Formale Methoden: Projekt

Machine Learning - Hidden Markov Model

Paul Pasler (Matr.Nr. 751325, huc)

Machine Learning

- Künstliche Generierung von Wissen / Modellierung von Lernen
  - Klassifizierung anhand von Merkmalen
  - Erkennen durch Generalisierung
- Ausprägungen (Auszug)
  - Neuronale Netze
  - Support Vector Machine
  - k-Means
- HMMs arbeiten mit Wahrscheinlichkeiten

Zeit-diskreter Prozess

- Zustände eines System ändern sich immer im selben zeitlichen Abstand (Äquidistante Zeitabstände)
- Beispiel: Jede Sekunde wird ein Tick ausgelöst, welcher zu einer Zustandsänderung führt
- Gegenstück: Zeit-kontinuierlicher Prozess

Bedingte Wahrscheinlichkeit

- Drückt die Wahrscheinlichkeit für das Eintreten eines Ereignisses A, unter Bedingung das vorher ein anderes Ereignis B eingetreten ist

$$P(A|B)=\frac{P(A\cap B)}{P(B)}$$

Markov Kette: Beispiel „Wetter“

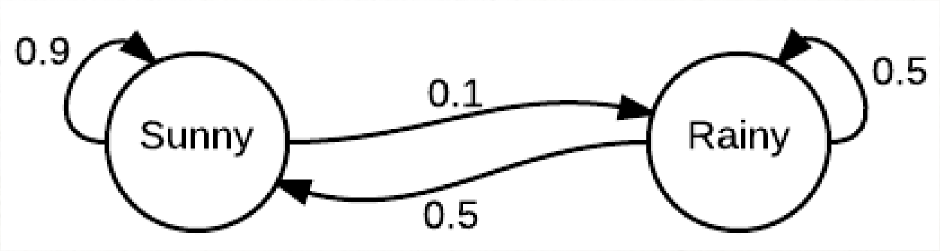
Vorhersage des Wetters

- Zustände X
- Übergangsmatrix A
- Startwahrscheinlichkeiten Π

$$X = [\,'sunny', 'rainy'\,]$$

$$A = \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix}$$

$$\Pi = x_0 = [1, 0]$$



http://upload.wikimedia.org/wikipedia/commons/7/7a/Markov_Chain_weather_model_matrix_as_a_graph.png

Markov Kette: Beispiel „Wetter“

- Tag 1 $$X_1 = X_0 \cdot A = [1,0] \cdot \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix} = [0.9, 0.1]$$

- Tag 2 $$X_2 = X_1 \cdot A = X_0 \cdot A^2 = [1,0] \cdot \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix}^2 = [0.86, 0.14]$$

- Allgemein Tag k

$$X_k = X_{k-1} \cdot A = X_0 \cdot A^k = [1,0] \cdot \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix}^k$$

Markov-Kette

- Statistische Beschreibung von Zustands- / Symbolfolgen [Mar13]
- Kann als Zustandsgraph veranschaulicht werden
- Definiert durch:
  - Menge von Zuständen X
  - Übergangswahrscheinlichkeits-Matrix A
  - Startwahrscheinlichkeiten Π

Markov-Eigenschaft

- Die bedingte Wahrscheinlichkeit für den aktuellen Zustand, hängt **nicht** von allen Vorgängern ab

$$P(X_{t+1}=s_{t+1} \mid X_0=s_{0,} \ldots, X_{t-1}=s_{t-1}, X_t=s_t)$$

- Der Folgezustand hängt nur vom aktuellen Zustand ab (Gedächtnislos)

