

Mustererkennung: Clusterung, Cohonen Netze

Aufgabe: partitioniere eine Menge der Objekte auf sinnvolle Teile – Clusters.
Die Objekte eines Clusters sollen „ähnlich“ sein.

Clustermenge: K

Indexmenge: $I = \{1, 2, \dots, |I|\}$

Merkmalsvektoren: $x^i, i \in I$.

Partitionierung: $C = (I_1, I_2, \dots, I_{|K|}), I_k \cap I_{k'} = \emptyset$ für $k \neq k', \bigcup_k I_k = I$

$x^i \in \mathbb{R}^n$, jeder Cluster hat einen „Repräsentant“ $y^k \in \mathbb{R}^n$

Die Aufgabe:

$$\sum_k \sum_{i \in I_k} \|x^i - y^k\|^2 \rightarrow \min_{C, y}$$

Alternativ – gesucht wird eine Abbildung $C : I \rightarrow K$

$$\sum_i \|x^i - y^{C(i)}\|^2 \rightarrow \min_{y, C}$$

$$\sum_i \min_k \|x^i - y^k\|^2 \rightarrow \min_y$$

Initialisiere Clusterzentren y^k zufällig.

Wiederhole bis sich die Clusterung C ändert:

1) Klassifikation:

$$C(i) = \arg \min_{k'} \|x^i - y^{k'}\|^2 \quad \Rightarrow \quad i \in I_k$$

2) Aktualisierung der Zentren:

$$y^k = \arg \min_y \sum_{i \in I_k} \|x^i - y\|^2 = \frac{1}{|I_k|} \sum_{i \in I_k} x^i$$

-
- NP-vollständig.
 - K-Means Konvergiert zum lokalen Optimum $\rightarrow$ abhängig von der Initialisierung (Beispiel lokaler Konvergenz auf der Tafel).

Varianten/Verallgemeinerungen

Ein anderer Abstandsmaß, zum Beispiel $\|x^i - y^k\|$ anstatt $\|x^i - y^k\|^2$:
beim K-Means ist die Klassifikation 1) dasselbe,
die Aktualisierung 2) – der geometrische Median der Punkte x^i , $i \in I_k$:

$$y_k = \arg \min_y \sum_{i \in I_k} \|x^i - y\|$$

(etwas schwieriger als der Mittelpunkt).

Problem: die Merkmale x lassen sich nicht immer mitteln (y^k existiert nicht).

Ein Ausweg – K-Mediod Algorithmus (y^k ist ein Punkt x^i aus der Lernstichprobe).

Eine andere Verallgemeinerung basiert auf der Beobachtung

$$\sum_i \|x^i - \bar{x}\|^2 \sim \sum_{ij} \|x^i - x^j\|^2,$$

daraus folgt

$$\sum_k \sum_{ij \in I_k} \|x^i - x^j\|^2 = \sum_k \sum_{ij \in I_k} d(i, j) \rightarrow \min_C,$$

mit der Abstandsmatrix d (die auf unterschiedlichste Weisen definiert werden kann).

Abstandsmaße

Graph basierte Abstandsmaße:

Gegeben ist ein Graph, dessen Knoten die Elementen von I sind. Jede Kante ist mit $d(i, j)$ bewertet. Der Abstand zwischen i und j ist die (summarische) Länge des kürzesten Pfaden zwischen den entsprechenden Knoten im Graphen.

→ So ein Abstandsmaß ist eine Metrik.

Pfad basierte Abstandsmaße (auch Graphen):

Die Idee – selbst wenn zwei Merkmale x^i und x^j von einander weit entfernt sind, gehören sie eher zum selben Cluster, wenn ein Pfad $(x^i, x^l, x^{l'}, \dots, x^j)$ existiert so, dass die Abstände $d(x^l, x^{l'})$ klein sind.

→ Der minimale aufspannende Baum wird benötigt.

