_N\,\vec{\phi}(\vec{x})$ wird kleiner, wenn Eingabe $\vec{x}$ nahe bei Eingabewerten der Trainingsdaten liegt und wird allgemein kleiner wird, wenn Anzahl N der der Trainingsdaten grösser wird

falls $\vec{x}$ an Rändern des Supports aller Basisfunktionen liegt → Kovarianz klein weil alle Features fast null sind, auch wenn $\vec{x}$ weit weg von Trainingsdaten (schwer zu behebendes Problem von diskriminativen Modellen)

Grenzen Linearer Modelle



Basisfunktionen lassen sich für hochdimensionale Eingaben $\vec{x}$ nicht einfach generisch wählen, weil (vereinfacht gesagt) der Träger jeder in den euklidischen Raum lokalisierten Basisfunktion nicht in der Nähe der Daten liegt

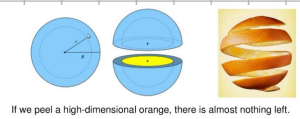
motiviert durch unsere Erfahrung/Intuition könnte man versuchen ein allgemeines, generisches Regressionsmodell zu konstruieren, indem man eine geeignet flexible Familie von Basisfunktionen so wählt dass deren Träger den Raum der Eingaben $\vec{x}$ gleichmässig überdecken

falls Eingaben $\vec{x}$ in zentriertem, d-dimensionalen Würfel enthalten sind wir dazu jede Achse gleichmässig in h Stücke schneiden, ist Anzahl Basis-Funktionen $\propto h^d$ und somit hängt auch die benötigte Menge an Daten exponentiell von d ab (-> keine gute Idee)

Curse of Dimensionality:

wir betrachten Gleichverteilung auf d-dimensionalen Einheitsball $B = \{\vec{x} \in \mathbb{R}^d \mid \|\vec{x}\| \leq 1\}$ und berechnen Anteil des Volumens (oder äquivalent W'keit der Daten) welche in der äussersten Shale $S_\epsilon = \{\vec{x} \in \mathbb{R}^d \mid 1-\epsilon \leq \|\vec{x}\| \leq 1\}$ der Dicke ϵ enthalten ist:

$$V(\epsilon) = \frac{C(d)\,1^d - C(d)\,(1-\epsilon)^d}{C(d)\,1^d} = 1 - (1-\epsilon)^d$$


If we peel a high-dimensional orange, there is almost nothing left.

-> bei höheren Dimensionen liegt der Anteil $V(\epsilon)$ auch schon für kleine ϵ nahe bei 1
-> fast alle Daten liegen in dünner Shale am Rande der Kugel

wenn wir folglich den Raum mithilfe der euklidischen Metrik in lokale Träger aufteilen, liegen gleichverteilte Daten mit hoher W'keit am Rand eines Trägers und diese Lokalität ist n. geeignet eine vorhandene Glattheit der Ausgabe t bezüglich der Eingabe $\vec{x}$ effizient auszunutzen

in realen Regressionsproblemen ist andererseits Verteilung der Eingaben $\vec{x} \in \mathbb{R}^d$ meistens sehr weit von einer Gleichverteilung entfernt, die $\vec{x}$ liegen meist sogar konzentriert in der Nähe einer (i.A. nichtlinearen) Untermannigfaltigkeit $M \subset \mathbb{R}^d$ viel kleinerer Dimension und zusätzlich hängt t häufig nur von einigen möglichen Veränderungsrichtungen der Eingabe $\vec{x}$ sensitiv ab

diese Untermannigfaltigkeit und darauf die lokalen sensitiven Richtungen (für die Abhängigkeit der Ausgabe von der Eingabe) definieren ein Distanzmass (Metrik) das i.A. von der euklidischen Distanz des Eingaberaumes $\mathbb{R}^d$ ziemlich verschieden ist

Familie von Basisfunktionen deren Träger auf der Untermannigfaltigkeit M liegen und lokal bezüglich dieses Distanzmasses gewählt sind kann prinzipiell die Glattheit der Ausgabe t viel effizienter ausnutzen und bis zu einem gewissen Grade den Fluch der Dimensionalität vermeiden
-> leider kann eine solche Familie von Basisfunktionen nur in seltenen Fällen einfach von Hand konstruiert werden!

Lineare Modelle für Klassifikationsprobleme

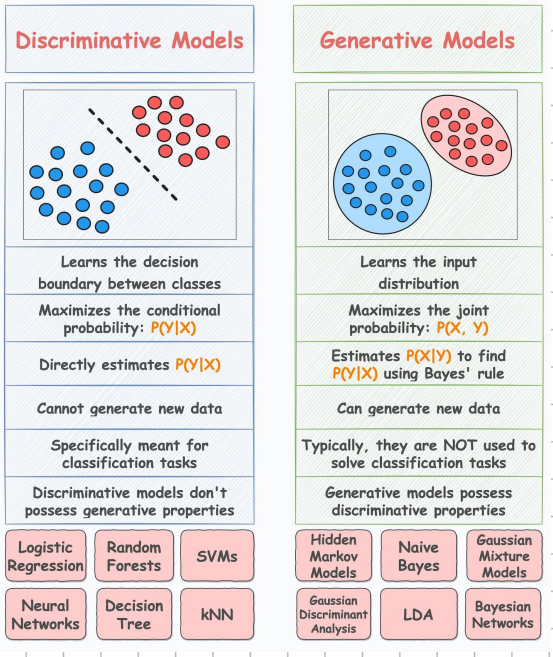
2 grundlegende lineare Modelle für beaufsichtigtes Klassifikationsproblem:

- generativ → $p(\vec{t}, \vec{x})$ $\left(= p(\vec{t}|\vec{x})p(\vec{x}) = p(\vec{x}|\vec{t})p(\vec{t})\right)$
- diskriminativ → $p(\vec{t}|\vec{x})$

bei generativ manchmal natürlicher $p(\vec{t}|\vec{x})p(\vec{x})$ oder $p(\vec{x}|\vec{t})p(\vec{t})$ zu verwenden

da bedingte Verteilung $p(\vec{t}|\vec{x})$ aus jedem generativen Modell $p(\vec{t}, \vec{x})$ berechnet werden kann, sind generative Modelle umfassender und damit potentiell schwieriger zu lernen als diskriminative Modelle

für Entscheidungsproblem nur $p(\vec{t}|\vec{x})$ benötigt, welche mit beiden Modellarten verfügbar ist



Kenntnis von $p(\vec{x})$ hat insbesondere den Vorteil, dass damit unwahrscheinliche Eingaben $\vec{x}$, für welche das Modell nicht trainiert wurde, detektiert werden können (novelty detection)

generativ

generatives Modell $p(\vec{x}, t)$ für beaufsichtigtes Klassifikationsproblem:

Klassen-Prior-V.: $p(t)$, oft als diskrete $p(t) = Cat_{\vec{\theta}_0}(t)$ modelliert

Klassen-konditionierte-V.: $p(\vec{x}|t)$

Klassen-Posterior-V.: $p(t|\vec{x}) = \frac{p(\vec{x}|t)\,p(t)}{p(\vec{x})} = \frac{p(\vec{x}|t)\,p(t)}{\sum_{j=1}^{C} p(\vec{x}|j)\,p(j)}$

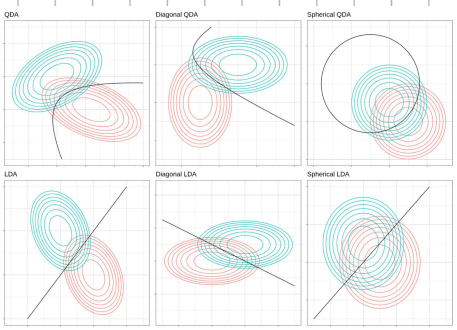
Klassifikation: $k = \text{argmax}_t\, p(t|\vec{x}) = \text{argmax}_t\, p(\vec{x}|t)\,p(t)$

Entscheidungsbereiche: $R_k = \{\vec{x} \mid k = \text{argmax}_t\, p(t|\vec{x})\}$

Achtung: Bezeichnungen Prior / Posterior beziehen sich in diesem Zusammenhang nicht auf die Parameter des Modells!

verschiedene generative Klassifikationsmodelle unterscheiden sich in der Form der Klassen-Bedingten-Verteilung $p(\vec{x}|t)$

Gausssche Diskriminanten Analyse (GDA) generativ



Upper Row: QDA, Diagonal QDA, Spherical QDA. **Lower Row:** LDA, Diagonal LDA, Spherical LDA.