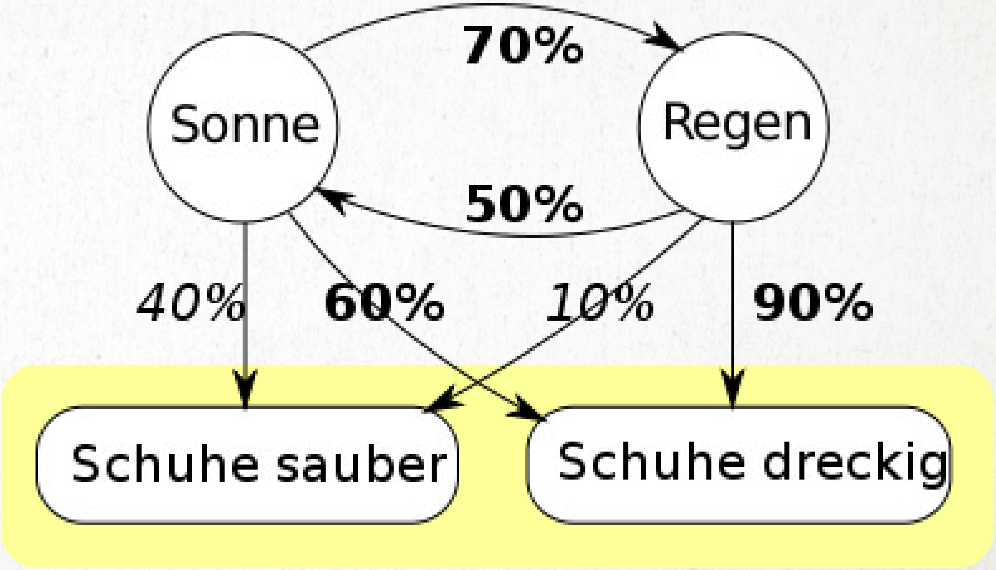
$$P(X_{t+1}=s_{t+1} \mid X_t=s_t)$$

Hidden Markov Model: Beispiel „Gefangener im Verlies“

- Gefangener im Verlies möchte wissen, welches Wetter draußen ist
- Er kann den Himmel aber nicht sehen (x), jedoch sieht er die Schuhe der Wärter (y)
- Zudem weiß er,
  - wie wahrscheinlich Regen auf Sonne und
    Sonne auf Regen folgt (a)
  - wie wahrscheinlich das Wetter und
    die Schuhe der Wärter zusammenhängen (b)

Hidden Markov Model: Beispiel „Gefangener im Verlies“

- Der Gefangene sieht nur die Schuhe der Wärter
- Der Gefangen kann Rückschlüsse auf das aktuelle Wetter ziehen, ohne den Himmel zu sehen.
- Zumindest welches Wetter am wahrscheinlichsten ist



http://upload.wikimedia.org/wikipedia/de/d/dc/Hidden_markov_model.svg

Hidden Markov Model: Beispiel „Spracherkennung“

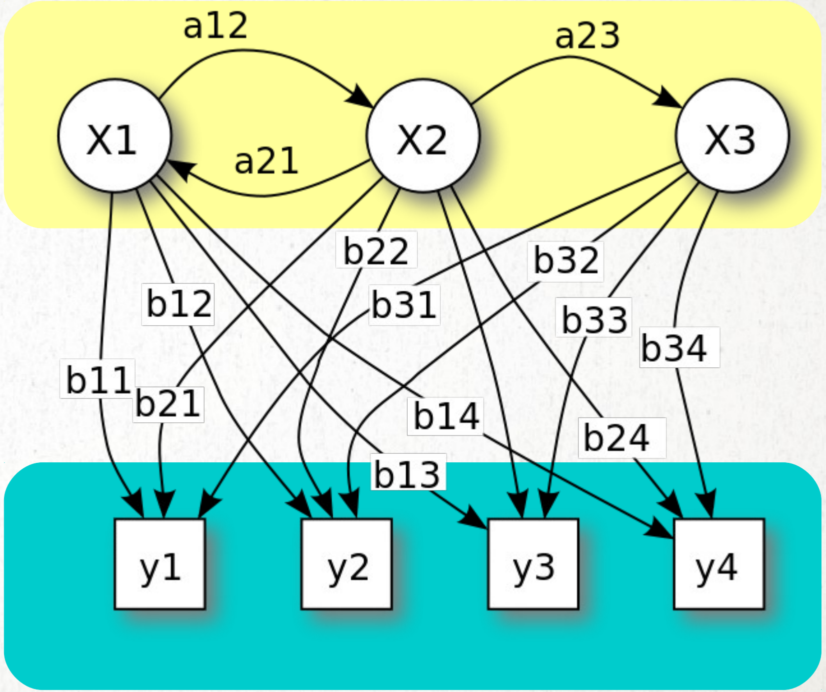
- Versteckte Zustände entsprechen gesprochenen Lauten (Phoneme)
- Silben oder Wörter sind die Beobachtungen
- Das HMM schließt aus einer Sequenz von Lauten
  auf eine Sequenz von Silben / Wörtern
- Das HMM passt sehr gut zur Vorstellung
  von Sprachsignalen als Abfolge (Sequenz) einzelner Ereignisse

Hidden Markov Model

- Ende der 60er Jahre von L.E. Baum [BP66] entwickelt, Weiterentwicklung Lawrence R. Rabiner [Rab89]
- Ziel: Erkenntnisse über ein System gewinnen,
  ohne das genau Aussehen / die Zuständen sehen zu können
- Basiert auf der Markov-Kette und beschreibt einen 2-stufigen stochastischen Prozess
- Einsatzgebiete
  - Sprach- / Gesten- und Schrifterkennung
  - Computerlinguistik
  - Bioinformatik

Hidden Markov Model

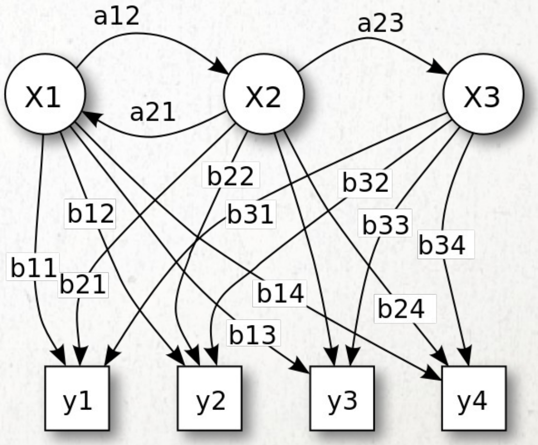
- Versteckter Teil (Gelb)
  - X: Versteckte Zustände
  - a: Übergangswahrscheinlichkeiten
- Sichtbarer Teil (Blau)
  - y: Observationen
    (Emissionen, Beobachtungen)
  - b: Observationswahrscheinlichkeiten



Definition

- Erweiterung einer Markov-Kette um einen weiteren Zufallsprozess (2-Stufig)
- Zustände sind versteckt, nur Beobachtungen (Observationen) sind sichtbar
    - Das Ergebnis ist eine Observationsfolge

$$o \in Y \, ; o = y_a , \ldots , y_z$$



Definition

- Ein HMM ist vollständig definiert durch: $\lambda = \left(X ; Y ; A ; B ; \pi \right)$

- Endliche Menge an Zuständen $X = \{ x | 1 \leq x \leq N \}$

- Endliche Menge an Observationen $Y = \{ y | 1 \leq y \leq M \}$

- Zustandsstartwahrscheinlichkeiten $\Pi = \{ \pi_i | \pi_i = P(X_1 = i) \}$

Definition

- Ein HMM ist vollständig definiert durch: $\lambda = (X; Y; A; B; \pi)$

- Übergangswahrscheinlichkeiten $A = \{a_{ij} | a_{ij} = P(X_t = j | X_{t-1} = i)\}$

- Observationswahrscheinlichkeiten

  Diskreter Fall $B = \{b_{jk} | b_{jk} = P(Y_t = o_k | X_t = j)\}$

  (Kontinuierlicher Fall) $B = \{b_j(x) | b_j(x) = p(x | X_t = j)\}$

Funktionsweise

Das Konzept des HMM lässt sich in drei größere Problemstellungen unterteilen

- Evaluierung
  Bestimme die Wahrscheinlichkeit für ein Model,
  mit der dieses, eine gegebene Observationsfolge (o) erzeugt