Abstandsmaße für „andere“ Objekte (nicht $\in \mathbb{R}^n$):

Zum Beispiel:

Edit distance (Levenstein Abstand) zwischen zwei Folgen,

Graph Isomorphismus basierte Abstände zwischen Graphen

Noch eine Variante – Minimierung der Durchmesser (bei $|K| = 2$ polynomiell lösbar):

$$\max_k \max_{ij \in I_k} d(i, j) \rightarrow \min_C$$

...

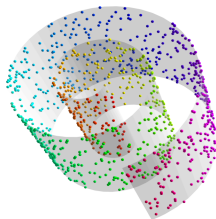
Die Elementen sind die Pixel des Bildes, die Merkmale sind Farbwerte, man zerlege das Bild auf Teile, die jeweils „charakteristischen Farben“ entsprechen.

Beispiel der Farbreduktion auf 8 Farben:



Selbstorganisierende Karten (Self Organizing Maps – SOM).

Die Aufgabe ist, die gegebene Datenmenge durch ein neuronales Netz vorgegebener Topologie „zu approximieren“ → Clusterung.



Gegeben sei eine Menge der Datenpunkten in $\mathbb{R}^n$, die einem Objekt entsprechen (nach welchem schließlich gesucht wird). Zusätzlich sei bekannt, dass der Objekt bestimmte topologische Eigenschaften besitzt. Zum Beispiel ist der Objekt eine Untermannigfaltigkeit niedriger Dimension.

Beispiel 1: Der Objekt ist eine 1D-Linie im 2D, d.h. sie ist durch eine Menge der schwarzen Pixel im $\mathbb{R}^2$ dargestellt.

Beispiel 2: Gegeben sei die Menge der Punkte im 3D-Raum. Gesucht wird nach einem Mesh (ein Dreiecksnetz – eine 2D-Untermannigfaltigkeit).

Cohonen Netze bestehen (meist) aus RBF-Neuronen r so, dass jedes Neuron einer Untermenge des Input-Raums entspricht. Dies erfolgt durch geeignete Wahl der Parameter (z.B. des Zentrums des RBF-Neurons μ^r).

Die Menge der Neuronen ist mit einem Distanzmaß versehen, das der gewünschten Topologie entspricht, d.h. für jedes Paar von Neuronen (r, r') gibt es einen Abstand $d(r, r')$.

Beispiel: die Neuronen sind die Knoten eines Graphen. Der Abstand $d(r, r')$ ist der kürzeste Weg von r nach r' .

Beispiel 1: der Graph ist eine Kette, die gewünschte Topologie entspricht einem 1D-Objekt.

Beispiel 2: der Graph ist ein 2D-Fitter – siehe Bildchen auf der vorigen Folie.

Zusammenfassung:

- RBF-Neuronen, jedes für sein Teilraum verantwortlich.
- Die Menge der Neuronen besitzt eine Topologie.
- Das unüberwachte Lernen (Clusterung, Approximation der Datenmenge).

Lernalgorithmus (sequenzielle Variante):

- 1) Nehme zufällig ein Muster x aus der Lernstichprobe
- 2) Bestimme das „Gewinner-Neuron“:

$$r^* = \arg \min_r \|x - \mu^r\|$$

- 3) Bestimme die Umgebung des Gewinner-Neurons im Netz:

$$R = \{r | d(r, r^*) < \Theta\}$$

- 4) Aktualisiere die Gewichte aller Neuronen aus R :

$$\mu^r = \mu^r + (x - \mu^r) \cdot \eta(t, d(r^*, r))$$

Varianten je nach Art der Funktion $\eta(t, d)$ (t ist die Zeit).

Generell ist η monoton fallend in t und d .

Ohne 3) und $d(r, r^*) \rightarrow$ sequenzielle Variante des K-Means Algorithmus.

Parallele Variante:

– gehe über alle Datenpunkte, summiere Gradienten, wende sie anschließend an.

Cohonen Netze – Beispiel

Gesucht wird ein Gitter (4-Nachbarschaft),
das die gegebene Punktwolke (im $\mathbb{R}^2$) approximiert.

Entwicklung der Geometrie des Netzes (d.h. μ^r) in Zeit:

