

Einführung in die Künstliche Intelligenz

Kapitel 6: Maschinelles Lernen Teil 5: Unüberwachtes Lernen

Prof. Dr. Johannes Maucher

HdM MIB

Version 1.9
22.01.2022

Document History

Version Nr.	Date	Changes
1.0	10.06.2009	
1.1	24.06.2010	
1.2	17.01.2011	Folien zum Information Retrieval hinzugefügt
1.3	20.06.2011	Folien zur Assoziationsanalyse und zu SOMs hinzugefügt
1.4	04.01.2013	Anpassungen für WS 12/13
1.5	21.01.2013	Weitere Korrekturen (K-Means Ablaufdiagramm) und Anpassungen.
1.6	13.01.2014	DBSCAN hinzugefügt.
1.7	15.01.2018	Anpassungen für WS 17/18
1.8	25.06.2018	Kapitel Self Organizing Maps entfernt
1.9	22.01.2022	Anpassungen für WS 21/22

Übersicht Kapitel Unüberwachtes Lernen

- 1 Einführung
 - Prinzip und Ziel
 - Anwendungen
- 2 Distanz- und Ähnlichkeitsmaße
 - Einführungsbeispiele
 - Gebräuchliche Maße
 - Ähnlichkeitsdarstellung in Matrix
- 3 Standardisierung
- 4 Clustering
 - Hierarchisches Clustering
 - K-Means Clustering
 - DBSCAN
- 5 Association Rule Mining
 - Einführung
 - Items und Transaktionen
 - Association Rule
 - Bestimmung von Association Rules

Unüberwachtes Lernen

- Im Gegensatz zum überwachten Lernen stehen beim unüberwachten Lernen **nur Eingabedaten** zur Verfügung. Trainingsmenge ist von der Form

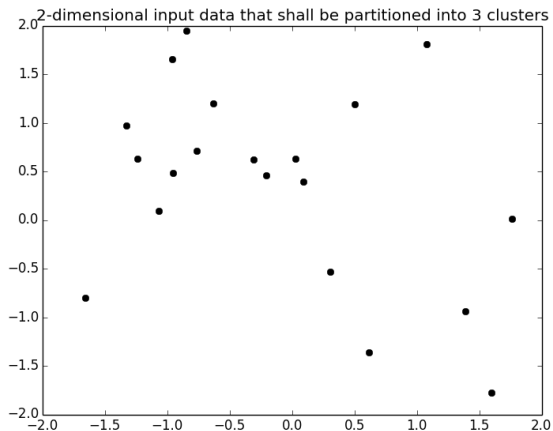
$$T = \{(\mathbf{x}_p)\}, \quad p \in \{1 \dots N\}$$

- Unterschieden werden folgende zwei Kategorien des unüberwachten Lernens:
 - Clusteranalyse** Bestimme **Ähnlichkeit zwischen Instanzen** und fasse ähnliche Instanzen in Cluster zusammen. Unähnliche Instanzen werden unterschiedlichen Clustern zugeteilt.
 - Assoziationsanalyse** Finde Assoziationen (Regeln) **zwischen Merkmalen**.

	Merkmal 1	Merkmal 2	...	Merkmal K
Instanz 1	$x_{1,1}$	$x_{1,2}$...	$x_{1,K}$
Instanz 2	$x_{2,1}$	$x_{2,2}$...	$x_{2,K}$
:	:	:	:	:
Instanz N	$x_{N,1}$	$x_{N,2}$...	$x_{N,K}$

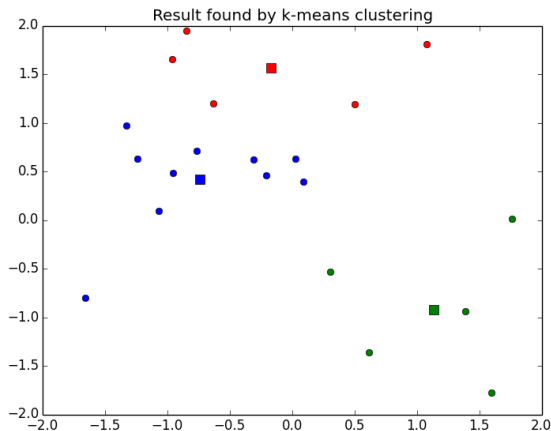
- In bisher behandelten **Klassifikationsaufgaben** war Klassenzugehörigkeit der Trainings-Eingabedaten vorgegeben. Im Fall des unüberwachten Lernens muss diese Klassenzugehörigkeit der Trainingselemente geschätzt werden.

Beispiel für Clustering von Objekten mit 2 Variablen



- Wie werden die Daten in 3 Cluster partitioniert?

Beispiel für Clustering von Objekten mit 2 Variablen



- Von einem k-means Clustering gefundene Partitionierung ($k = 3$)

Clustering Anwendungen

- Clustering von Kunden- oder Produktgruppen
- Mustererkennung, z.B. in der Bildverarbeitung, Segmentierung
- Nichtlineare Vektorquantisierung wie sie z.B. in nahezu allen modernen Sprachcodecs eingesetzt wird
- Farbquantisierung
- Clustering von Dokumenten
- Clustering von Musiktiteln \Rightarrow Automatische Playlistgenerierung
- Bioinformatik, z.B. für die Kategorisierung von Genen
- Bildung von Communities in Social Networks
- Autovervollständigung z.B. bei der Eingabe von Suchbegriffen

Aspekte des Clustering

- Clusteringalgorithmus z.B. kmeans-Algorithmus, hierarchisches Clustering, usw.
- Ähnlichkeitsmaß festlegen, z.B. euklidische Distanz, Cosinus-Ähnlichkeit, Pearson Distanz, Jaccard-Distanz
- Preprocessing z.B. Normierung, sodass Varianz hinsichtlich aller Attribute gleich 1 ist (whiten)
- Bewertung des gefundenen Clusterings

Beispiel 1:

Die nebenstehende Tabelle zeigt einen Ausschnitt aus einer Kundendatenbank

- Welche Kunden sind sich am ähnlichsten?
- Wie läßt sich Ähnlichkeit numerisch berechnen?
- Wie läßt sich Ähnlichkeit möglichst übersichtlich (eventuell grafisch) darstellen?
- Wie können Gruppen ähnlicher Instanzen (hier: Kunden) berechnet werden?

Viewer

Relation: kundenkarte.txt-weka.filters.unsupervised.attribute.Remove-R7

No.	Besuche Numeric	Einkaufswert Numeric	Alter Numeric	Jahre Numeric	Long Numeric	Lat Numeric
1	4.0	736.0	31.0	3.0	48.73605433	9.09040058
2	2.0	777.0	27.0	3.0	48.73212311	9.09730563
3	2.0	745.0	26.0	4.0	48.73688473	9.10854931
4	4.0	735.0	25.0	4.0	48.729635	9.11304053
5	2.0	754.0	31.0	2.0	48.72015437	9.09905785
6	3.0	794.0	27.0	4.0	48.72084744	9.09970837
7	3.0	794.0	27.0	4.0	48.72341142	9.10516778
8	3.0	983.0	32.0	3.0	48.72971999	9.10217957
9	2.0	700.0	32.0	2.0	48.73655442	9.12464164
10	4.0	809.0	31.0	4.0	48.73150301	9.10487871
11	2.0	851.0	32.0	3.0	48.7197011	9.09442741
12	1.0	838.0	27.0	4.0	48.72376392	9.0987183
13	5.0	716.0	31.0	2.0	48.73261004	9.11948255
14	3.0	943.0	28.0	4.0	48.73091606	9.09912767
15	3.0	737.0	30.0	2.0	48.71788325	9.10163125
16	7.0	35.0	17.0	4.0	48.73268537	9.11389672
17	8.0	37.0	11.0	2.0	48.7241735	9.11452133
18	8.0	45.0	18.0	4.0	48.73238758	9.10007342
19	10.0	72.0	14.0	5.0	48.71957602	9.11423898
20	9.0	47.0	20.0	2.0	48.73560597	9.08993816
21	8.0	68.0	17.0	2.0	48.72321133	9.1158245
22	9.0	26.0	12.0	2.0	48.72635014	9.11448958
23	6.0	59.0	19.0	5.0	48.72960021	9.115888
24	7.0	99.0	12.0	3.0	48.72744707	9.10687799
25	9.0	61.0	20.0	4.0	48.72790526	9.0963387
26	6.0	33.0	13.0	4.0	48.73560643	9.11182193
27	8.0	30.0	15.0	3.0	48.72296396	9.08904385
28	8.0	58.0	19.0	2.0	48.73486046	9.11108185
29	7.0	66.0	19.0	3.0	48.72488586	9.09037418

Undo OK Cancel

Beispiel 2:

- Für ein Recommender-System soll die Ähnlichkeit zwischen Kunden bestimmt werden. Hierzu werden die bisherigen Einkäufe herangezogen: Der Verkäufer hat insgesamt 8 Produkte im Angebot.
 - Kunde 1 hat die Produkte 1 bis 7 gekauft
 - Kunde 2 hat die Produkte 1 bis 5 gekauft
 - Kunde 3 hat das Produkt 8 gekauft
 - Kunde 4 hat das Produkt 7 gekauft
- Wie läßt sich Ähnlichkeit numerisch berechnen?