- Dekodierung
  Finde interne Abläufe für eine gegebene Observationsfolge

- Training
  Finde Modellparameter für gegebene Beispieldaten

Evaluierung

- Gegeben: HMM λ, Observationsfolge o
- Gesucht: Die (Produktions-) Wahrscheinlichkeit,
  mit der o in einer beliebigen Zustandsfolge generiert wird

$$p(o|\lambda) \Rightarrow \alpha_t(i) = P(o_{1,} o_{2,} \ldots, o_t, x_t = i|\lambda)$$

- Lösung: Der Forward-Algorithmus

Forward-Algorithmus

Forward Variable $$\alpha_t(i) = P(o_{1,} \ldots, o_t; q_t = x_i | \lambda)$$

- Initialisierung $$\alpha_1(i) = \pi_i \cdot b_i(o_1)$$

- Rekursion $$\alpha_{t+1}(j) = \sum_{i=1}^{|X|} \{\alpha_t(i) a_{ij}\} b_j(o_{t+1})$$

- Rekursionsabschluss $$P(O|\lambda) = \sum_{i=1}^{|X|} \alpha_t(i)$$

Dekodierung

- Gegeben: HMM λ, Observationsfolge o
- Gesucht: Die wahrscheinlichste Zustandsfolge s* für das gegebene o

$$\delta_t(i) = \max_{x_1, \ldots, x_t} \left(P\left(o_1, \ldots, o_t; x_1, \ldots, x_t = i \mid \lambda\right)\right)$$

- Lösung: Der Viterbi-Algorithmus

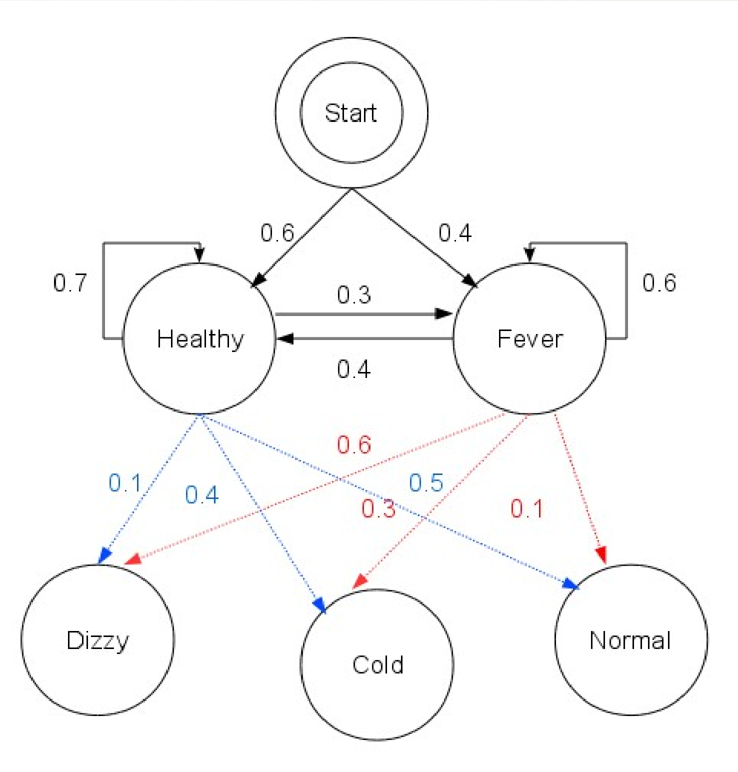
Viterbi-Algorithmus [Vit06]

- Initialisierung
  - Max. Verbundwahrscheinlichkeit $$\delta_1(i) = \pi_i \cdot b_i(o_1)$$
  - Weg-Variable $$\varphi_1(i) = 0$$
- Rekursion
  $$\delta_{t+1}(j) = max(\delta_t(i)\, a_{ij}) \cdot b_j(o_{t+1})$$
  $$\varphi_{t+1}(j) = argmax_i(\lambda_t(i)\, a_{ij})$$
- Rekursionsabschluss
  $$P^*(O|\lambda) = P(O, x^*|\lambda) = max_i \lambda_T(i)\, \alpha_T(i)$$
- Rückverfolgung des Pfades
  $$s_1^* = \varphi_{t+1}(s_{t+1}^*)$$

Viterbi-Algorithmus: Code Beispiel

- Dorfbewohner sind gesund oder haben Fieber
- Initialisierung

```
states = ('Healthy', 'Fever')

observations = ('normal', 'cold', 'dizzy')

start_probability = {'Healthy': 0.6, 'Fever': 0.4}

transition_probability = {
   'Healthy' : {'Healthy': 0.7, 'Fever': 0.3},
   'Fever' : {'Healthy': 0.4, 'Fever': 0.6}
   }

emission_probability = {
   'Healthy' : {'normal': 0.5, 'cold': 0.4, 'dizzy': 0.1},
   'Fever' : {'normal': 0.1, 'cold': 0.3, 'dizzy': 0.6}
   }
```



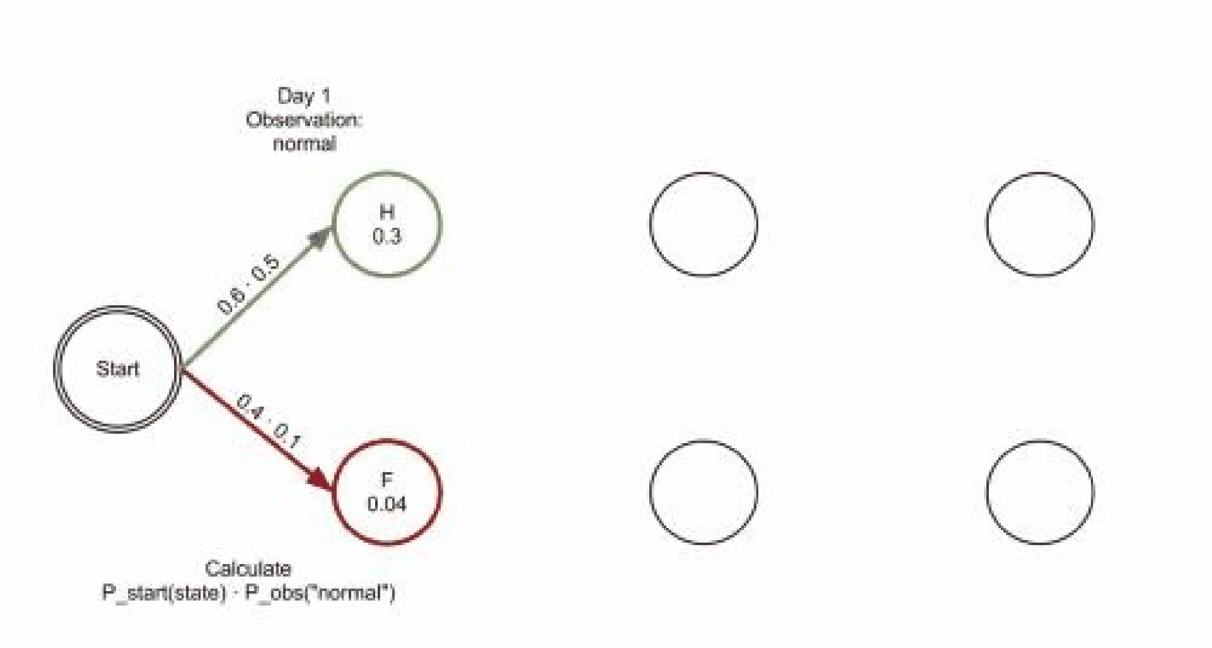
Quelle: https://en.wikipedia.org/wiki/Viterbi_algorithm

Viterbi-Algorithmus: Code Beispiel

```
def viterbi(obs, states, start_p, trans_p, emit_p):
    V = [{}]
    path = {}

    # Initialize base cases (t == 0)
    for y in states:
        V[0][y] = start_p[y] * emit_p[y][obs[0]]
        path[y] = [y]

    # Run Viterbi for t > 0
    for t in range(1, len(obs)):
        V.append({})
        newpath = {}

        for y in states:
            (prob, state) = max((V[t-1][y0] * trans_p[y0][y] * emit_p[y][obs[t]], y0) for y0 in states)
            V[t][y] = prob
            newpath[y] = path[state] + [y]

        # Don't need to remember the old paths
        path = newpath
    n = 0           # if only one element is observed max is sought in the initialization values
    if len(obs) != 1:
        n = t
    print_dptable(V)
    (prob, state) = max((V[n][y], y) for y in states)
    return (prob, path[state])
```


Quelle: https://en.wikipedia.org/wiki/Viterbi_algorithm

Viterbi-Algorithmus: Animation



Training

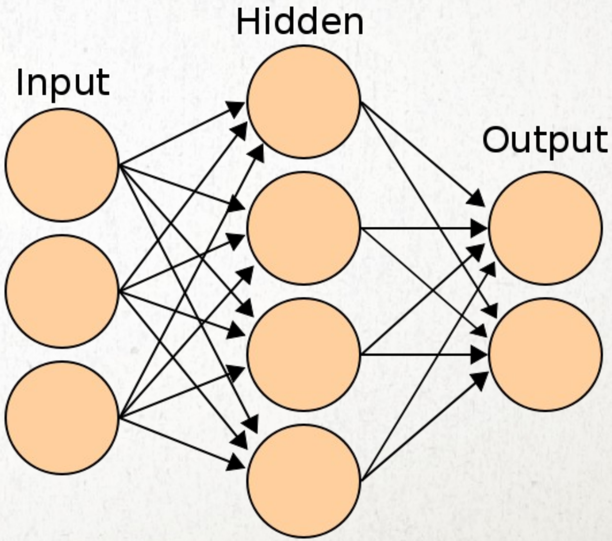
- Gegeben: Observationsfolgen(n) o
- Gesucht: Parameter eines HMM,
  die diese Ausgabe am wahrscheinlichsten erzeugen
- Lösung: Es existieren diverse Algorithmen
  - Der Baum-Welch-Algorithmus nutzt die
    Produktionswahrscheinlichkeit
  - Der Viterbi-Algorithmus / seqmental k-Means nutzen nur die
    Wahrscheinlichkeit der optimal Zustandsfolge

Vergleich mit anderen Machine Learning Ansätzen

- Sehr viele Ansätze im Machine Learning Kontext
- Jedes Verfahren
    - wurde für einen speziellen Anwendungsfall entwickelt
    - hat unzählige Erweiterung, Algorithmen und Implementierungen
- Vergleiche sind daher schwierig und hängen von den Anforderungen ab

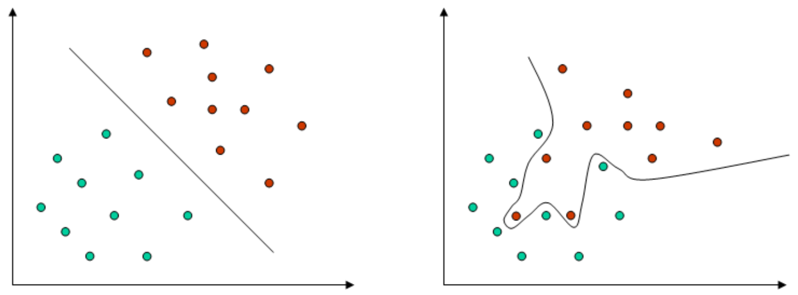
Neuronale Netze

- Einfaches Modell des menschlichen Gehirns (Neuronen, Synapsen)
- Beim Training verändern sich Gewichte an den Synapsen
- Die Gewichte sind aber keine nachvollziehbaren Größen
- Benötigen sehr viele Trainingsdaten
- Nicht so gut für sequentielle Daten geeignet (HMM schon)
- Hybride mit dem HMM für die Spracherkennung
  (Jeder Zustand ist Eingang eines Neuron)



Support Vector Machine

- SVM versuchen die Daten geometrisch zu trennen
- Gelingt keine lineare Trennung (häufig der Fall) wird der Kernel-Trick angewendet und die Daten in einen höher-dimensionalen Raum überführt
- Nicht für große Datenmengen geeignet
  (HMM ist schneller)
- Kann mit wenig Trainingsdaten arbeiten
- Auch hier Hybride mit dem HMM
  Variabilität mit HMM
  Akustische Komponente mit SVM