Klassen-Bedingte-V.: $p(\vec{x}|t,\vec{\mu}_t,\Sigma_t) = \mathcal{N}(\vec{x}|\vec{\mu}_t,\Sigma_t)$

Klassen-Prior-V.: $p(t|\vec{\pi}) = Cat_{\vec{\pi}}(t)$

generatives Modell: $p(t,\vec{x}|\vec{\theta}) = p(\vec{x}|t,\vec{\mu}_t,\Sigma_t)\,p(t|\vec{\pi})$

Aktivierung der Klasse k bei Eingabe $\vec{x}$:

$$\begin{aligned} a_k(\vec{x}) &= \ln(p(\vec{x}|k)p(k)) = \ln p(\vec{x}|k) + \ln p(k) \\ &= -\tfrac{1}{2}\left(d\ln(2\pi) + \ln(\det\Sigma_k) + (\vec{x}-\vec{\mu}_k)^T\Sigma_k^{-1}(\vec{x}-\vec{\mu}_k)\right) + \ln p(k) \\ &= -\tfrac{1}{2}\left(\vec{x}^T\Sigma_k^{-1}\vec{x} - 2\vec{x}^T\Sigma_k^{-1}\vec{\mu}_k + c_k\right) \end{aligned}$$

Softmax-Funktion:

$$\vec{s}: \mathbb{R}^c \to \{\vec{p}\in\mathbb{R}_+^c \mid p_1+\dots+p_c=1\}, \quad \vec{a} \mapsto \vec{s}(\vec{a}) = (s_1(\vec{a}),\dots,s_c(\vec{a}))^T \text{ wobei } s_k(\vec{a}) = e^{a_k}/\textstyle\sum_{j=1}^c e^{a_j}$$

transformiert Aktivierungen in Wahrscheinlichkeitsvektor

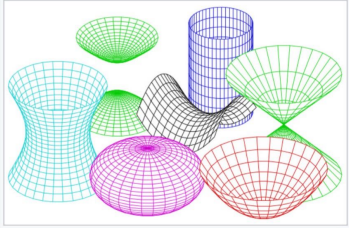
mit Softmax kann man für jedes gen. M. die Klassen-Posterior-V. $p(t|\vec{x})$ universell in der Form $p(t|\vec{x}) = s_t(\vec{a}(\vec{x}))$ schreiben wobei $s_t(\vec{a}(\vec{x})) = Cat_{\vec{s}(\vec{a}(\vec{x}))}(t)$ gilt

Entscheidungsbereiche: $R_k = \{\vec{x} \mid k = \operatorname{argmax}_j a_j(\vec{x})\}$

Grenze zwischen R_j und R_k ist in der Menge $\{\vec{x} \mid a_j(\vec{x}) = a_k(\vec{x})\}$ enthalten, welche durch die Quadrik

$$\vec{x}^T\Sigma_k^{-1}\vec{x} - 2\vec{x}^T\Sigma_k^{-1}\vec{\mu}_k + c_k = \vec{x}^T\Sigma_j^{-1}\vec{x} - 2\vec{x}^T\Sigma_j^{-1}\vec{\mu}_j + c_j$$

gegeben ist (Quadratische Diskriminanten Analyse (QDA))



Quadriken im dreidimensionalen Raum: ein- und zweischaliges Hyperboloid, Ellipsoid, hyperbolisches Paraboloid, Zylinder, elliptisches Paraboloid und Kegel (von links nach rechts)

falls $\Sigma_k = \Sigma_j = \Sigma$ gilt, erhält man eine Hyperebene: $\vec{x}^T\Sigma^{-1}(\vec{\mu}_k - \vec{\mu}_j) = \frac{c_k - c_j}{2}$

falls dies gefordert wird, spricht man von Linearer Diskriminanten Analyse (LDA)

↳ Parameter-Sharing / -Tying

ML-Methode

QDA: $\vec{\pi}_{ML} = \frac{\vec{n}}{\|\vec{n}\|_1}$, $\vec{\mu}_{ML,k} = \frac{\sum_{t_i=k}\vec{x}_i}{n_k}$, $\Sigma_{ML,k} = \frac{\sum_{t_i=k}(\vec{x}_i-\vec{\mu}_{ML,k})(\vec{x}_i-\vec{\mu}_{ML,k})^T}{n_k}$

LDA: $\Sigma_{ML} = \sum_{k=1}^c \frac{n_k}{N}\Sigma_{ML,k}$

MAP- und Bayes-Methode

LDA als Reduktion d. Flexibilität ohne Kunde → mit geeigneter Prior-V. in Kombination mit MAP kann Flexibilität auch feiner kontrolliert werden

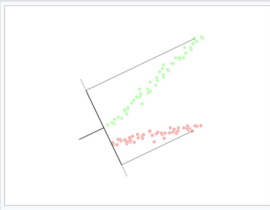
MAP und Bayes sind insbesondere für konj. Prior-V. berechenbar

Feature Extraktion und Dimensionsreduktion

anstatt Eingaben $\vec{x}$ direkt zu verwenden, kann es auch sinnvoll sein, diese zuerst geeignet zu transformieren und die für die Klassifikation relevanten Features herauszuarbeiten

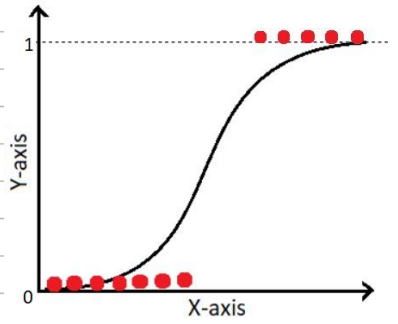
bei sehr hochdimensionalen Eingaben $\vec{x}$ kann Flexibilität der GDA auch durch Dimensionsreduktion begrenzt werden, wozu häufig linear auf Unterraum projiziert wird

→ zB Fishers-Lineare-Diskriminanten-Analyse



Fisher's Linear Discriminant visualised as an axis

Logistische Regression diskriminativ



diskr. Modell: $p(t|\vec{x}, \underline{W}) = Cat_{\vec{s}(\underline{W}\vec{\phi}(\vec{x}))}(t) = S_t(\underline{W}\vec{\phi}(\vec{x}))$

Basisfunktionen: $\vec{\phi}(\vec{x}) = (\phi_0(\vec{x}), \dots, \phi_M(\vec{x}))^T$

Parameter: $\underline{W} = (\vec{w}_1 \dots \vec{w}_C)^T = \begin{bmatrix} w_{1,0} & \dots & w_{1,M} \\ \vdots & & \vdots \\ w_{C,0} & \dots & w_{C,M} \end{bmatrix}$

für $\phi_0(\vec{x}) = 1$, $\phi_k(\vec{x}) = x_k$ resultiert affine Form der log. Regression

obwohl Symmetrie in Parametrisierung ansprechend ist, hat sie den Nachteil, dass $\underline{W}$ nicht identifizierbar sind; man kann einen beliebigen Vektor $\vec{0}^T$ zu allen Zeilen addieren und man verändert nichts an der Verteilung der Ausgabe
→ diese Überparametrisierung kann man eliminieren, indem man $\underline{W}$ geeignet normalisiert
→ häufig fordert man, dass letzte Zeile $= \vec{0}^T$ ist

mit Fehlklassifikations-Verlustmaß erhält man die Entscheidungsbereiche

$$R_t = \{\vec{x} \mid t = \operatorname{argmax}_k p(k|\vec{x}, \underline{W}) = \operatorname{argmax}_k \vec{w}_k^T \vec{\phi}(\vec{x})\}$$

wobei die Grenze zw. R_k und R_j in $R_k \cap R_j \subseteq \{\vec{x} \mid (\vec{w}_k - \vec{w}_j)^T \vec{\phi}(\vec{x}) = 0\}$ enthalten ist

Entscheidungsbereiche sind im Feature-Raum $\vec{y} = \vec{\phi}(\vec{x})$ konvex und Entscheidungsbereiche in Hyperebenen enthalten
→ bei affiner Basisfunktion gilt dies auch im Urbildraum

ML- und MAP-Parameter

Likelihood nicht mehr analytisch maximierbar (kurz gesagt weil $\ln()$ einer Summe nicht vereinfacht werden kann), jedoch immer noch konkav, was numerische Maximierung einfacher und robuster macht

falls Klassen separierbar sind, existiert jedoch kein Maximum und ML-Methode konvergiert gegen Funktion im Stile der Heavyside resp. einen unendlichen Overflow
→ kann mit MAP vermieden werden

falls Prior $p(\vec{w}|\alpha) = \mathcal{N}(\vec{w} \mid \vec{0}, \alpha^{-1}\underline{E})$ tritt zusätzlicher Summand $-\frac{\alpha}{2}\|\vec{w}\|_2^2$ auf wenn man logarithmiert (MAP)
→ dies wird L_2 regularisierte Maximum Likelihood Methode genannt

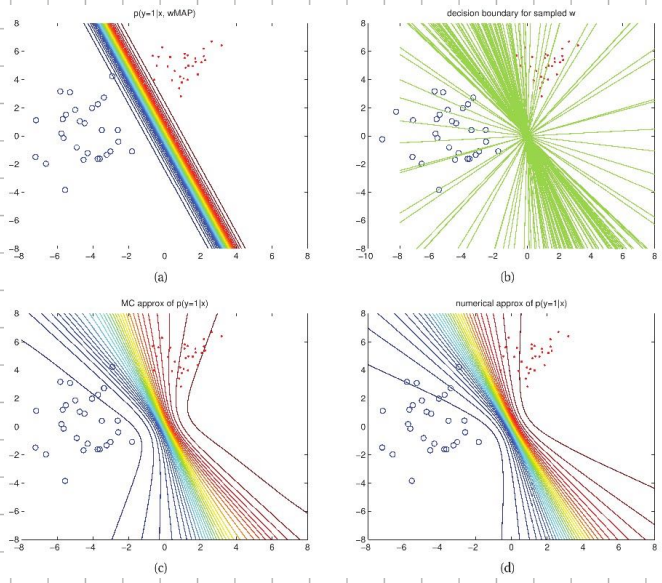
numerisch wird Optimierungsproblem der ML- oder MAP-Methode mit verschiedenen Standardmethoden gelöst (historisch relevant aber heute nicht mehr verwendet: IRLS (basiert auf Newton-Methode))

Bayessches Lernen

Posterior-V.: $p(\underline{w}|\vec{t},\underline{X}) = \frac{p(\vec{t}|\underline{X},\underline{w})\,p(\underline{w})}{p(\vec{t}|\underline{X})}$

$= \frac{p(\vec{t}|\underline{X},\underline{w})\,p(\underline{w})}{\int p(\vec{t}|\underline{X},\underline{w})\,p(\underline{w})\,d\underline{w}}$

hochdimensionales Integral numerisch heikel!



für keine Prior-V. geschlossen berechenbar (im Gegensatz zu lineaver Regression)
→ müssen approximieren

Monte-Carlo-Approximation einer Verteilung:

$p(\vec{x}) \simeq q(\vec{x}) = \frac{1}{|S|}\sum_{s\in S}\delta_s(\vec{x})$ S: Menge von Samples

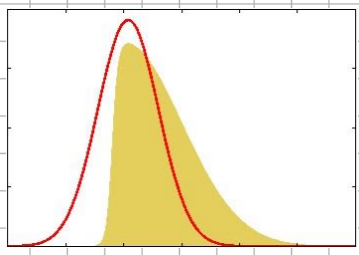
→ viel Rechenzeit nötig!

Variationsapproximation:
aus Familie geeignet parametrisierter Verteilungen Q jene diejenige Verteilung $q(\vec{x})$ gewählt welche ein Distanzmass (häufig KL-Divergenz) zur exakten Verteilung $p(\vec{x})$ minimiert
→ basiert auf num. Optimierung
→ weniger rechenaufwändig als Monte-Carlo-Appr., aber abhängig zu spezifizierender

Laplace-Approximation:

$p(\vec{z}) = \frac{f(\vec{z})}{\int f(\vec{z})d\vec{z}} = \frac{f(\vec{z})}{Z}$ ← berechenbar, ← nicht berechenbar } Annahmen

$\simeq q(\vec{z}) = \mathcal{N}(\vec{z}|\vec{z}_0,\underline{\Lambda}^{-1}) = \mathcal{N}(\vec{z}\,|\,\text{argmax}_{\vec{z}}\,p(\vec{z}),\,(-\underline{H\ln f(z_0)})^{-1})$



→ Normalverteilung mit gleichem Modus und Krümmung bei diesem

$Z \simeq \frac{f(\vec{z})}{q(\vec{z})} \xrightarrow{\ln()} \ln Z \simeq \ln f(\vec{z}) - \ln q(\vec{z}) = \ln f(\vec{z}) - \frac{1}{2}\ln\det\underline{\Lambda} + \frac{d}{2}\ln(2\pi)$

bei Posterior-V. $p(\vec{\theta}|D) = p(D|\vec{\theta})p(\vec{\theta})/p(D)$ ist vielfach $p(D|\vec{\theta})p(\vec{\theta})$ analytisch verfügbar aber Normalisierungskonstante $p(D)=\int p(D|\vec{\theta})p(\vec{\theta})d\vec{\theta}$ nicht berechenbar

Laplace-Posterior-Approximation (allgemein):

$q(\vec{\theta}) = \mathcal{N}(\vec{\theta}|\vec{\theta}_{MAP},\underline{\Lambda}^{-1})$ mit $\underline{\Lambda} = -\underline{H\ln p(D|\vec{\theta}_{MAP})} - \underline{H\ln p(\vec{\theta}_{MAP})}$

$\Rightarrow \ln p(D) \simeq \ln p(D|\vec{\theta}_{MAP}) + \ln p(\vec{\theta}_{MAP}) - \frac{1}{2}\ln\det\underline{\Lambda} + \frac{d}{2}\ln(2\pi)$ (Logarithmierte marginale Likelihood)

Occams Faktor → bestraft(en) flexible/komplexe Modelle

log. Regression mit Bayesscher Methode:

Prior: $p(\underline{w}) = \mathcal{N}(\underline{w}|\vec{m}_0,\underline{\Lambda}_0^{-1})$

Lapl.-Post.-Appr.: $p(\underline{w}|\vec{m}_0,\underline{\Lambda}_0,\underline{X},\vec{t})$

zuerst numerisch mit $\underline{X}$, $\vec{t}$ $\underline{w}_{MAP}$ bestimmen und damit anschliessend $\underline{\Lambda} = -\underline{H\ln p(\vec{t}|\underline{X},\underline{w}_{MAP})} + \underline{\Lambda}_0$

Post.-Voraussage: $p(t|\vec{x},\underline{X},\vec{t}) = \int p(t|\vec{x},\underline{w})\,p(\underline{w}|\vec{m}_0,\underline{\Lambda}_0,\underline{X},\vec{t})\,d\underline{w}$ nicht geschlossen berechenbar → müssen numerisch approximieren

$\simeq \frac{1}{|S|}\sum_{\underline{w}\in S} p(t|\vec{x},\underline{w})$ Monte-Carlo-Approximation

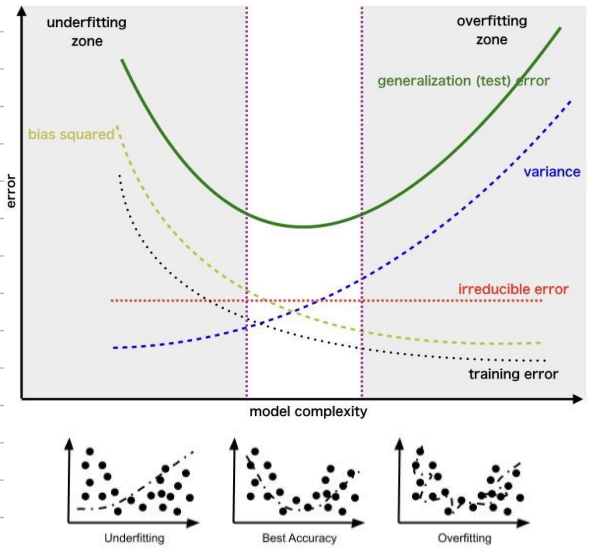
Erweiterungen und Schwächen der logistischen Regression

generelle lineare Modelle:
- Verallgemeinerung d. log. Regression auf andere Ausgabeverteilungen
- anstatt Softmax wird eine zur Verteilung passende Aktivierungsfunktion verwendet
- lineare Abhängigkeit in Gewichten und Features

Schwächen:
- Fluch der Dimension in Kombination mit fixer Wahl von nicht adaptiven Features
→ Probleme bei hochdimensionalen Eingaben

Komplexität und Modellselektion

erwarteter Verlust und Flexibilität von Modellen



der Einfachheit halber betrachten wir eindimensionale Regressionsprobleme und wählen quadratisches Verlustmass zur Bewertung der Verallgemeinerungsqualität

Verlustmass: $L(t_{True}, t_{Pred}) = (t_{True} - t_{Pred})^2$

exakte Verteilung: $p(t, \vec{x})$ → nicht bekannt

Voraussagefunktion: $f(\vec{x})$

optimale Voraussagef.: $m(\vec{x}) = E_{p(\cdot|\vec{x})}[t]$

erwarteter Verlust:

$$E[L_f] = \int (t - f(\vec{x}))^2 p(t,\vec{x})\,dt\,d\vec{x} = \int (t - m(\vec{x}) + m(\vec{x}) - f(\vec{x}))^2 p(t,\vec{x})\,dt\,d\vec{x}$$

$$= \int \left((t - m(\vec{x}))^2 + 2(t - m(\vec{x}))(m(\vec{x}) - f(\vec{x})) + (m(\vec{x}) - f(\vec{x}))^2\right) p(t,\vec{x})\,dt\,d\vec{x}$$

$$= \int (t - m(\vec{x}))^2 p(t,\vec{x})\,dt\,d\vec{x} + \int 2(t - m(\vec{x}))(m(\vec{x}) - f(\vec{x}))\,p(t,\vec{x})\,dt\,d\vec{x} + \int (m(\vec{x}) - f(\vec{x}))^2 p(t,\vec{x})\,dt\,d\vec{x}$$

$\int (t - m(\vec{x}))^2 p(t,\vec{x})\,dt\,d\vec{x} = E[L_m]$ unvermeidbarer erwarteter Verlust

$$\int 2(t - m(\vec{x}))(m(\vec{x}) - f(\vec{x}))\,p(t,\vec{x})\,dt\,d\vec{x} = \int \left(\int 2(t - m(\vec{x}))(m(\vec{x}) - f(\vec{x}))\,p(t|\vec{x})\,dt\right) p(\vec{x})\,d\vec{x} = \int \left(2(m(\vec{x}) - f(\vec{x})) \int (t - m(\vec{x}))\,p(t|\vec{x})\,dt\right) p(\vec{x})\,d\vec{x}$$

$$= \int \left(2(m(\vec{x}) - f(\vec{x}))\left(\int t\,p(t|\vec{x})\,dt - \int m(\vec{x})\,p(t|\vec{x})\,dt\right)\right) p(\vec{x})\,d\vec{x} = \int \left(2(m(\vec{x}) - f(\vec{x}))\left(\int t\,p(t|\vec{x})\,dt - m(\vec{x}) \int p(t|\vec{x})\,dt\right)\right) p(\vec{x})\,d\vec{x}$$

$$= \int \left(2(m(\vec{x}) - f(\vec{x}))\left(\int t\,p(t|\vec{x})\,dt - m(\vec{x}) \cdot 1\right)\right) p(\vec{x})\,d\vec{x} = \int \left(2(m(\vec{x}) - f(\vec{x}))(m(\vec{x}) - m(\vec{x}))\right) p(\vec{x})\,d\vec{x} = 0$$

$$\int (m(\vec{x}) - f(\vec{x}))^2 p(t,\vec{x})\,dt\,d\vec{x} = \int \left(\int (t - m(\vec{x}))^2 p(t|\vec{x})\,dt\right) p(\vec{x})\,d\vec{x} = \int \left((m(\vec{x}) - f(\vec{x}))^2 \int p(t|\vec{x})\,dt\right) p(\vec{x})\,d\vec{x}$$

$$= \int \left((m(\vec{x}) - f(\vec{x}))^2 \cdot 1\right) p(\vec{x})\,d\vec{x} = \int (m(\vec{x}) - f(\vec{x}))^2 p(\vec{x})\,d\vec{x} = \int (m(\vec{x}) - f(\vec{x}))^2 p(\vec{x})\,d\vec{x}$$

$$E[L_f] = \int (m(\vec{x}) - f(\vec{x}))^2 p(\vec{x})\,d\vec{x} + E[L_m]$$

unser Modell (und damit auch erw. Verlust) hängt von Daten ab:

$$E[L_{f_m}] = E_{\mathcal{D}}[E[L_{f_m(\cdot|D)}]] = \int E_{\mathcal{D}}\left[(m(\vec{x}) - f_m(\vec{x}|D))^2\right] p(\vec{x})\,d\vec{x} + E[L_m]$$

$$= \int \left((m(\vec{x}) - E_{\mathcal{D}}[f_m(\vec{x}|D)])^2 + Var_{\mathcal{D}}[f_m(\vec{x}|D)]\right) p(\vec{x})\,d\vec{x} + E[L_m]$$

$$= \int (m(\vec{x}) - E_{\mathcal{D}}[f_m(\vec{x}|D)])^2 p(\vec{x})\,d\vec{x} + \int Var_{\mathcal{D}}[f_m(\vec{x}|D)]\,p(\vec{x})\,d\vec{x} + E[L_m]$$

$$= \text{bias}^2 + \text{variance} + \text{noise}$$

wobei $\mathcal{D} = \{D \mid \text{mit } |D| = N\}$ alle möglichen Trainingsdaten vom Umfang N bezeichnet