Beispiel 3:

- Für ein DVD Recommender-System soll die Ähnlichkeit zwischen Kunden basierend auf bisherigen Filmbewertungen berechnet werden

	Film 1	Film 2	Film 3	Film 4
Kunde 1	1.5	1.5	2.5	3.5
Kunde 2	3	3	4	5
Kunde 3	3	3	2	1

- Wie läßt sich Ähnlichkeit numerisch berechnen?

Beispiel 4:

Welche der folgenden Texte sind sich am Ähnlichsten?

- 1 *Der Papst schreibt von Reue, von Schande und entschuldigt sich bei den Opfern - doch das reicht vielen Iren nicht: Sie kritisieren Benedikts Hirtenbrief zum Missbrauchsskandal als Versuch der Vertuschung. Pop-Sängerin Sinéad O'Connor wirft dem Vatikan vor, er wolle sich aus der Affäre ziehen*
- 2 *Die Erschütterung und die Scham des Papstes in seinem Brief an Irlands Katholiken sind echt. Doch er macht die Liberalisierung der Kirche für die Missbrauchsfälle verantwortlich. Das ist Unsinn. Zu den wahren Ursachen der Krise schweigt er leider.*
- 3 *Die schwarz-gelbe Koalition wackelt, die Wähler liebäugeln mit Rot-Grün: Jürgen Rüttgers kämpft in Nordrhein-Westfalen um seine Regierungsmehrheit - und seine politische Karriere. Um wieder in die Offensive zu kommen, geht der Ministerpräsident nun auf Distanz zur Krisenkoalition in Berlin.*
- 4 *Rückschlag im Titelrennen: Der FC Bayern hat durch zwei Gegentore in der Schlussphase bei Eintracht Frankfurt verloren - und könnte damit an diesem Spieltag die Tabellenführung verlieren. Werder Bremen gewann gegen den VfL Bochum, Freiburg setzte sich gegen Mainz durch.*
- 5 *Werder Bremen macht hinten Fehler und liefert vorne ein Spektakel, Bayern verliert in letzter Minute in Frankfurt und Leverkusen erlebt in Dortmund zwei völlig unterschiedliche Halbzeiten. Lange sah es in Frankfurt beim Gastspiel der Bayern nach einem typisch minimalistischen Münchner Sieg aus, doch dann drehte die Eintracht in einer verrückten Schlussphase das Spiel und gewann mit 2:1*
- 6 *Auf dem Landesparteitag in Münster versucht Ministerpräsident Rüttgers, die von der Affäre gebeutelte CDU in NRW mitzureißen - mit Hilfe der Kanzlerin und schrillen Warnungen vor der Opposition. CDU-Chefin Angela Merkel wirft am Samstag einiges in die Waagschale, um den nordrhein-westfälischen Parteifreunden Mut und Kampfkraft für den Landtagswahlkampf einzuhauchen.*

Euklidische Distanz

- **Euklidische Distanz** $d_E(\underline{a}, \underline{b})$ zwischen zwei n-dimensionalen Vektoren $\underline{a} = (a_1, \dots, a_n)$ und $\underline{b} = (b_1, \dots, b_n)$:

$$d_E(\underline{a}, \underline{b}) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$$

- Variante: **Quadrierte Euklidische Distanz**

$$d_{SE}(\underline{a}, \underline{b}) = \sum_{i=1}^n (a_i - b_i)^2$$

- **Anwendbar** für numerische und boolsche Daten
- **Ähnlichkeitsmaß**:

$$s_E(\underline{a}, \underline{b}) = \frac{1}{1 + d_E(\underline{a}, \underline{b})}$$

Cosinus

- **Cosinus Ähnlichkeit** $s_C(\underline{a}, \underline{b})$ zwischen zwei n-dimensionalen Vektoren $\underline{a} = (a_1, \dots, a_n)$ und $\underline{b} = (b_1, \dots, b_n)$:

$$s_C(\underline{a}, \underline{b}) = \frac{\underline{a} \cdot \underline{b}^T}{||\underline{a}|| \cdot ||\underline{b}||}$$

Skalarprodukt der beiden Vektoren geteilt durch die Länge der Vektoren. Cosinus nimmt Maximalwert von 1 an, wenn die Vektoren aufeinander liegen. Er ist = 0, wenn die Vektoren senkrecht zueinander sind.

- **Anwendbar** für numerische und boolsche Daten. Bevorzugtes Maß im Bereich des Text-Mining für die Bestimmung der Ähnlichkeit zwischen Dokumenten.
- Wenn alle Vektoren normiert sind ($||\underline{x}|| = 1$), ist die Cosinus-Ähnlichkeit gleich der Ähnlichkeit, die sich aus der euklidischen Distanz ergibt. In diesem Fall kann die Ähnlichkeit einfach durch das Skalarprodukt im Zähler berechnet werden.
- **Distanzmaß:**

$$d_C(\underline{a}, \underline{b}) = 1 - s_C(\underline{a}, \underline{b})$$

Pearson

- **Pearson Ähnlichkeit** $s_P(\underline{a}, \underline{b})$ zwischen zwei n-dimensionalen Vektoren $\underline{a} = (a_1, \dots, a_n)$ und $\underline{b} = (b_1, \dots, b_n)$:

$$s_P(\underline{a}, \underline{b}) = \frac{\underline{a}_d \cdot \underline{b}_d^T}{\|\underline{a}_d\| \cdot \|\underline{b}_d\|}$$

Dabei sind die sogenannten Differenzvektoren:

$$\underline{a}_d = (a_1 - \bar{a}, a_2 - \bar{a}, \dots, a_n - \bar{a})$$

$$\underline{b}_d = (b_1 - \bar{b}, b_2 - \bar{b}, \dots, b_n - \bar{b})$$

\bar{a} ist das arithmetische Mittel über die Komponenten in Vektor \underline{a} und \bar{b} ist das arithmetische Mittel über die Komponenten in Vektor \underline{b} .

- **Anwendbar** für numerische und boolsche Daten. Ist ein Maß für die lineare Korrelation zwischen zwei Vektoren. Falls die Mittelwerte \bar{a} und $\bar{b} = 0$ sind, ist die Pearson Korrelation gleich der Cosinus-Korrelation.
- **Distanzmaß**:

$$d_P(\underline{a}, \underline{b}) = 1 - s_P(\underline{a}, \underline{b})$$

Jaccard (Tanimoto)

Für die binären Vektoren \underline{a} und \underline{b} werden folgende Zählvariablen definiert:

- α ist die Anzahl der Stellen in denen beide Vektoren gleichzeitig gleich 1 sind,
- β ist die Anzahl der Stellen in denen Vektor \underline{b} gleich 1, und Vektor \underline{a} gleich 0 ist,
- γ ist die Anzahl der Stellen in denen Vektor \underline{b} gleich 0, und Vektor \underline{a} gleich 1 ist,
- δ ist die Anzahl der Stellen in denen beide Vektoren gleichzeitig gleich 0 sind.
- **Jaccard Ähnlichkeit** $s_J(\underline{a}, \underline{b})$ zwischen zwei n-dimensionalen Vektoren $\underline{a} = (a_1, \dots, a_n)$ und $\underline{b} = (b_1, \dots, b_n)$:

$$s_J(\underline{a}, \underline{b}) = \frac{\alpha}{\alpha + \beta + \gamma}$$

- **Anwendbar** für boolesche Daten, mit asymmetrischer Information (unbedeutende 0)
- **Distanzmaß:**

$$d_J(\underline{a}, \underline{b}) = 1 - s_J(\underline{a}, \underline{b})$$

Distanz- oder Ähnlichkeitsmatrix

- Ähnlichkeit (oder Unähnlichkeit) zwischen Instanzen wird in Ähnlichkeits- oder Distanzmatrizen dargestellt.
- Eintrag an Stelle (i, j) ist die Ähnlichkeit (Unähnlichkeit) zwischen den Instanzen i und j .

```

Bewertungsmatrix: Zeilen ->Kunden; Spalten -> Produkte -----
[[ 1.  1.  1.  1.  1.  1.  1.  0.]
 [ 1.  1.  1.  1.  1.  0.  0.  0.]
 [ 0.  0.  0.  0.  0.  0.  0.  1.]
 [ 0.  0.  0.  0.  0.  0.  1.  0.]]

-----Euklidische Distanz-----
[[ 0.          1.41421356  2.82842712  2.44948974]
 [ 1.41421356  0.          2.44948974  2.44948974]
 [ 2.82842712  2.44948974  0.          1.41421356]
 [ 2.44948974  2.44948974  1.41421356  0.          ]]

-----Cosinus Distanz-----
[[ 0.          0.15484575  1.          0.62203553]
 [ 0.15484575  0.          1.          1.          ]
 [ 1.          1.          0.          1.          ]
 [ 0.62203553  1.          1.          0.          ]]

-----Pearson Distanz-----
[[ 0.          0.51204996  2.          0.85714286]
 [ 0.51204996  0.          1.48795004  1.48795004]
 [ 2.          1.48795004  0.          1.14285714]
 [ 0.85714286  1.48795004  1.14285714  0.          ]]

-----Jaccard (Tanimoto) Distanz-----
[[ 0.          0.28571429  1.          0.85714286]
 [ 0.28571429  0.          1.          1.          ]
 [ 1.          1.          0.          1.          ]
 [ 0.85714286  1.          1.          0.          ]]

```

Standardisierung

- Was erwarten Sie, wenn die Ähnlichkeit zwischen den Kunden in Beispiel 1 mit z.B. der euklidischen Distanz bestimmt wird? Worin liegt das Problem?
- Bei der Ähnlichkeitsbestimmung fallen die Merkmale stärker ins Gewicht, deren Varianz größer ist. Dieser Effekt ist i.d.R. unerwünscht.
- **Abhilfe:** Standardisierung aller Merkmale in der Vorverarbeitung. D.h.
 - 1 von jedem Wert wird der Merkmalsmittelwert abgezogen,
 - 2 anschließend wird durch die Standardabweichung des Merkmals geteilt.
- Die so standardisierten Merkmale sind mittelwertfrei und haben eine Standardabweichung (und Varianz) von 1.

Standardisierung

- Wird mit $x_{i,k}$ das k.te Merkmal der i.ten Instanz bezeichnet, dann wird für die Standardisierung des k.ten Merkmals zunächst der Mittelwert dieses Merkmals über alle N Instanzen wie folgt berechnet:

$$\bar{x}_k = \frac{1}{N} \sum_{i=1}^N x_{i,k}$$

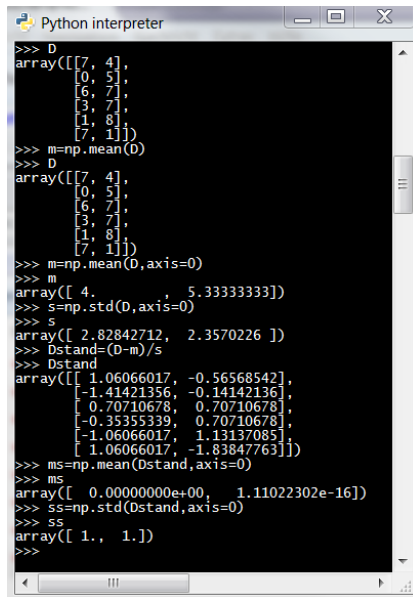
- Mit diesem Mittelwert berechnet sich die Varianz des k.ten Merkmals zu

$$\sigma_k^2 = \frac{1}{N-1} \sum_{i=1}^N (x_{i,k} - \bar{x}_k)^2$$

- Der standardisierte Wert des k.ten Merkmals in der i.ten Instanz ist dann

$$\tilde{x}_{i,k} = \frac{x_{i,k} - \bar{x}_k}{\sigma_k}$$

Standardisierung in Python/Numpy

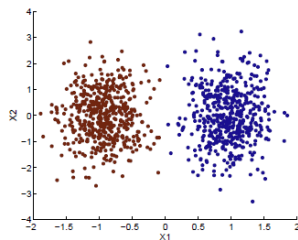
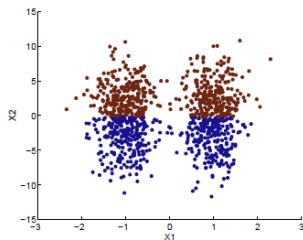


```

Python interpreter
>>> D
array([[7, 4],
       [0, 5],
       [6, 7],
       [3, 7],
       [1, 8],
       [7, 1]])
>>> m=np.mean(D)
>>> D
array([[7, 4],
       [0, 5],
       [6, 7],
       [3, 7],
       [1, 8],
       [7, 1]])
>>> m=np.mean(D,axis=0)
>>> m
array([ 4.          ,  5.33333333])
>>> s=np.std(D,axis=0)
>>> s
array([ 2.82842712,  2.3570226 ])
>>> Dstand=(D-m)/s
>>> Dstand
array([[ 1.06066017, -0.56568542],
       [-1.41421356, -0.14142136],
       [ 0.70710678,  0.70710678],
       [-0.35355339,  0.70710678],
       [-1.06066017,  1.13137085],
       [ 1.06066017, -1.83847763]])
>>> ms=np.mean(Dstand,axis=0)
>>> ms
array([ 0.00000000e+00,  1.11022302e-16])
>>> ss=np.std(Dstand,axis=0)
>>> ss
array([ 1.,  1.])
>>>

```

Clustering ohne (links) und mit (rechts) Standardisierung



Min-Max-Scaling

- Minimalwert im k.ten Merkmal

$$x_{k,min} = \min_i \{x_{i,k}\}$$

- Maximalwert im k.ten Merkmal

$$x_{k,max} = \max_i \{x_{i,k}\}$$

- Skalierter Wert:

$$\tilde{x}_{i,k} = \frac{x_{i,k} - x_{k,min}}{x_{k,max} - x_{k,min}}$$

- Alle skalierten Werte liegen im Bereich $[0, \dots, 1]$
- Problem: Verzerrung im Fall von Ausreißern

MinMaxScaling mit Python (Numpy)

```

IPython: C:\maucher\Documents
...: [7,1]).astype(float)

In [27]: D
Out[27]:
array([[ 7.,  4.],
       [ 0.,  5.],
       [ 6.,  7.],
       [ 3.,  7.],
       [ 1.,  8.],
       [ 7.,  1.]])

In [28]: mini=np.min(D,axis=0)

In [29]: mini
Out[29]: array([ 0.,  1.])

In [30]: maxi=np.max(D,axis=0)

In [31]: maxi
Out[31]: array([ 7.,  8.])

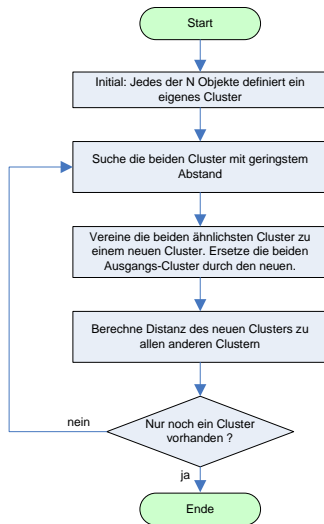
In [32]: Dscaled=(D-mini)/(maxi-mini)

In [33]: Dscaled
Out[33]:
array([[ 1.          ,  0.42857143],
       [ 0.          ,  0.57142857],
       [ 0.85714286,  0.85714286],
       [ 0.42857143,  0.85714286],
       [ 0.14285714,  1.          ],
       [ 1.          ,  0.          ]])

In [34]:

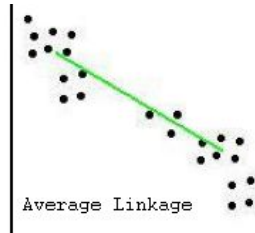
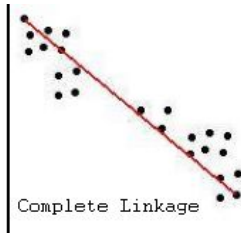
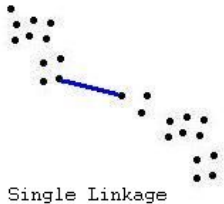
```

Algorithmus Hierarchisches Clustering



- Methoden zur Berechnung der Distanz zwischen Clustern
 - **Single Linkage**: Abstand zwischen Cluster i und j ist **minimaler Abstand** zwischen einer Instanz in Cluster i und einer Instanz in Cluster j .
 - **Complete Linkage**: Abstand zwischen Cluster i und j ist **maximaler Abstand** zwischen einer Instanz in Cluster i und einer Instanz in Cluster j .
 - **Average Linkage**: Berechne alle möglichen Abstände zwischen Instanzen in Cluster i und Instanzen in Cluster j und bestimme davon den Mittelwert.
- Hierarchisches Clustering wird meist in Form eines Dendrogramms dargestellt.

Abstandsmaße zwischen Clustern



- Beim Complete-Linkage werden bevorzugt **kompakte Cluster** zusammengefügt
- Beim Single-Linkage können bei der Clusterbildung **Ketteneffekte** entstehen

Abstandsmaße zwischen Clustern: Ward Linkage

- Der **Mean Square Error (MSE)** eines Clusters C_i ist:

$$MSE_i = \frac{1}{n_i} \sum_{\forall \mathbf{x}_p \in C_i} \|\mathbf{x}_p - \mathbf{m}_i\|^2$$

wobei

$$\mathbf{m}_i = \frac{1}{n_i} \sum_{\forall \mathbf{x}_p \in C_i} \mathbf{x}_p$$

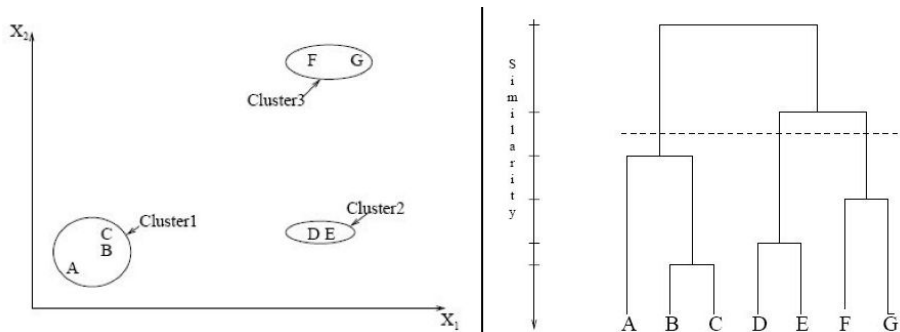
den Mittelpunkt von Cluster C_i mit n_i zugehörigen Instanzen beschreibt.

- Werden zwei Cluster C_i und C_j zu einem neuen Cluster C_* vereinigt, dann ist

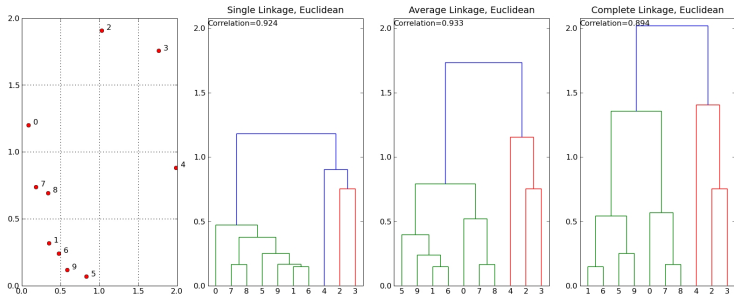
$$MSE_* \geq MSE_i + MSE_j$$

- Durch das iterative Vereinigen von Clustern im hierarchischen Clustering nimmt also der gesamte *MSE* über alle Cluster zu.
- Nach der **Ward-Linkage** Methode werden in jedem Schritt die beiden Cluster vereinigt, durch deren Vereinigung der MSE minimal zunimmt.

Dendrogramm



Dendrogramm



Anwendungsbeispiel Kundenclustering

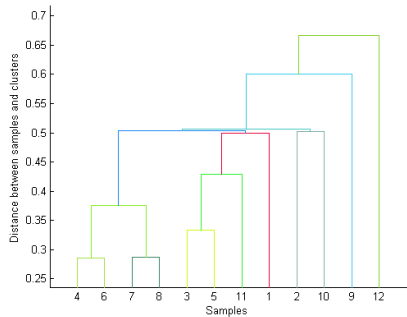
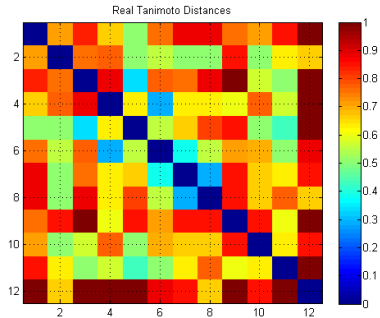
Kunden-Produkt-Matrix

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
K1	1	0	0	1	1	0	0	0	0	0
K2	1	1	1	1	0	0	0	0	1	1
K3	0	0	1	0	1	1	0	0	0	1
K4	1	0	0	1	0	1	1	1	0	0
K5	1	0	1	1	1	1	0	0	0	1
K6	1	1	1	1	0	1	1	1	0	0
K7	0	1	1	1	0	0	1	1	0	1
K8	0	1	0	1	0	0	1	1	1	1
K9	1	0	0	0	0	0	1	0	0	0
K10	1	0	1	0	1	0	0	1	1	1
K11	1	0	1	0	0	1	1	0	0	1
K12	0	1	0	0	0	0	0	0	1	0

Bildung der Cluster 13 - 23

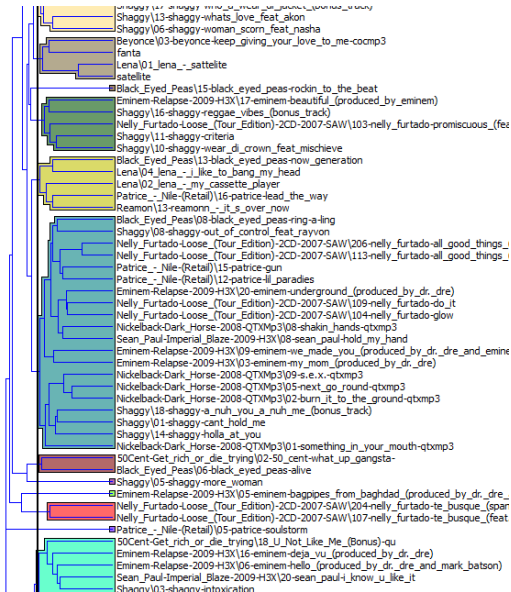
Cluster Id	Cluster I	Cluster J	Tanimoto Distanz
13	4	6	0.29
14	7	8	0.29
15	3	5	0.33
16	13	14	0.375
17	11	15	0.429
18	1	17	0.5
19	2	10	0.5
20	16	18	0.5
21	19	20	0.5
22	9	21	0.6
23	12	22	0.667

Anwendungsbeispiel Kundenclustering



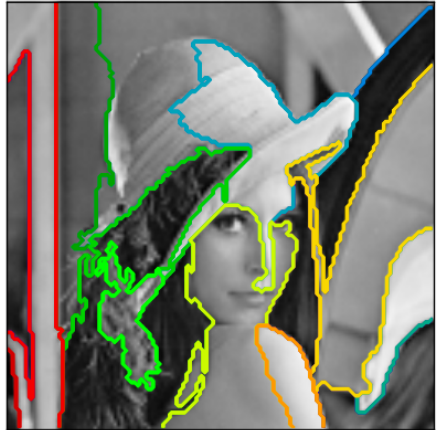
Anwendungsbeispiel: Ähnlichkeit von Musikfiles

- Musikfiles können durch eine Vielzahl von Audio-Merkmalen beschrieben werden.
- **Marsyas** (<http://marsyas.info/>) ist ein Music Information Retrieval System.
- **Marsyas** berechnet für jedes Musikfile die Werte bezüglich ca. 125 Audio-Merkmalen
- Die Merkmalsvektoren von ca. 1500 Musikfiles werden einem hierarchischen Clustering übergeben.



Anwendungsbeispiel: Bildsegmentierung

- Linkage Methode: *Ward + Connectivity-Constraint*
- Connectivity-Constraint: Füge nur benachbarte Cluster zusammen, also Cluster deren Pixel eine zusammenhängende Bildregion bilden,
- Dadurch wird auch die Komplexität deutlich reduziert



Qualität des Clustering

- **Allgemein:** Gewünschte Eigenschaft eines gefundenen Clusterings:
 - **Kleine Intra-Cluster Distanz:** Objekte im selben Cluster sollten nahe beieinander liegen
 - **Große Inter-Cluster Distanz:** Objekte in unterschiedlichen Clustern sollten große Distanz zueinander haben
- **Ähnlichkeit** zwischen Clustern und deren Elementen hängt stark von der verwendeten **Linkage-Methode** ab.
- Im Dendrogramm wird die Ähnlichkeit zwischen Clustern durch die Höhe auf der diese verbunden werden repräsentiert: Je höher der Verbindungslink umso unähnlicher die Cluster.
- **Cophenetische Distanz** zwischen zwei Objekten ist die Höhe, auf der beide im Dendrogramm verbunden werden.
- Wenn Cluster auf einer bestimmten Höhe verbunden werden, deren eigene Verbindungslinien weit unterhalb dieser Höhe liegen, dann werden damit zwei relativ ungleiche Cluster verbunden.

Qualität des Clustering

- **Problem:** Wie gut reflektiert, die cophenetische Distanz die tatsächliche Distanz zwischen Objekten?
- **Lösung: Berechne Cophenetische Korrelation¹:**

$$c = \frac{\sum_{i < j} (d_{i,j} - \bar{d}) \cdot (cd_{i,j} - \overline{cd})}{\sqrt{\sum_{i < j} (d_{i,j} - \bar{d})^2 \cdot \sum_{i < j} (cd_{i,j} - \overline{cd})^2}}$$

wobei $d_{i,j}$ die tatsächliche Distanz zwischen zwei Objekten, $cd_{i,j}$ die cophenetische Distanz zwischen zwei Objekten und mit \bar{d} und \overline{cd} die entsprechenden Mittelwerte bezeichnet werden.

- Liegt der Korrelationskoeffizient c nahe bei 1, dann ist die vom Clustering gefundene Ähnlichkeit eine gute Repräsentation der tatsächlichen Ähnlichkeit.

¹Entspricht Pearson Korrelation

Qualität des Clustering: Silhouette Score

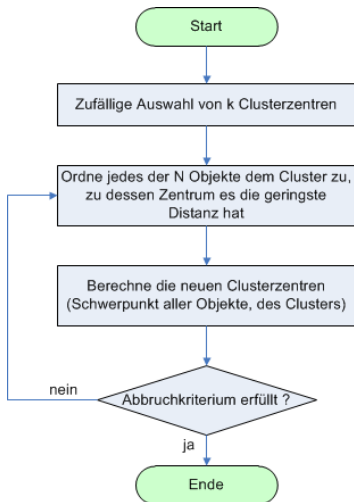
- Silhouette Score ist **mittlerer Silhouette Coefficient**
- Silhouette Coefficient eines einzelnen Vektors \mathbf{x}

$$sc(\mathbf{x}) = \frac{(b_x - a_x)}{\max(a_x, b_x)}$$

mit

- a_x ist die mittlere Distanz von \mathbf{x} zu allen anderen Vektoren im gleichen Cluster
- b_x ist die mittlere Distanz von \mathbf{x} zu allen Vektoren im nächst liegenden Cluster.

K-Means Clustering Algorithmus



- Bestimmung des Schwerpunktes (x_S, y_S) von n Punkten (x_i, y_i) im 2-dimensionalen euklidischen Raum

$$x_S = \frac{1}{n} \sum_{i=1}^n x_i$$

$$y_S = \frac{1}{n} \sum_{i=1}^n y_i$$

- Mögliche Abbruchkriterien
 - Definiere Wert für ϵ und breche ab, sobald die Summe aller Schwerpunktverschiebungen im Vergleich zur letzten Iteration kleiner als ϵ ist.
 - Vorgegebene Anzahl von Iterationen

Leistungskriterium: Rekonstruktionsfehler

- Für eine Eingabe $\mathbf{x}_p \in T$ ist der zugehörige Clustermittelpunkt \mathbf{m}_i , wenn gilt:

$$\|\mathbf{x}_p - \mathbf{m}_i\| = \min_{j \in 1 \dots K} \|\mathbf{x}_p - \mathbf{m}_j\|$$

wobei mit K die voreingestellte Clusteranzahl, und mit $\|x - m\|$ die Distanz zwischen x und m hinsichtlich der verwendeten Metrik bezeichnet wird.

- Wird mit C_i das Cluster mit Mittelpunkt \mathbf{m}_i bezeichnet und ist n_i die Anzahl der Vektoren im Cluster C_i , dann berechnet sich der Clustermittelpunkt zu

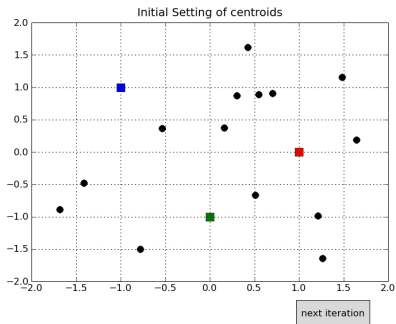
$$\mathbf{m}_i = \frac{1}{n_i} \sum_{\forall \mathbf{x}_p \in C_i} \mathbf{x}_p$$

- Rekonstruktionsfehler des ermittelten Clusterings bezüglich der Trainingsmenge T ist die Summe über alle Distanzen zwischen den Eingabevektoren und ihren Clustermittelpunkten.

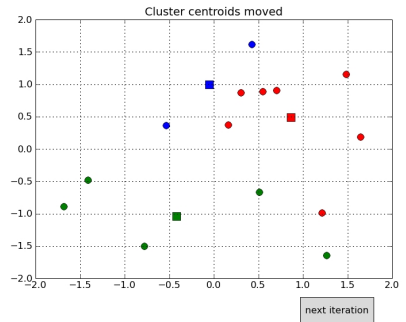
$$E = \sum_{i=1}^K \sum_{\forall \mathbf{x}_p \in C_i} \|\mathbf{x}_p - \mathbf{m}_i\|$$

Beispiel k-means Clustering von 14 Objekten (1)

Initiale Konfiguration

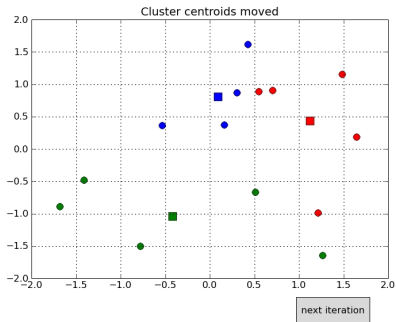


Iteration 1

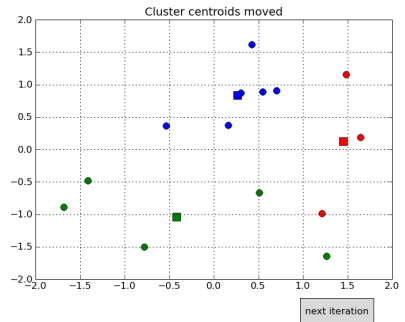


Beispiel k-means Clustering von 14 Objekten (2)

Iteration 2

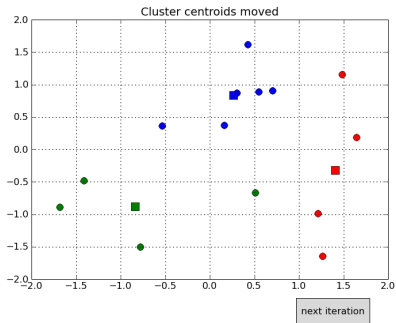


Iteration 3

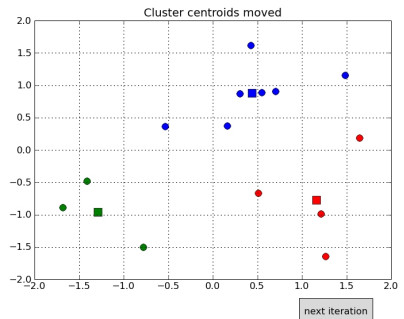


Beispiel k-means Clustering von 14 Objekten (3)

Iteration 4



Iteration 5 (Stabiler Endzustand)



Anwendung: Quantisierung

- Siehe Anwendungen Clustering allgemein
- **Speziell: Ungleichmäßige Quantisierung**, z.B.
 - Farbquantisierung zur Farbraumreduktion
 - Vektorquantisierung: in einer Vielzahl moderner Codecs eingesetzt
- Ziel der Quantisierung allgemein: **Bilde ähnliche Werte (Vektoren, Symbole, Objekte) auf einen gemeinsamen Repräsentanten ab, so dass bei der Ersetzung der Werte durch ihren Repräsentanten eine möglichst geringe Verzerrung zwischen Original und Abbild entsteht.**
- **k-means** Algorithmus berechnet ein **Codebuch**, welches den Eingabevektoren ihre Repräsentanten (=Referenzvektoren) zuordnet.
- Die Repräsentanten sind die Mittelpunkte der Cluster, zu dem der jeweilige Eingabevektor gehört.
- Der k-means Algorithmus gewährleistet eine Minimierung (lokales Minimum!) des Rekonstruktionsfehlers. Häufig vorkommende Wertbereiche werden fein, selten vorkommende Bereiche grob quantisiert.

Beispiel Farbraumtransformation von 8 nach 5 Bit/Kanal

Original: 8 Bit / Farbkanal

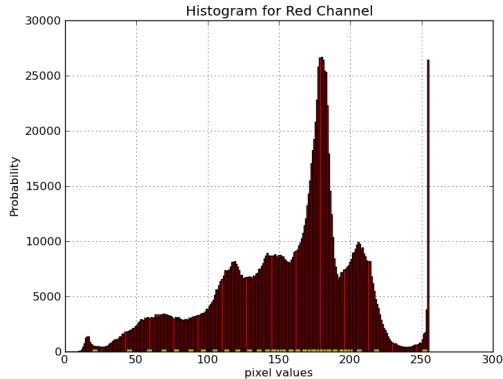


Quantisiert: 5 Bit / Farbkanal



Häufigkeitsverteilung und Quantisierung

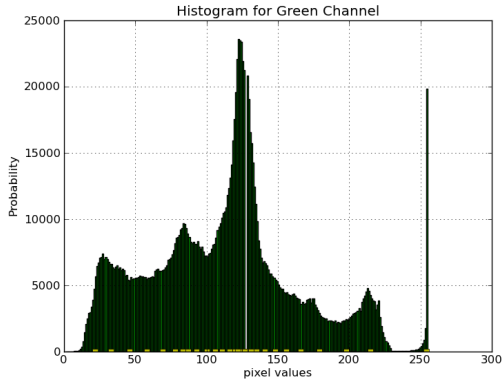
Häufigkeitsverteilung Rot



- Anzahl der Cluster: $K = 2^5 = 32$ durch die Anzahl der darstellbaren Farben im reduzierten Farbraum gegeben
- Clustermittelpunkte: Gelbe Punkte auf der x-Achse in den Schaubildern