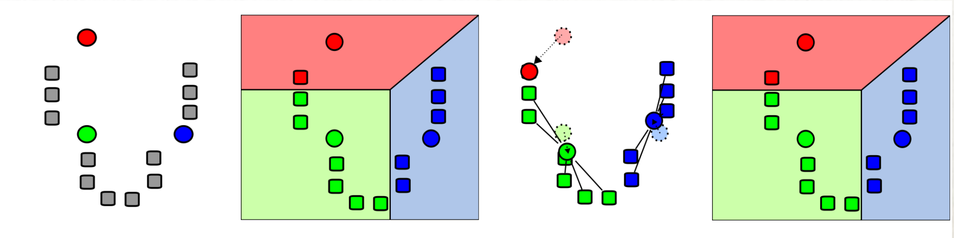


linear trennbar

nicht linear trennbar

k-Means

- k-Means versucht de Daten zu gruppieren / clustern (in „k“ Cluster)
- Die Clusterzentren werden zufällig in das Koordinatensystem eingefügt und bei jedem Schritt besser an die Daten angepasst
- Ansatz für unüberwachtes Lernen (Wie das HMM)
- Hybride für schreiberunabhängige Schrifterkennung



Fazit

- Das HMM eignet sich besonders für die Verarbeitung von sequentiellen Daten
    - Durch das Verweilen in internen Zuständen kommt das HMM auch mit einem lang gezogen Laut klar
- Mathematisches Konzept gut erforscht und formal definiert
    - Parameter sind Wahrscheinlichkeiten und somit „verstehbar“
    - Vorwissen erforderlich, was den Einstieg u.U. erschwert
- Viele hybride Ansätze mit Stärken aus HMM und weiteren Algorithmen

Quellen

[AD77] D.B. Rubin A.P. Dempster, N.M. Laird. Maximum-Likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, 1977.

[BGV92] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT ’92, pages 144–152, New York, NY, USA, 1992. ACM.

[BM94] H. Bourlard and N. Morgan. Connectionist speech recognition. A hybrid approach. Kluwer international series in engineering and computer science, 1994.

[BP66] Leonard E. Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. Ann. Math. Statist., 37(6):1554–1563, 12 1966.

[Eul05] S. Euler. Grundkurs Spracherkennung. Friedr. Vieweg und Sohn Verlag - GWV Fachverlage GmbH, Wiesbaden, 2005.

[Fin03] Gernot A. Fink. Mustererkennung mit Markov-Modellen. B. G. Teubner Verlag, 2003.

[JR90] Biing-Hwang Juang and L. R. Rabiner. The segmental K-means algorithm for estimating parameters of hidden Markov models. IEEE Transactions on Acoustics, Speech, and Signal Processing, 38:1639–1641, 1990.

Quellen

[KHW13] U.M. Stocker K.-H. Waldmann. Stochastische Modelle - Eine anwendungsorientierte Einführung. Springer-Verlag, Berlin Heidelberg, 2013.

[Mar13] A.A. Markov. Example of a statistical investigation of the text of “eugene onegin” illustrating the dependence between samples in chain., 1913.

[Mar09] Stephen Marsland. Machine Learning - An Algorithmic Perspective. Chapman I& Hall, 2009.

[MP88] Warren S. McCulloch and Walter Pitts. Neurocomputing: Foundations of research. chapter A Logical Calculus of the Ideas Immanent in Nervous Activity, pages 15–27. MIT Press, Cambridge, MA, USA, 1988.

[PC00] Michael P. Perrone and Scott D. Connella. K-means clustering for hidden markov models. 2000.

[Rab89] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In Proceedings of the IEEE, pages 257–286, 1989.

[Ste57] Hugo Steinhaus. Sur la division des corps matériels en parties. Bull. Acad. Polon. Sci., 12:801–804, 1957.

[Stu08] André Stuhlsatz. Hybride Spracherkennung: Eine HMM/SVM- Systemintegration. VDM Verlag Dr. Müller, 2008.

[Vit06] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Trans. Inf. Theor., 13(2):260–269, September 2006.

Danke, noch Fragen?