Modelle selektieren mit der Bayes-Methode

Method	Definition
Maximum likelihood	$\hat{\theta} = \operatorname{argmax}_\theta p(\mathcal{D}\|\theta)$
MAP estimation	$\hat{\theta} = \operatorname{argmax}_\theta p(\mathcal{D}\|\theta)p(\theta\|\eta)$
ML-II (Empirical Bayes)	$\hat{\eta} = \operatorname{argmax}_\eta \int p(\mathcal{D}\|\theta)p(\theta\|\eta)d\theta = \operatorname{argmax}_\eta p(\mathcal{D}\|\eta)$
MAP-II	$\hat{\eta} = \operatorname{argmax}_\eta \int p(\mathcal{D}\|\theta)p(\theta\|\eta)p(\eta)d\theta = \operatorname{argmax}_\eta p(\mathcal{D}\|\eta)p(\eta)$
Full Bayes	$p(\theta, \eta\|\mathcal{D}) \propto p(\mathcal{D}\|\theta)p(\theta\|\eta)p(\eta)$

Posterior auf Raum der Modelle:

$$p(M|D) = \frac{p(D|M)\,p(M)}{p(D)}$$

wichtigste Formel dieses Abschnitts

Evidenz / marginale Likelihood:

$$p(D|M) = \int p(D|\vec{\theta}, M)\,p(\vec{\theta}|M)\,d\vec{\theta}$$

für viele Modelle nicht geschlossen berechenbar
für lineare Modelle berechenbar ✓

Posterior über Parameter:

$$p(\vec{\theta}|D, M) = \frac{p(D|\vec{\theta}, M)\,p(\vec{\theta}|M)}{p(D|M)}$$

falls man nur 1 Modell betrachtet, kann man Bedingung M weglassen

Normalisierungsfaktor in $p(M|D)$:

$$p(D) = \int p(D|M)\,p(M)\,dM$$

Hyperprior $p(M)$ hat auch wieder Parameter → mit Bayes führt das konsequenterweise auf hierarchische bayessche Modelle (oder zu graphischen probabilistischen Modellen)
→ prinzipiell standard aber selten berechenbar
→ Annäherungen

Bayessche Posterior-Voraussage-Verteilung (bayessche Modell-Mittelung):

$$p(\vec{t}|\vec{x}, D) = \int p(\vec{t}|\vec{x}, M, D)\,p(M|D)\,dM$$

Mittelung aller Modelle

wobei $p(\vec{t}|\vec{x}, M, D) = \int p(\vec{t}|\vec{x}, \vec{\theta}, M)\,p(\vec{\theta}|D, M)\,d\vec{\theta}$

Vereinfachung

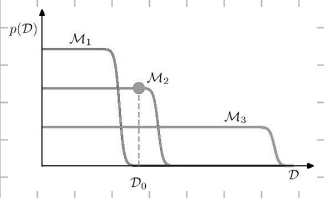
Bayessche Modell-Selektion / Type II MAP:

$$p(\vec{t}|\vec{x}, \hat{M}, D) \quad \text{wobei } \hat{M} = \operatorname{argmax}_M p(M|D) = \operatorname{argmax}_M p(D|M)\,p(M)$$

empirische Bayessche Modell-Selektion / Type II ML:

Vereinfachung

$$p(\vec{t}|\vec{x}, \hat{M}, D) \quad \text{wobei } \hat{M} = \operatorname{argmax}_M p(D|M)$$



Selektion eines Modells mittels empirischer bayesscher Modellselektion, also basierend auf der höchsten marginalen Likelihood $p(D|M)$ der Trainingsdaten (bei welcher über Parameter $\vec{\theta}$ mit Hilfe einer Prior-Verteilung integriert und nicht maximiert wird) neigt viel weniger zu Überanpassung als Selektion des Modells mit maximaler Likelihood

Bayessche Hyperparameterselektion für lineare Regressionsprobleme

Modell:

$$p(t|\vec{x},\vec{w},\beta) = \mathcal{N}(t|\Phi(\vec{x})^T\vec{w},\beta^{-1})$$

ganzscher Prior:

$$p(\vec{w}|\alpha) = \mathcal{N}(\vec{w}|\vec{0},\alpha^{-1}\underline{E})$$

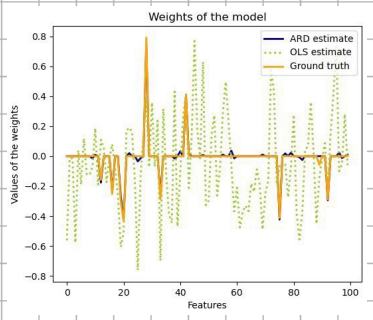
(logarithmierte) marginale Likelihood:

$$\ln p(\vec{t}|\underline{X},\alpha,\beta) = \frac{-1}{2}(N\ln(2\pi) + \ln\det(\underline{C}) + \vec{t}^T\underline{C}^{-1}\vec{t}) \quad \text{wobei} \quad \underline{C} = \beta^{-1}\underline{E} + \alpha^{-1}\underline{\Phi}(\underline{X})\,\underline{\Phi}(\underline{X})^T$$

man kann nun logarithmierte marginale Likelihood (Typ II ML) mit Optimierung maximieren

man kann auch Prior über α, β wählen und Typ II MAP oder sogar volle bayessche Modellmittelung mittels Monte-Carlo-Methoden oder Variationsmethoden approximieren

Automatische-Relevanz-Bestimmung (ARD)



kann man als Feature Selektion für lineare Regressionsmodelle auffassen

Modell:

$$p(t|\vec{x},\vec{w},\beta) = \mathcal{N}(t|\Phi(\vec{x})^T\vec{w},\beta^{-1})$$

ganzscher Prior:

$$p(\vec{w}|\vec{\alpha}) = \mathcal{N}(\vec{w}|\vec{0},\underline{D}_{\vec{\alpha}^{-1}}) = \prod_{k=0}^{M}\mathcal{N}(w_k|0,\alpha_k^{-1})$$

(logarithmierte) marginale Likelihood:

$$\ln p(\vec{t}|\underline{X},\vec{\alpha},\beta) = \frac{-1}{2}(N\ln(2\pi) + \ln\det(\underline{C}) + \vec{t}^T\underline{C}^{-1}\vec{t}) \quad \text{wobei} \quad \underline{C} = \beta^{-1}\underline{E} + \underline{\Phi}(\underline{X})\,\underline{D}_{\vec{\alpha}^{-1}}\,\underline{\Phi}(\underline{X})^T$$

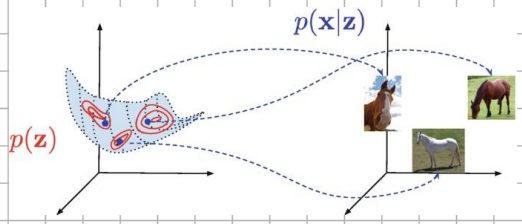
nun maximiert man logarithmierte marginale Likelihood (Typ II ML) mit Optimierung

dabei streben viele der Präzisions-Hyper-Parameter $\alpha_k \to \infty$ und folglich werden die dazugehörigen Parameter $w_k \to 0$ gezwungen

bei ARD wird vielfach auch noch die Präzision β der Ausgabeverteilung optimiert und mittels Gamma-Prior-Verteilung über $\vec{\alpha}$ und β kann regularisiert werden (Typ II MAP), was erlaubt die Stärke der Selektion zu kontrollieren

mit ARD können auch bei (log)stischen Regressionen Features selektiert werden, wobei die marginale Likelihood jedoch nicht exakt berechenbar ist und somit approximiert werden muss

Modelle mit verborgenen Variablen



Definition:

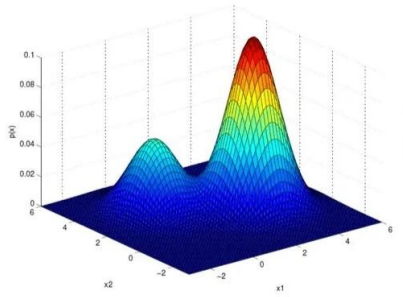
$p(\vec{x}, \vec{z} \mid \vec{\theta})$ $\vec{x}$: beobachtete Variablen, $\vec{z}$: verborgene Variablen

induzierte Verteilung über beobachteten Variablen:

$$p(\vec{x} \mid \vec{\theta}) = \int p(\vec{x}, \vec{z} \mid \vec{\theta})\, d\vec{z}$$

manchmal aber nicht immer haben verborgene Variablen eine Bedeutung

Mischungsmodelle



$$p(\vec{x}, z \mid \vec{\theta}, \vec{\pi}) = p(\vec{x} \mid z, \vec{\theta})\, p(z \mid \vec{\pi})$$

mit $p(z=k \mid \vec{\pi}) = \pi_k$ und $p(\vec{x} \mid z=k, \vec{\theta}) = p_k(\vec{x} \mid \vec{\theta}_k)$

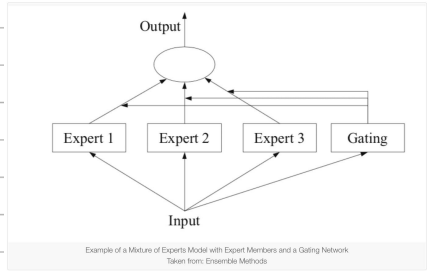
marginale Verteilung:

$$p(\vec{x} \mid \vec{\theta}, \vec{\pi}) = \sum_{k=1}^{K} p(\vec{x}, z=k \mid \vec{\theta}, \vec{\pi}) = \sum_{k=1}^{K} \pi_k\, p_k(\vec{x} \mid \vec{\theta}_k)$$