Häufigkeitsverteilung und Quantisierung

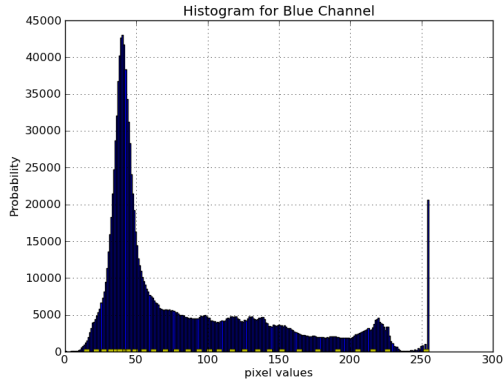
Häufigkeitsverteilung Grün



- Anzahl der Cluster: $K = 2^5 = 32$ durch die Anzahl der darstellbaren Farben im reduzierten Farbraum gegeben
- Clustermittelpunkte: Gelbe Punkte auf der x-Achse in den Schaubildern

Häufigkeitsverteilung und Quantisierung

Häufigkeitsverteilung Blau



- Anzahl der Cluster: $K = 2^5 = 32$ durch die Anzahl der darstellbaren Farben im reduzierten Farbraum gegeben
- Clustermittelpunkte: Gelbe Punkte auf der x-Achse in den Schaubildern

K-Means: Vor- und Nachteile

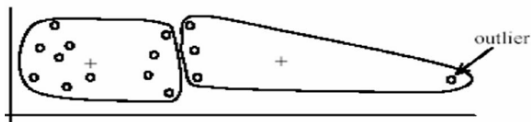
Vorteile

- Einfach und schnell
- Konvergiert in lokalem Minimum des Rekonstruktionsfehlers

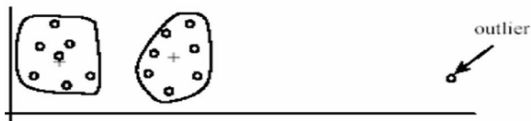
Nachteile

- Wie ist k zu wählen?
- Resultat hängt stark von initialen Clustermittelpunkten ab
- Keine Berücksichtigung von Ausreißern
- Liefert keine Aussage über Wahrscheinlichkeit der Clusterzugehörigkeit
- Keine Berücksichtigung von unterschiedlichen Clusterformen

K-Means Nachteile: Keine Berücksichtigung von Ausreißern



(A): Undesirable clusters



(B): Ideal clusters

DBSCAN: Idee und Eigenschaften

Idee:

- *Density-Based-Spatial-Clustering of Applications with Noise*
- Cluster werden gebildet durch zusammenhängende Gebiete mit **großer Sample-Dichte**
- Clustergrenzen sind Gebiete **geringer Sample-Dichte**
- Einzelne Samples, die ausserhalb von Bereichen großer Sample-Dichte liegen, werden als Rauschen interpretiert und **keinem Cluster zugeordnet**.

Eigenschaften:

- Anzahl der Cluster muss **nicht** vorgegeben werden.
- Clusterform beliebig (vgl. K-Means liefert immer konvexe Clusterformen)
- Skaliert gut auch für sehr große Datenmengen
- Kann für beliebige Distanz- und Ähnlichkeitsmaße angewandt werden.

Unterscheidung von Cluster und Clustergrenze

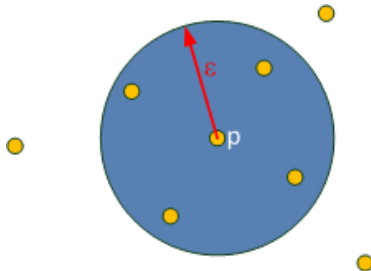
- **Problem:** Wie werden Gebiete großer Sample Dichte (Cluster), wie Gebiete kleiner Sample-Dichte (Clustergrenzen) definiert?
- **Antwort:** Durch die Parameter ϵ -Umgebung und *MinPoints*.

ϵ -Umgebung

ϵ -Umgebung

Die Epsilonumgebung $N_\epsilon(p)$ eines Samples p ist die Menge aller Samples q , die in einem Abstand $d(p, q) < \epsilon$ um p liegen.

$$N_\epsilon(p) = \{q \in T \mid d(p, q) \leq \epsilon\}.$$



MinPoints und Punkttypen

Abhängig von der Wahl der Parameter ϵ und *MinPoints* werden folgende Punkttypen unterschieden:

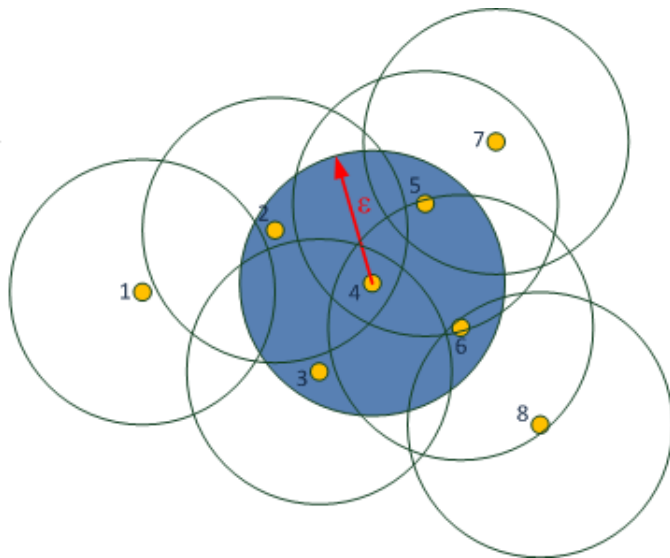
Kernpunkt: Ein Punkt (=Sample) mit mindestens *MinPoints* anderen Samples in seiner ϵ -Umgebung

Randpunkt: Ein Punkt der selbst kein Kernpunkt ist, aber in der ϵ -Umgebung eines Kernpunktes liegt.

Rauschpunkt: Ist weder Kern- noch Randpunkt und gehört zu keinem Cluster.

Beispiel: Punkttypen

MinPoints=3:
Kernpunkte ?
Randpunkte?
Rauschpunkte?



DBSCAN Algorithm Pseudocode

```
DBSCAN(D, eps, MinPts)
  C = 0
  for each unvisited point P in dataset D
    mark P as visited
    NeighborPts = regionQuery(P, eps)
    if sizeof(NeighborPts) < MinPts
      mark P as NOISE
    else
      C = next cluster
      expandCluster(P, NeighborPts, C, eps, MinPts)

expandCluster(P, NeighborPts, C, eps, MinPts)
  add P to cluster C
  for each point P' in NeighborPts
    if P' is not visited
      mark P' as visited
      NeighborPts' = regionQuery(P', eps)
      if sizeof(NeighborPts') >= MinPts
        NeighborPts = NeighborPts joined with NeighborPts'
  if P' is not yet member of any cluster
    add P' to cluster C

regionQuery(P, eps)
  return all points within P's eps-neighborhood (including P)
```

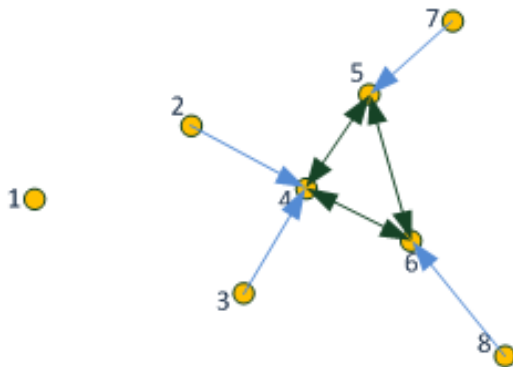
Source: <http://en.wikipedia.org/wiki/DBSCAN>

Beispiel: Gefundenes Cluster

Cluster enthält

- Kernpunkte 4,5,6
- Randpunkte 2,3,7,8

Punkt 1 ist Rauschpunkt



Übung DBSCAN

Gegeben ist die folgende Distanzmatrix. Wie werden die 10 Instanzen mit dem DBSCAN ($\epsilon = 0.7$, $MinPoints = 3$) in Cluster eingeteilt? Welche Instanzen sind Kernpunkte, Randpunkte, Ausreisser?

[0.	0.41	0.98	3.3	2.59	1.2	3.29	2.68	0.81	3.13]
[0.41	0.	0.65	3.11	2.43	0.8	3.1	2.41	0.53	2.88]
[0.98	0.65	0.	3.37	2.77	0.41	3.38	2.58	0.21	3.05]
[3.3	3.11	3.37	0.	0.74	3.04	0.08	0.88	3.44	0.54]
[2.59	2.43	2.77	0.74	0.	2.48	0.71	0.74	2.81	0.85]
[1.2	0.8	0.41	3.04	2.48	0.	3.05	2.22	0.59	2.69]
[3.29	3.1	3.38	0.08	0.71	3.05	0.	0.92	3.44	0.61]
[2.68	2.41	2.58	0.88	0.74	2.22	0.92	0.	2.67	0.47]
[0.81	0.53	0.21	3.44	2.81	0.59	3.44	2.67	0.	3.15]
[3.13	2.88	3.05	0.54	0.85	2.69	0.61	0.47	3.15	0.]]

Ziel und Anwendungen Association Rule Mining

Ziel: Auffinden von Mustern, welche Beziehungen über das gemeinsame Auftreten von Merkmalswerten beschreiben.

Annahme: Alle Merkmale sind nominell (nicht anwendbar für numerische Merkmale)

Anwendungen:

- Warenkorbanalyse, z.B. über Daten die über Kundenkarten erhoben werden.
- Auffinden von Mustern in großen Datenmengen (Mustererkennung), z.B. Kraftstoffdatenanalyse
- Web Mining: Erkennung von Usage Profilen, wie z.B. *Nutzer, welche die Seite A besucht haben besuchen mit einer Wahrscheinlichkeit von 78 % innerhalb einer Woche auch Seite B.*
- Fraud Detection: Betrug kann häufig über bestimmte Verhaltensmuster erkannt werden.

Beispiel für Assoziationen zwischen Attributen

Relation: NotebookDecTree2010_04_30.txt

No.	Title	CPU Speed	CPU Type	ProcCount	System	System	Display	Rate	Price	Kat
	Nominal	Numeric	Nominal	Numeric	Numeric	Nominal	Numeric	Numeric	Numeric	Nominal
1	Toshiba Mini NB205-A410EL...	1.66	Intel Atom N450	1.0	1000.0	SDRAM	10.1	4.0	399.99	low
2	Toshiba Mini 300 Series NB3...	1.66	Intel Atom N450	1.0	1000.0	SDRAM	10.1	4.0	349.99	low
3	Apple MacBook Pro MC371L...	2.4	Intel Core Duo	2.0	4000.0	SDRAM	15.4	5.0	1799.0	high
4	Apple MacBook MC070LLA...	2.26	Intel Core 2 Duo	1.0	2000.0	SDRAM	13.3	4.5	999.0	med
5	HP G60-630US 15.6-Inch La...	2.2	Pentium	2.0	3000.0	PC 1100 DDR Memory	15.6	4.0	836.0	med
6	Apple MacBook Pro MC373L...	2.66	Intel Core Duo	2.0	4000.0	SDRAM	15.4	4.5	2199.0	high
7	ASUS UL30A-A1 Thin and Li...	1.3	Intel Core 2 Duo	2.0	4000.0	DDR2 SDRAM	13.3	4.5	999.0	med
8	Toshiba Satellite U505-E590...	2.2	Intel Pentium Dual-Core	1.0	3000.0	SDRAM	15.6	4.5	579.99	med
9	Toshiba Mini NB205-A410ER...	1.66	Intel Atom N450	1.0	1000.0	SDRAM	10.1	4.0	399.99	low
10	ASUS UL20A-A1 Thin and Li...	1.3	Intel Core 2 Duo	2.0	2000.0	SDRAM	12.1	4.5	599.0	med
11	Gateway N7922u 17.3-Inch...	2.26	Intel Core i5	2.0	4000.0	SDRAM	17.3	4.0	679.99	med
12	ASUS U30C-A1 13.3-Inch...	2.26	Intel Core Duo	2.0	4000.0	SDRAM	13.3	5.0	899.0	med
13	Toshiba Satellite T115D-S11...	1.5	AMD Athlon	2.0	2000.0	SDRAM	11.6	4.5	449.99	low
14	HP TouchSmart TM5-1070LA...	1.3	Intel Core 2 Duo	2.0	4000.0	SDRAM	12.1	3.0	1250.0	high
15	Toshiba Satellite T115D-S11...	1.6	AMD Athlon	2.0	2000.0	SDRAM	11.6	4.5	449.99	low
16	Toshiba Satellite U5850-570...	2.3	AMD Turion 64 X2	1.0	2000.0	SDRAM	17.3	4.5	649.99	med
17	ASUS Republic of Gamers G...	2.8	Intel Core i7	4.0	6000.0	SDRAM	15.6	3.5	1499.0	high
18	Apple MacBook Pro MC118L...	2.53	Intel Core Duo	2.0	4000.0	DDR2 SDRAM	15.4	4.5	1599.0	high
19	Toshiba Mini NB205-A410W...	1.66	Intel Atom N450	1.0	1000.0	SDRAM	10.1	4.0	399.99	low
20	Acer Aspire Timeline AS161...	1.3	Intel Core 2 Duo	1.0	4000.0	SDRAM	11.1	4.5	599.99	med
21	Acer AS5732Z-4857 15.6-I...	2.2	Intel Pentium Dual-Core	1.0	3000.0	SDRAM	15.6	4.0	449.99	med
22	ASUS UL80V-A1 14-Inch T...	1.3	Intel Core 2 Duo	2.0	4000.0	SDRAM	14.0	4.0	849.0	med
23	Sony VAIO VPC-CW21FX/L...	2.13	Intel Core i3	1.0	4096.0	DRAM	14.0	4.0	849.99	med
24	HP Pavilion Dv4-2160US 14...	2.26	Intel Core i5	2.0	4000.0	SDRAM	14.1	4.0	1035.0	med
25	HP Pavilion Dv6-2170US 15...	2.26	Intel Core i5	2.0	4000.0	SDRAM	15.6	4.0	1137.0	med
26	HP Pavilion Dv5-1130US 13...	1.6	AMD Athlon	2.0	4000.0	SDRAM	13.3	4.5	549.0	med
27	HP Pavilion Dv4-2160US 15...	1.6	Intel Core i7	4.0	4000.0	SDRAM	15.6	4.5	1312.0	high
28	ASUS N61JQ-A1 16-Inch Ve...	2.8	Intel Core i7	2.0	4000.0	DDR2 SDRAM	16.0	4.0	1199.0	med
29	Toshiba Mini NB205-A325W...	1.66	Intel Atom N280	1.0	1000.0	SDRAM	10.1	4.0	399.99	low
30	HP Pavilion Dv7-2180US 17...	1.6	Intel Core i7	1.0	6000.0	SDRAM	17.3	3.5	1792.0	high
31	ASUS UL30A-A2 Thin and Li...	1.3	Intel Core 2 Duo	2.0	4000.0	SDRAM	13.3	4.5	799.0	med
32	Toshiba Qosmio X505-Q660...	2.26	Intel Core i5	4.0	4000.0	SDRAM	18.4	3.5	1199.99	med
33	ASUS UL80V-A2 14-Inch T...	1.3	Intel Core 2 Duo	2.0	4000.0	DDR2 SDRAM	14.0	4.0	849.0	med
34	HP Pavilion Dv7-3160US 17...	2.5	AMD Turion 64 X2	2.0	4000.0	SDRAM	17.3	3.5	1100.0	med
35	Gateway N7922u 17.3-Inch...	2.26	Intel Core i5	2.0	4000.0	SDRAM	17.3	4.0	679.99	med
36	Toshiba Mini NB205-A325PK...	1.66	Intel Atom N280	1.0	1000.0	SDRAM	10.1	4.0	399.99	low
37	Apple MacBook Pro MB985L...	2.66	Intel Core Duo	2.0	4000.0	DDR2 SDRAM	15.4	4.5	1799.0	high
38	Disney Netpal by ASUS - 8...	1.6	Intel Atom N270	1.0	2000.0	SDRAM	8.9	4.0	349.99	low
39	Toshiba Qosmio X505-Q660...	2.8	Intel Core i7	4.0	6000.0	SDRAM	18.4	4.0	1899.99	high
40	Acer Aspire AS1410-3697 T...	1.3	Intel Celeron	1.0	2000.0	SDRAM	11.6	4.5	399.99	low
41	Apple MacBook Air MC234L...	2.13	Intel Core Duo	2.0	2000.0	DDR2 SDRAM	13.3	4.5	1799.0	high
42	HP Pavilion Dv6-2160US 15...	2.13	Intel Core i3	2.0	4000.0	SDRAM	15.6	4.5	1007.0	med
43	Acer Aspire AS7740-6656 U...	2.26	Intel Core i5	2.0	4000.0	SDRAM	17.3	4.0	699.99	med
44	HP G71-340US 17.3-Inch B...	2.2	Intel Core 2 Duo	2.0	4000.0	SDRAM	17.3	4.0	779.99	med
45	Acer Aspire Timeline AS581...	1.3	Intel Core Solo	1.0	3000.0	SDRAM	15.6	4.0	599.99	med
46	Acer Aspire Timeline AS481...	1.3	Intel Pentium Dual-Core	1.0	4000.0	SDRAM	14.0	4.5	449.99	med
47	Sony VAIO VPC-CW21FX/P...	2.13	Intel Core i3	1.0	4096.0	DRAM	14.0	4.0	849.99	med
48	HP Pavilion Dv4-2140US 14...	2.3	AMD Turion 64 X2	2.0	4000.0	SDRAM	14.1	3.0	927.0	med
49	Toshiba Mini NB205-A325ER...	1.66	Intel Atom N280	1.0	1000.0	SDRAM	10.1	4.0	399.99	low
50	Sony VAIO VPC-CW21FX/W...	2.13	Intel Core i3	1.0	4096.0	DRAM	14.0	4.0	849.99	med
51	ASUS Republic of Gamers G...	1.6	Intel Core i7	4.0	8000.0	SDRAM	17.3	4.0	1699.0	high
52	Toshiba Satellite A555-S609...	2.13	Intel Core i3	2.0	4000.0	SDRAM	13.3	4.0	799.99	med
53	Toshiba Satellite U505-S509...	2.13	Intel Core i3	1.0	4000.0	SDRAM	13.3	4.5	799.99	med
54	Apple MacBook Pro MC226L...	2.8	Intel Core Duo	2.0	4000.0	DDR2 SDRAM	17.0	5.0	2199.0	high
55	ASUS UL50V-A1 Thin and Li...	1.3	Intel Core 2 Duo	2.0	4000.0	SDRAM	15.6	4.0	849.0	med
56	Toshiba Mini NB205-A325SL...	1.66	Intel Atom N280	1.0	1000.0	SDRAM	10.1	4.0	399.99	low
57	Sony VAIO VGN-NW310P/F...	2.1	PowerPC G4	1.0	2000.0	SDRAM	15.5	4.5	679.99	med
58	Sony VAIO VGN-NW310P/B...	2.1	Intel Core 2 Duo	1.0	2000.0	SDRAM	15.5	4.5	679.99	med