Gausssche Mischungsmodelle:

$$p(\vec{x} \mid \vec{\theta}, \vec{\pi}) = \sum_{k=1}^{K} \pi_k\, \mathcal{N}(\vec{x} \mid \vec{\mu}_k, \Sigma_k)$$

Mischung von Experten:



$$p(t, z \mid x, \vec{\theta}, \vec{\pi}) = p(t \mid x, z=k, \vec{\theta})\, p(z \mid x, \vec{\pi} = W)$$
$$= \mathcal{N}(t \mid \vec{\phi}(x)^T \vec{w}_k, \beta_k^{-1})\, \mathrm{Cat}_{\vec{z}}(W \vec{\phi}(x))(z)$$

- z hängt (über logistisches Regressionsmodell) von x ab
- lineare Regressionsmodelle für die einzelnen z bedingten Komponenten

→ einzelnen Komponenten können sich auf Regionen der Eingaben spezialisieren («Experten»)

Inferenz in Mischungsmodellen:

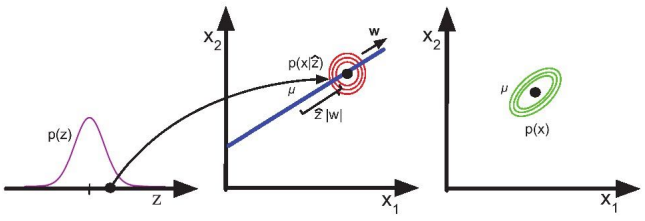
$$p(z=j \mid \vec{x}, \vec{\theta}, \vec{\pi}) = \frac{p(\vec{x} \mid z=j, \vec{\theta})\, p(z=j \mid \vec{\pi})}{p(\vec{x} \mid \vec{\theta}, \vec{\pi})} = \frac{\pi_j \cdot p_j(\vec{x} \mid \vec{\theta}_j)}{\sum_{k=1}^{K} \pi_k \cdot p_k(\vec{x} \mid \vec{\theta}_k)}$$

«responsibility»

da Bezeichnungen $\{1, \dots, K\}$ der Mischungskomponenten (genauer die W'keit π_k zusammen mit Parameter $\vec{\theta}_k$) permutiert werden können ohne Verteilung $p(\vec{x} \mid \vec{\theta}, \vec{\pi})$ über beobachtbaren Variablen zu verändern, sind Parameter $\vec{\pi}, \vec{\theta}$ höchstens modulo dieser Permutationssymmetrie bestimmbar

→ indem zusätzliche Bedingungen formuliert werden, kann diese Symmetrie bei Bedarf gebrochen werden

Faktoranalysemodelle



können zur (linearen) Dimensionsreduktion verwendet werden

zur Hauptkomponentenanalyse analoges, generatives Modell mit kontinuierlichem $\vec{z}$

$p(\vec{x}, \vec{z}) = p(\vec{x}|\vec{z})p(\vec{z})$ mit $\vec{x} \in \mathbb{R}^d$, $\vec{z} \in \mathbb{R}^L$ wobei typischerweise $L \ll d$ ist

$p(\vec{z}) = \mathcal{N}(\vec{z}|\vec{0}, \underline{E})$ Standardnormalverteilte verborgene Variable

$p(\vec{x}|\vec{z}) = \mathcal{N}(\vec{x}|\underline{W}\vec{z} + \vec{\mu}, \text{diag}(\vec{c}))$ $\underline{W}$ = Faktormatrix, $\underline{W} \in \mathbb{R}^{d \times L}$

da bedingte Verteilung $p(\vec{x}|\vec{z})$ unkorreliert ist, muss die affine Transformation $\underline{W}\vec{z} + \vec{\mu}$ der niederdimensionalen, auch unkorrelierten, verborgenen Variable $\vec{z}$ die Korrelationen in der beobachteten Variablen $\vec{x}$ erklären. in diesem Sinne ist verborgene Variable eine komprimierte Darstellung der beobachteten Variable in verschiedene unkorrelierte Faktoren

diese Auffassung wird auch durch die induzierte Verteilung

$p(\vec{x}) = \mathcal{N}(\vec{x}|\vec{\mu}, \text{diag}(\vec{c}) + \underline{W}\underline{W}^T)$

über die beobachtete Variable bestätigt, weil $\underline{W}$ die Korrelation daran beteiligt

die Flexibilität der Korrelation wird durch die Dimension L des Raums der verborgenen Variablen (bottleneck) festgelegt und liegt zwischen unkorreliert (diagonale Kovarianz) und voller Kopplung (allgemeine symmetrische Kovarianz)

indem wir $\text{diag}(\vec{c}) = \sigma \underline{E}$ wählen erhalten wir im Wesentlichen die generative Version der Hauptkomponentenanalyse (probabilistische PCA, PPCA)

ein Vorteil des Faktoranalysemodells gegenüber PPCA ist, dass die nicht durch die affine Transformation $\underline{W}\vec{z} + \vec{\mu}$ erklärbaren Anteile von $\vec{x}$ nicht in allen Komponenten dieselbe Varianz haben müssen

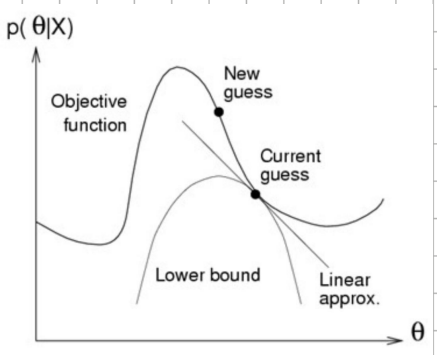
Inferenzproblem für das Faktoranalysemodell:

$\vec{m}_{\vec{z}} = E[\vec{z}|\vec{x}]$ Latenz-Faktor der Beobachtung $\vec{x}$

$p(\vec{z}|\vec{x}) = \mathcal{N}(\vec{z}|\vec{m}_{\vec{z}}, \underline{\Sigma})$ mit $\underline{\Sigma} = (\underline{E} + \underline{W}^T \text{diag}(\vec{c})^{-1} \underline{W})^{-1}$ und $\vec{m}_{\vec{z}} = \underline{\Sigma}\underline{W}^T \text{diag}(\vec{c})^{-1}(\vec{x} - \vec{\mu})$

Faktormatrix $\underline{W}$ nicht identifizierbar da mit einer beliebigen orthogonalen Matrix $\underline{R} \in \mathbb{R}^{L \times L}$ von rechts multipliziert eine andere Matrix $\tilde{\underline{W}} = \underline{W}\underline{R}$ dieselbe Verteilung $p(\vec{x})$ resultiert
→ dies muss man beachten, wenn man z.B. die Spalten von $\underline{W}$ interpretieren möchte
→ um den Grad der Nicht-Identifizierbarkeit zu verringern, kann man zusätzliche Bedingungen an $\underline{W}$ formulieren, z.B. orthogonale Spaltenvektoren fordern (→ Singulärwertzerlegung!)

Lernproblem für Modelle m. v. Variablen: EM-Algorithmus



«Anleitung um Algorithmus herzuleiten, mit dem man eine Annäherung an die ML- oder MAP-Parameter erhält wenn verborgene Variablen vorhanden sind»

um die Parameter $\vec{\theta}$ eines Modells $p(\vec{x}, \vec{z} \mid \vec{\theta})$ mit verborgenen Variablen $\vec{z}$ an beobachtbare Daten D anzupassen, könnte man versuchen direkt ML auf induzierte Verteilung $p(\vec{x} \mid \vec{\theta}) = \int p(\vec{x}, \vec{z} \mid \vec{\theta}) d\vec{z}$ anzuwenden (maximieren in $\vec{\theta}$)

Problem: analytisch selten möglich und numerisch schnell schwierig (Funktion komplizierter und nicht konkav, benötigte Marginalisierung über verborgene Variable ziemlich rechenintensiv)

(eine robuste und häufig verwendete) Lösung: EM-Algorithmus (iterativ)

konvergiert garantiert gegen lokales Maximum der Likelihood $\ln(D \mid \vec{\theta}) = \sum_{k=1}^{N_D} \ln p(\vec{x}_k \mid \vec{\theta}) = \sum_{k=1}^{N_D} \ln\left(\int p(\vec{x}_k, \vec{z} \mid \vec{\theta}) d\vec{z}\right)$

2 Phasen:

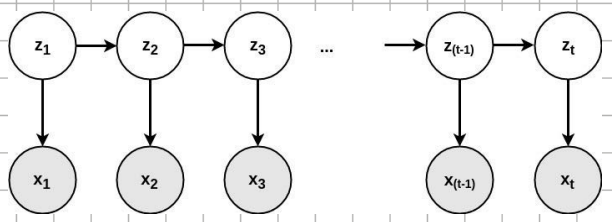
Expectation-Phase:

1. bedingte Verteilung $p(\vec{z} \mid D, \vec{\theta}_t) = \prod_{j=1}^{N_D} p(\vec{z}_j \mid \vec{x}_j, \vec{\theta}_t)$ der unbeobachteten Variablen $Z = \{\vec{z}_k \mid 1 \le k \le N_D\}$ gegeben die Daten $D = \{\vec{x}_k \mid 1 \le k \le N_D\}$ unter aktuellen Parametern $\vec{\theta}_t$ (analytisch oder numerisch) bestimmen
2. erwartete totale logarithmierte Likelihood $Q(\vec{\theta} \mid D, \vec{\theta}_t) = E_{p(\vec{z} \mid D, \vec{\theta}_t)}[\ln p(D, Z \mid \vec{\theta})] = \sum_{i=1}^{N_D} E_{p(\vec{z}_i \mid \vec{x}_i, \vec{\theta}_t)}[\ln(\vec{x}_i, \vec{z}_i \mid \vec{\theta})]$ der Daten D in Abhängigkeit von $\vec{\theta}$ berechnen