Gefundene Regeln:

- Wenn *System* = 1000 dann *ProcCount* = 1
- Wenn *Display* = 10.1 dann *ProcCount* = 1
- Wenn *CPU Speed* = 1.66 dann *Display* = 10.1
- ...

Einführungsbeispiel

Gegeben: Kundendatenbank:

Kunde	MacBook	iPhone	Computerlexikon
1	1	1	1
2	0	1	1
3	1	1	0
4	0	1	0
5	0	1	0
6	1	0	1
7	1	0	1
8	1	0	1
9	1	0	1
10	1	0	0

Gesucht: Muster (Regeln) welche typisches Kaufverhalten der Kunden beschreiben.

Items und Transaktionen

- **Item i** ist ein Merkmals-Wert-Paar $M_j = val_{i,j}$ wobei $val_{i,j}$ ein Element aus dem Wertebereich W_j von Merkmal M_j ist, z.B. $MacBook = 1$.
- **Itemmenge I** : Menge aller möglichen Merkmals-Wert-Paare. Im Einführungsbeispiel enthält I insgesamt $2 \cdot 3 = 6$ Items.
- **Transaktion t** : Eine beliebige Untermenge von I : $t = \{i_1, i_2, \dots, i_L\} \subseteq I$
- **Transaktionsmenge T** : Die Menge aller Transaktionen
- Falls alle Merkmale einen Booleschen Wertebereich haben, werden in einer Transaktion nur die Items $M_j = val_{i,j}$ mit Wert $val_{i,j} = 1$ aufgenommen und diese nur mit dem Merkmalsname bezeichnet. Also, z.B.

$$t_3 = \{MacBook, iPhone\}$$

anstelle von

$$t_3 = \{MacBook = 1, iPhone = 1, Computerlexikon = 0\}$$

Anwendungsspezifische Bedeutung der Definitionen

- Warenkorbanalyse:
 - Itemset I ist die Menge aller angebotenen Waren.
 - Item i_j ist ein Produkt des gesamten Angebots.
 - Transaktion t_i ist die Menge der von einem Kunden gekauften Produkte
 - Transaktionsmenge T sind die Transaktionen aller Kunden
- Dokumentanalyse
 - Itemset I ist die Menge aller Worte, die in den gegebenen Dokumenten vorkommen.
 - Item i_j ist ein Wort aus I .
 - Transaktion t_i ist die Menge aller in einem Dokument vorkommenden Worte (Bag of Words)
 - Transaktionsmenge T enthält alle Dokumente

Definition Association Rule

- Eine **Assoziationsregel** ist eine Implikation der Form

$$X \rightarrow Y$$

mit den Eigenschaften $X, Y \subset I$, und $X \cap Y = \emptyset$

- **Support** einer Assoziationsregel:

$$\text{sup}(X \rightarrow Y) = P(X \wedge Y) = \frac{|X \wedge Y|}{|T|}$$

dabei wird mit $|X \wedge Y|$ die Anzahl der Transaktionen in T , in denen sowohl X als auch Y enthalten sind, und mit $|T|$ die Anzahl aller Transaktionen in T bezeichnet.

- **Confidence** einer Assoziationsregel:

$$\text{conf}(X \rightarrow Y) = \frac{|X \wedge Y|}{|X|}$$

dabei wird mit $|X|$ die Anzahl der Transaktionen in denen X enthalten ist bezeichnet.

- Eine Assoziationsregel beschreibt mit welcher Wahrscheinlichkeit (**confidence**) aus dem Vorhandensein von X auf das gleichzeitige Vorhandensein von Y geschlossen werden kann.

Anforderungen an den Algorithmus

- Nutzer gibt vor: Finde alle Assoziationsregeln mit einem Support von mindestens sup_{min} und einer Confidence von mindestens $conf_{min}$
- Anforderungen an den Algorithmus:
 - Vollständigkeit
 - Komplexität
- Es existieren viele Algorithmen für die Bestimmung der Assoziationsregeln. Alle liefern das gleiche Resultat, jedoch differieren Sie in Geschwindigkeit und Speicheranforderung.
- Am bekanntesten: Apriori Algorithmus

Apriori Algorithmus: Idee

2 Schritte:

- 1 Finde alle Itemsets, deren Support größer ist als der vorgegebene Minimalwert sup_{min} . Diese Itemsets sind die **Frequent Itemsets**
- 2 Suche in den gefundenen Frequent Itemsets die Assoziationsregeln, deren Confidence größer als der Minimalwert $conf_{min}$ ist.

Apriori Algorithmus: Schritt 1: Bestimme Frequent Itemsets

Die Idee:

- Alle Untermengen eines Frequent Itemset sind auch Frequent Itemsets
 - Gleichbedeutend: Wenn ein Itemset X nicht zur Menge der Frequent Itemsets gehört, dann sind auch alle Mengen welche X enthalten keine Frequent Itemsets.
- 1 Setze $k = 1$
 - 2 Zunächst wird die Menge C_k erzeugt, die alle Itemsets mit nur einem Element enthält.
 - 3 Dann wird die Teilmenge $F_k \subseteq C_k$ der Itemsets aus C_k bestimmt, deren Support größer als sup_{min} ist.
 - 4 Setze $k = k + 1$. Erzeuge die Menge C_k aller Itemsets mit k Elementen. Entferne aus C_k alle Itemsets, die ein Element enthalten, das nicht in F_{k-1} enthalten ist.
 - 5 F_k wird berechnet indem aus dem zuvor berechneten C_k , die Itemsets gestrichen werden, deren Support kleiner als sup_{min} ist.
 - 6 Wiederhole ab Schritt 4 solange bis ein leeres F_k entsteht.

Apriori Algorithmus: Schritt 2: Bestimme Association Rules

Für alle gefundenen Frequent Itemsets X :

① Für alle nicht-leeren Untermengen A von X :

① Setze $B = X - A$

② Füge die Assoziationsregel $A \rightarrow B$ der Menge aller Regeln hinzu, falls

$$\text{conf}(A \rightarrow B) = \frac{\text{sup}(X)}{\text{sup}(A)} \geq \text{conf}_{\min}$$

Beispiel: Bestimmung der Assoziationsregeln mit dem Apriori Algorithmus

- Bestimmen Sie für das Einführungsbeispiel (Kunden die Macbook, Iphone oder Computerlexikon gekauft haben) unter Anwendung des Apriori-Algorithmus alle Assoziationsregeln mit minimalen Support 0.3 und minimaler Confidence 0.8

Referenzen

- [EA] Ethem Alpaydin Maschinelles Lernen; deutschsprachige Ausgabe erschienen im Oldenbourg Verlag, München 2008
- [TM] Tom Mitchell Machine Learning; McGraw Hill International Editions, 1997
- [RN] S. Russell, P. Norvig Künstliche Intelligenz; Ein moderner Ansatz Pearson Studium 2004 (Deutsche Ausgabe)
- [ERTL] W. Ertel Grundkurs Künstliche Intelligenz; Eine praxisorientierte Einführung Vieweg Verlag 2008
- [KOH] T. Kohonen Self-organized formation of topologically correct feature maps Biological Cybernetics, Vol. 43, pp. 59-69, 1982
- [Jain] A.K. Jain, M.N. Murty, P.J. Flynn Data Clustering: A Review ACM Computing Surveys, Vol. 31, No. 3, September 1999