Maximization-Phase:

$\vec{\theta}_{t+1} = \operatorname{argmax}_{\vec{\theta}} Q(\vec{\theta} \mid D, \vec{\theta}_t)$

Modelle für sequentielle Daten



einzelne Beobachtungen $\vec{x}$ nicht unabhängig!
→ hängt vom vorangegangenen Zustand ab

Annahmen:

- $p(\vec{x}_{t+1} | \vec{x}_1, \dots, \vec{x}_t) = p(\vec{x}_{t+1} | \vec{x}_t) = p_{trans}(\vec{x}_{t+1} | \vec{x}_t)$
- $p(\vec{x}_1) = p_{init}(\vec{x}_1)$

Annahme, dass nur vorangegangener Zustand Einfluss auf nächsten hat

Markov-Modell:

$$p(\vec{x}_1, \dots, \vec{x}_T) = p_{init}(\vec{x}_1) \prod_{t=2}^{T} p_{trans}(\vec{x}_t | \vec{x}_{t-1})$$

→ oft zu restriktiv, da nur unmittelbarer Vorgänger → MM höherer Ordnung

Markov-Modell höherer Ordnung $j > 1$:

$$p(\vec{x}_1, \dots, \vec{x}_T) = p_{init}(\vec{x}_1, \dots, \vec{x}_j) \prod_{t=2}^{T} p_{trans}(\vec{x}_t | \vec{x}_{t-1}, \dots, \vec{x}_{t-j})$$

Zustandsraummodelle

Annahmen:

- Transitionsmodell: $p(\vec{z}_{t+1} | \vec{z}_t) = p_{trans}(\vec{z}_{t+1} | \vec{z}_t)$ für alle Zeiten t — stationäres MM 1ter Ordnung
- Beobachtungsmodell: $p(\vec{x}_t | \vec{z}_t) = p_{obs}(\vec{x}_t | \vec{z}_t)$

gemeinsame Verteilung einer Sequenz von beob. und verb. Variablen:

$$p(\vec{x}_{1:T}, \vec{z}_{1:T}) = p_{init}(\vec{z}_1) \prod_{t=2}^{T} p_{trans}(\vec{z}_t | \vec{z}_{t-1}) \prod_{t=1}^{T} p_{obs}(\vec{x}_t | \vec{z}_t)$$

→ mittels Marginalisierung über alle verborgenen Variablen $\vec{z}_{1:T}$ erhält man die induzierte Verteilung $p(\vec{x}_{1:T})$ über die beobachteten Werte

i.A. werden für p_{trans} & p_{obs} parametrisierte Vert. verwendet → $\vec{\theta} = (\vec{\theta}_{init}, \vec{\theta}_{trans}, \vec{\theta}_{obs})$

in einigen Anwendungen hängen diese Verteilungen auch noch von (immer beobachteten) Eingabevariablen $\vec{u}_t$ ab:

$$p(\vec{x}_{1:T}, \vec{z}_{1:T} | \vec{u}_{1:T}, \vec{\theta}) = p_{init}(\vec{z}_1 | \vec{u}_1, \vec{\theta}_{init}) \prod_{t=2}^{T} p_{trans}(\vec{z}_t | \vec{z}_{t-1}, \vec{u}_t, \vec{\theta}_{trans}) \prod_{t=1}^{T} p_{obs}(\vec{x}_t | \vec{z}_t, \vec{u}_t, \vec{\theta}_{obs})$$

Inferenzproblem: $p(\vec{z}_{1:T} | \vec{x}_{1:T}, \vec{\theta})$, $p(\vec{z}_t | \vec{x}_{1:T}, \vec{\theta})$ oder auch $p(\vec{x}_{1:T} | \vec{\theta})$

es folgen 2 Spezialisierungen von Zustandsraummodellen

- HMMs → $z_t \in \{1, \dots, K\}$ diskret, $\vec{x}_t$ diskret oder kontinuierlich
- LDSs → $\vec{z}_t \in \mathbb{R}^d$ kontinuierlich, p_{trans} & p_{obs} linear, alles Gauss-Verteilungen

Hidden Markov Modelle (HMMs)

$p_{init}(z_1 = k) = \pi_k \quad \rightarrow \quad \vec{\pi} \in \mathbb{R}^K$

$p_{trans}(z_t = k \mid z_{t-1} = i) = A_{i,k} \quad \rightarrow \quad \underline{A} \in \mathbb{R}^{K \times K}$ Transitionsmatrix

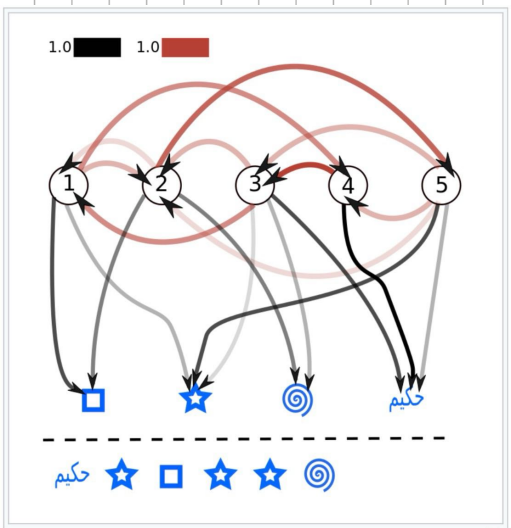
Bemerkungen:
- $\vec{\pi} \in \mathbb{R}^K$ mit $K-1$ Freiheitsgraden da $\sum_k \pi_k = 1$
- $\underline{A} \in \mathbb{R}^{K \times K}$ mit $K \cdot (K-1)$ Freiheitsgraden da $\sum_k A_{i,k} = 1$ für alle i

für Beobachtungsmodell müssen die K bedingten V. $p_{obs}(\vec{x}_t \mid z_t = k)$ definiert werden

falls Beobachtungen $x_t \in \{1, \dots, L\}$ auch diskret sind, liefert allgemeinste Parametrisierung ohne Wahrscheinlichkeitsmatrix $\underline{O} \in \mathbb{R}^{K \times L}$ mit $K \cdot (L-1)$ Freiheitsgraden mit Einträgen $p_{obs}(x_t = l \mid z_t = i) = O_{i,l}$

bei kontinuierlichen Beobachtungen $\vec{x}_t \in \mathbb{R}^d$ werden häufig gaussche Beobachtungsmodell $p_{obs}(\vec{x}_t \mid z_t = k) = \mathcal{N}(\vec{x}_t \mid \vec{\mu}_k, \underline{\Sigma}_k)$ angesetzt
→ können als gaussche Mischungsmodelle interpretiert werden, in welchen die Mischungskomponenten zeitlich korreliert sind

natürlich können auch anders parametrisierte Beobachtungsmodelle eingesetzt werden wie zB Neuronale Netze



The state transition and output probabilities of an HMM are indicated by the line opacity in the upper part of the diagram. Given that we have observed the output sequence in the lower part of the diagram, we may be interested in the most likely sequence of states that could have produced it. Based on the arrows that are present in the diagram, the following state sequences are candidates:
5 3 2 5 3 2
4 3 2 5 3 2
3 1 2 5 3 2
We can find the most likely sequence by evaluating the joint probability of both the state sequence and the observations for each case (simply by multiplying the probability values, which here correspond to the opacities of the arrows involved). In general, this type of problem (i.e. finding the most likely explanation for an observation sequence) can be solved efficiently using the Viterbi algorithm.

Algorithmen dieses Abschnitts sind Spezialfälle von Message-Passing-Algorithmen (dynamische Programmierung) für das Inferenzproblem bei Probabilistischen Graphischen Modellen

$$p(\vec{x}_{1:T}) = \sum_{z_{1:T} \in \{1,\dots,K\}^T} p(\vec{x}_{1:T}, z_{1:T}) = \sum_{z_{1:T} \in \{1,\dots,K\}^T} p_{init}(z_1) \prod_{t=2}^{T} p_{trans}(z_t \mid z_{t-1}) \prod_{t=1}^{T} p_{obs}(\vec{x}_t \mid z_t)$$

dieser Ausdruck besteht naiv implementiert aus K^T Summanden und lässt sich geschickter organisieren

Filterung

Def. Filterung. rekursive Berechnung von $p(z_t \mid \vec{x}_{1:t})$ ausgehend von $p(z_{t-1} \mid \vec{x}_{1:t-1})$ resp $\vec{\alpha}_t$ aus $\vec{\alpha}_{t-1}$

Th. Filterung. für ein HMM mit der Transitionsmatrix $\underline{A}$ und der Initialverteilung $\vec{\pi}$ kann man $\vec{\alpha}_t$ wie folgt rekursiv berechnen:

(1) $\vec{\alpha}_1 = \frac{\vec{\alpha}_1}{L_1}$ mit $\vec{\alpha}_1 = \vec{\lambda}_1 \odot \vec{\pi}$ und $L_1 = p(\vec{x}_1) = \|\vec{\alpha}_1\|_1$ für $t = 1$

(2) $\vec{\alpha}_t = \frac{\vec{\alpha}_t}{L_t}$ mit $\vec{\alpha}_t = \vec{\lambda}_t \odot \underline{A}^T \vec{\alpha}_{t-1}$ und $L_t = p(\vec{x}_t \mid \vec{x}_{1:t-1}) = \|\vec{\alpha}_t\|_1$ für $t \geq 2$

wobei $\vec{\lambda}_t$ die lokale Likelihood mit Komponenten $\lambda_{t,k} = p_{obs}(\vec{x}_t \mid z_t = k)$ bezeichnet

Bew. Filterung.

(1) $$p(z_1 = k \mid \vec{x}_1) \overset{\text{Bayes}}{=} \frac{p(\vec{x}_1 \mid z_1 = k)\, p(z_1 = k)}{p(\vec{x}_1)} \overset{\text{tot. Wkeit}}{=} \frac{p(\vec{x}_1 \mid z_1 = k)\, p(z_1 = k)}{\sum_{k'=1}^{K} p(\vec{x}_1 \mid z_1 = k')\, p(z_1 = k')} = \frac{(\vec{\lambda}_1 \odot \vec{\pi})_k}{\|\vec{\lambda}_1 \odot \vec{\pi}\|_1} = \frac{\alpha_{1,k}}{\|\vec{\alpha}_1\|_1}$$

(2) mit $p(\vec{x}_t \mid z_t = k, \vec{x}_{1:t-1}) = p_{obs}(\vec{x}_t \mid z_t = k)$, $p(\vec{x}_t \mid \vec{x}_{1:t-1}) = \sum_{k'=1}^{K} p_{obs}(\vec{x}_t \mid z_t = k') p(z_t = k' \mid \vec{x}_{1:t-1})$, $p(z_t = k \mid \vec{x}_{1:t-1}) = \sum_{k''=1}^{K} p(z_t = k, z_{t-1} = k'' \mid \vec{x}_{1:t-1})$
$= \sum_{k''=1}^{K} p(z_t = k \mid z_{t-1} = k'', \vec{x}_{1:t-1})\, p(z_{t-1} = k'' \mid \vec{x}_{1:t-1}) \overset{\text{Markov}}{=} \sum_{k''=1}^{K} p_{trans}(z_t = k \mid z_{t-1} = k'')\, p(z_{t-1} = k'' \mid \vec{x}_{1:t-1}) = \sum_{k''=1}^{K} A_{k'',k}\, \alpha_{t-1,k''}$ folgt:

$$p(z_t = k \mid \vec{x}_{1:t}) = p(z_t = k \mid \vec{x}_t, \vec{x}_{1:t-1}) \overset{\text{Bayes}}{=} \frac{p(\vec{x}_t \mid z_t = k, \vec{x}_{1:t-1})\, p(z_t = k \mid \vec{x}_{1:t-1})}{p(\vec{x}_t \mid \vec{x}_{1:t-1})} = \frac{p_{obs}(\vec{x}_t \mid z_t = k) \sum_{k''=1}^{K} A_{k'',k}\, \alpha_{t-1,k''}}{\sum_{k'=1}^{K} p_{obs}(\vec{x}_t \mid z_t = k')\, p(z_t = k' \mid \vec{x}_{1:t-1})}$$
$$= \frac{p_{obs}(\vec{x}_t \mid z_t = k) \sum_{k''=1}^{K} A_{k'',k}\, \alpha_{t-1,k''}}{\sum_{k'=1}^{K} p_{obs}(\vec{x}_t \mid z_t = k') \sum_{k''=1}^{K} A_{k'',k}\, \alpha_{t-1,k''}} = \frac{(\vec{\lambda}_t \odot \underline{A}^T \vec{\alpha}_{t-1})_k}{\|\vec{\lambda}_t \odot \underline{A}^T \vec{\alpha}_{t-1}\|_1} = \frac{\alpha_{t,k}}{\|\vec{\alpha}_t\|_1}$$

□

aus gefilterten Verteilung kann beispielsweise Vorhersageverteilung $p(\vec{x}_{t+k} \mid \vec{x}_{1:t})$ oder auch die marginale Verteilung $p(\vec{x}_{1:T})$ berechnen

Glättung

Def. Glättung. Berechnung von $p(z_t | \vec{x}_{1:T})$ resp $\vec{\gamma}_t$ gegeben alle verfügbaren Beobachtungen $\vec{x}_{1:T}$

Th. Glättung. für ein HMM mit der Transitionsmatrix $\underline{A}$ und der Initialverteilung $\vec{\pi}$ kann man $\vec{\gamma}_t$ wie folgt rekursiv berechnen:

(0) $\vec{\alpha}_t$ gemäss Filterung vorwärts in der Zeit für $t = 1, \dots, T$ berechnen

(1) $\vec{\beta}_t = \underline{A}(\vec{\lambda}_{t+1} \odot \vec{\beta}_{t+1})$ mit Startwert $\vec{\beta}_T = \vec{1}_k$ (→ sicheres Ereignis) rückwärts in der Zeit für $t = T, \dots, 1$ berechnen

(2) $\vec{\gamma}_t = \frac{\vec{\tilde{\gamma}}_t}{n_t}$ mit $\vec{\tilde{\gamma}}_t = \vec{\alpha}_t \odot \vec{\beta}_t$ und $n_t = \|\vec{\tilde{\gamma}}_t\|_1$ für alle t berechnen

wobei $\vec{\lambda}_t$ die lokale Likelihood mit Komponenten $\lambda_{t,k} = p_{obs}(\vec{x}_t | z_t = k)$ bezeichnet

Bew. Glättung.

$$p(z_t = k | \vec{x}_{1:T}) = p(z_t = k | \vec{x}_{t+1:T}, \vec{x}_{1:t}) \overset{\text{Bayes}}{=} \frac{p(\vec{x}_{t+1:T} | z_t = k, \vec{x}_{1:t}) p(z_t = k | \vec{x}_{1:t})}{p(\vec{x}_{t+1:T} | \vec{x}_{1:t})} = \frac{p(\vec{x}_{t+1:T} | z_t = k) p(z_t = k | \vec{x}_{1:t})}{\sum_{k'=1}^{K} p(\vec{x}_{t+1:T} | z_t = k') p(z_t = k' | \vec{x}_{1:t})}$$

$p(z_t = k | \vec{x}_{1:t}) =: \alpha_{t,k}$ (Filterung)

$p(\vec{x}_{t+1:T} | z_t = k) =: \beta_{t,k}$

$$\beta_{t,k} = p(\vec{x}_{t+1:T} | z_t = k) = p(\vec{x}_{t+2:T}, \vec{x}_{t+1} | z_t = k) \overset{\text{tot. Wkeit}}{=} \sum_{k''=1}^{K} p(\vec{x}_{t+2:T}, \vec{x}_{t+1}, z_{t+1} = k'' | z_t = k) \overset{\text{Mult.-R.}}{=} \sum_{k''=1}^{K} p(\vec{x}_{t+2:T} | \vec{x}_{t+1}, z_{t+1} = k'', z_t = k) p(\vec{x}_{t+1} | z_{t+1} = k'', z_t = k) p(z_{t+1} = k'' | z_t = k)$$
$$= \sum_{k''=1}^{K} p(\vec{x}_{t+2:T} | z_{t+1} = k'') p_{obs}(\vec{x}_{t+1} | z_{t+1} = k'') p_{trans}(z_{t+1} = k'' | z_t = k) = \sum_{k''=1}^{K} \beta_{t+1,k''} \lambda_{t+1,k''} A_{k,k''} = \sum_{k''=1}^{K} A_{k,k''} \lambda_{t+1,k''} \beta_{t+1,k''} = (\underline{A}(\vec{\lambda}_{t+1} \odot \vec{\beta}_{t+1}))_k$$

$$\Rightarrow p(z_t = k | \vec{x}_{1:T}) = \frac{p(\vec{x}_{t+1:T} | z_t = k) p(z_t = k | \vec{x}_{1:t})}{\sum_{k'=1}^{K} p(\vec{x}_{t+1:T} | z_t = k') p(z_t = k' | \vec{x}_{1:t})} = \frac{\beta_{t,k} \alpha_{t,k}}{\sum_{k'=1}^{K} \beta_{t,k'} \alpha_{t,k'}} = \frac{(\vec{\alpha}_t \odot \vec{\beta}_t)_k}{\|\vec{\alpha}_t \odot \vec{\beta}_t\|_1} = \frac{\vec{\tilde{\gamma}}_{t,k}}{\|\vec{\tilde{\gamma}}_t\|_1}$$ □

Glättung wird insbesondere für das Lernen benötigt

sowohl die Kosten Filter-Rekursion als auch die der Rückwärtsrekursion hängen linear von T und quadratisch von K ab und somit haben auch die Kosten der Glättung, also die Berechnung aller Verteilungen $p(z_t | \vec{x}_{1:T})$ für $1 \leq t \leq T$, insgesamt diese Abhängigkeit

mithilfe dieser beiden Rekursionen können beispielsweise die geglättete Nachfolgerverteilung $p(z_t, z_{t+1} | \vec{x}_{1:T})$ (resp. $\underline{\xi}_{t,t+1} \in \mathbb{R}^{K \times K}$) oder auch die wahrscheinlichste Abfolge $\hat{z}_{1:T}$ gegeben $\vec{x}_{1:T}$ (Viterbi-Algorithmus, linearer Aufwand in T) berechnet werden

Möglichkeiten für das Lernen der Parameter $\vec{\theta} = (\underline{A}, \vec{\pi}, \vec{\theta}_{obs})$ gegeben die Sequenzen $D = \{\vec{x}_{j,1:T_j} \mid 1 \leq j \leq N_D\}$:
- EM-Algorithmus, welcher in diesem Kontext auch Baum-Welch-Algorithmus genannt wird
- SGD (v.a. bei grossen Datenmengen)
- Bayesche Ansätze, welche auf Variationsapproximationen (Variational Bayes-EM) oder Monte-Carlo-Methoden basieren (da nicht exakt berechenbar)

Lineare dynamische Systeme (LDSs)

Transitionsmodell:

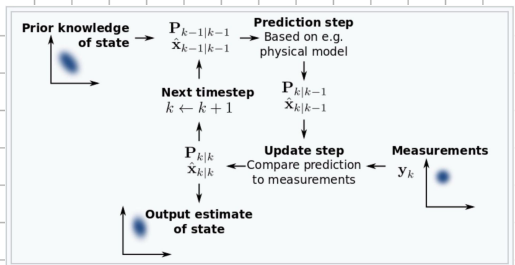
$$p_{trans}(\vec{z}_{t+1} | \vec{z}_t, \vec{u}_t) = \mathcal{N}(\vec{z}_{t+1} | \underline{E}\vec{z}_t + \underline{B}\vec{u}_t, \underline{Q}) \quad \text{mit } \underline{E} \in \mathbb{R}^{d \times d}, \underline{B} \in \mathbb{R}^{d \times c}, \underline{Q} \in \mathbb{R}^{d \times d}_{pos}$$

resp $\vec{z}_t = \underline{E}\vec{z}_{t-1} + \underline{B}\vec{u}_{t-1} + \vec{q}_t$ mit Prozessrauschen $\vec{q}_t \sim \mathcal{N}(\vec{0}, \underline{Q})$

Beobachtungsmodell:

$$p_{obs}(\vec{x}_t | \vec{z}_t) = \mathcal{N}(\vec{x}_t | \underline{H}\vec{z}_t + \underline{D}\vec{u}_t, \underline{R}) \quad \text{mit } \underline{H} \in \mathbb{R}^{n \times d}, \underline{D} \in \mathbb{R}^{n \times c}, \underline{R} \in \mathbb{R}^{n \times n}_{pos}$$

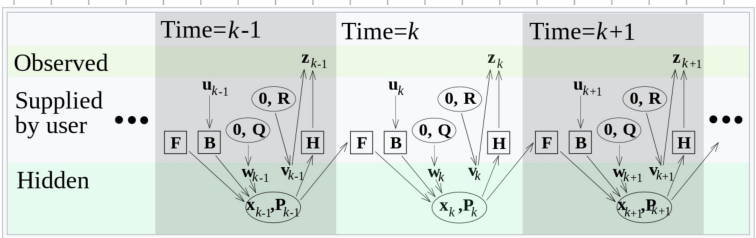
resp $\vec{x}_t = \underline{H}\vec{z}_t + \underline{D}\vec{u}_t + \vec{r}_t$ mit Messrauschen $\vec{r}_t \sim \mathcal{N}(\vec{0}, \underline{R})$



The Kalman filter keeps track of the estimated state of the system and the variance or uncertainty of the estimate. The estimate is updated using a state transition model and measurements. $\hat{x}_{k|k-1}$ denotes the estimate of the system's state at time step k before the k-th measurement y_k has been taken into account; $P_{k|k-1}$ is the corresponding uncertainty.

manchmal hängen einige der Parameter E, B, Q, H, D, R von der Zeit ab, was ohne nicht-stationäre Varianten nicht geht

ohne Eingaben $\vec{u}_t$ kann die Darstellung natürlich vereinfacht werden



Model underlying the Kalman filter. Squares represent matrices. Ellipses represent multivariate normal distributions (with the mean and covariance matrix enclosed). Unenclosed values are vectors. For the simple case, the various matrices are constant with time, and thus the subscripts are not used, but Kalman filtering allows any of them to change each time step.

Kalman Filter und Glätter

wie bei HMMs können auch bei LDSs explizite (rekursive) Formeln

$$p(\vec{z}_t | \vec{x}_{1:t}, \vec{u}_t) = \mathcal{N}(\vec{z}_t | \vec{\mu}_{t|t}, \underline{\Sigma}_{t:t})$$

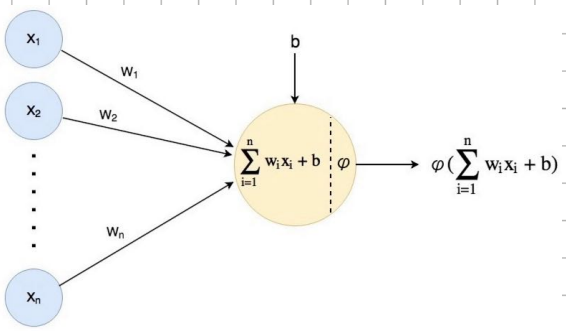
$$p(\vec{z}_t | \vec{x}_{1:T}, \vec{u}_T) = \mathcal{N}(\vec{z}_t | \vec{\mu}_{t|T}, \underline{\Sigma}_{t:T})$$

für die Filter- und Glätter-Verteilungen berechnet werden

für das Lernen gibt es mehrere Möglichkeiten:

- EM, wobei die Verteilungen über die verborgenen Zustände mithilfe der Kalman-Filter und Kalman-Glätter Rekursionen berechnet werden
- Bayesche Ansätze, welche auf Variationsapproximationen (Variational Bayes-EM) oder Monte-Carlo-Methoden basieren

Neuronale Netze



Basisfunktionen adaptiv, Parameter der Basisfamilie werden im Lernprozess einbezogen

Feedforward Neural Net (FNN)

eine verborgene Schicht:

$$\vec{f}_{2,\vec{\theta}_2} \circ \vec{f}_{1,\vec{\theta}_1} : \mathbb{R}^{n_{in}} \to \mathbb{R}^{n_{hidden}} \to \mathbb{R}^{n_{out}}$$
$$\vec{x} \mapsto \vec{f}_{2,\vec{\theta}_2}\left(\vec{f}_{1,\vec{\theta}_1}(\vec{x})\right) = \vec{\varphi}_2\left(\underline{W}_2\left(\vec{\varphi}_1\left(\underline{W}_1\vec{x} + \vec{b}_1\right)\right) + \vec{b}_2\right)$$

ein solches Modell (1 verborgene Schicht resp 1 Basisexpansion) kann schon mit einer einfachen Aktivierungsfunktion (zB ReLU) und der linearen Ausgabeaktivierung jede stetige Funktion auf einer kompakten Menge gleichmässig approximieren, wenn n_{hidden} genügend gross gewählt wird

es hat sich jedoch als lohnend erwiesen iterierte Basisexpansion zu verwenden anstatt breiter zu werden:

$$\vec{f}_{\vec{\theta}} : \mathbb{R}^{n_{in}} \to \cdots \to \mathbb{R}^{n_{out}} \quad \text{mit } \vec{\theta} = (\vec{\theta}_1, \dots, \vec{\theta}_K) \in \mathbb{R}^{p_1} \times \cdots \times \mathbb{R}^{p_K}$$
$$\vec{x} \mapsto \vec{f}_{K,\vec{\theta}_K} \circ \cdots \circ \vec{f}_{1,\vec{\theta}_1}(\vec{x}) \text{ wobei: } \vec{f}_{j,\vec{\theta}_j}(\vec{a}) = \vec{\varphi}_j(\underline{W}_j\vec{a} + \vec{b}_j)$$

die Ausgabe-Aktivierungsfunktion $\vec{\varphi}_K$ wird durch die Anwendung bestimmt

dense layer: volle Matrix und Verschiebungsvektor sind adaptive Parameter

tiefe FNNs mit dense layers nicht minimaler Breite sind sehr flexibel (no bias, high variance)
→ Regularisierung und viel Trainingsdaten nützlich

obwohl durch Verwendung von partitionierten und strukturierten Matrizen auch komplexere Architekturen mit Faltungen, Skip-Connections durch das Schichtmodell dargestellt werden können, ist es irgendwann sowohl für die Modellierung als auch die Implementierung natürlicher das Netz als gewichteten, azyklischen Graphen von Basisexpansionen aufzufassen

Backpropagation

$$\vec{f}(\vec{x}, \vec{\theta}_1, \dots, \vec{\theta}_K) = \vec{f}_K\Big(\underbrace{\vec{f}_{K-1}\Big(\cdots \vec{f}_2\Big(\underbrace{\vec{f}_1(\vec{x}, \vec{\theta}_1)}_{\vec{a}_1}, \vec{\theta}_2\Big)\cdots, \vec{\theta}_{K-1}\Big)}_{\vec{a}_{K-1}}, \vec{\theta}_K\Big)$$

Ableitung nach $\vec{\theta}_j$:

$$\frac{\partial}{\partial \vec{\theta}_j}\vec{f}(\vec{x}, \vec{\theta}_1, \dots, \vec{\theta}_K) = \frac{\partial}{\partial \vec{a}}\vec{f}_K(\vec{a}_{K-1}, \vec{\theta}_K) \cdots \frac{\partial}{\partial \vec{a}}\vec{f}_{j+1}(\vec{a}_j, \vec{\theta}_{j+1}) \frac{\partial}{\partial \vec{\theta}}\vec{f}_j(\vec{a}_{j-1}, \vec{\theta}_j)$$ Kettenregel

diese Matrizen sollte man von links nach rechts berechnen (effizienter)
→ Ableitungsinformationsfluss rückwärts durchs Netz

Trainieren

diskriminatives, probabilistisches Modell mit ML:

$$NLL(\vec{\theta}) = -\ln p(\vec{t} \mid \underline{X}, \vec{\theta}) = -\sum_{k=1}^{n_D} \ln p(t_k \mid \vec{x}_k, \vec{\theta})$$

allfällige Regularisierungsterme (MAP) können dazugeaddiert werden

Minimierung davon nicht analytisch berechenbar ausserdem ist NLL nicht konvex, hat i.d.R. viele lokale nicht optimale Minima

Mini-Batch-Gradient:

$$\frac{dNNL}{d\vec{\theta}}(\vec{\theta}) \approx \frac{n_D}{n_B}\frac{dNNL_B}{d\vec{\theta}}(\vec{\theta}) = -\frac{n_D}{n_B}\sum_{j=1}^{n_B}\frac{\partial}{\partial\vec{\theta}}\ln p(t_{k_j}|\vec{x}_{k_j},\vec{\theta})$$

SGD-Iteration:

$$\vec{\theta}_{t+1} = \vec{\theta}_t - \eta_t \nabla_B f(\vec{\theta}_t)$$

Robbinson-Monro:

$$\sum_{t=1}^{\infty}\eta_t = \infty \text{ und } \sum_{t=1}^{\infty}\eta_t^2 < \infty$$

bekannte hinreichende Bedingung für lokale Konvergenz

Möglichkeiten, inhärente Flexibilität zu zähmen (Regularisierung):
- Prior-V. über Parameter, häufig unkorrelierte in null zentrierte Normalv.
- Early Stopping: Minimieren stoppen sobald Verlust auf Evaluationsdaten anzusteigen beginnt
- Faltungsschichten (Form von Parameter-Sharing)
- Bayes
- viele weitere...