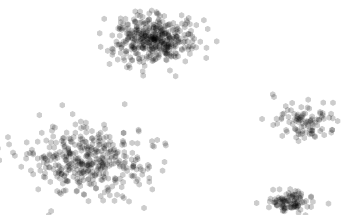
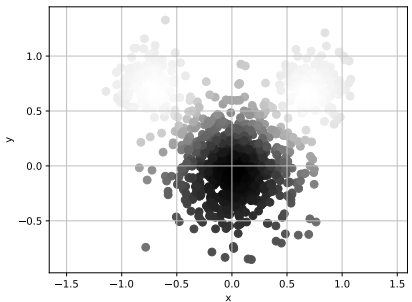
Einstieg in Clustering Algorithmen und der k-Means Algorithmus

Prof. Dr. Jörg Frochte

Maschinelles Lernen





Clustering-Verfahren

- Clustering-Verfahren sind Verfahren für unüberwachtes Lernen.

Beispiel Clustering vs. Klassifikation

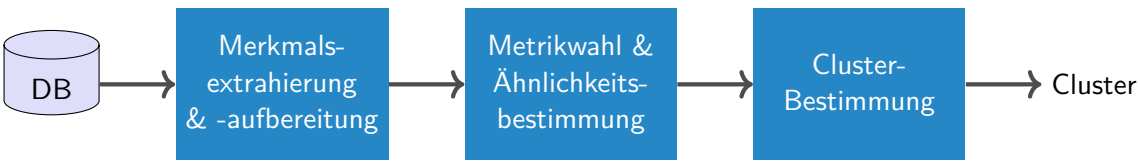
Bei der Klassifikation haben wir gelabelte Daten vorliegen und wissen z. B. um welche Art von Schwertlilien es sich handelt. Dieses Wissen – die Labels – haben wir bei einem unüberwachten Verfahren nicht. Die Aufgabe unseres Algorithmus ist es die Pflanzen danach zu gruppieren, welche Einträge sich am ähnlichsten sind.

Clustering ist somit die ...

... Einteilung einer Menge von Objekten in Gruppen, wobei wir folgende Ziele verfolgen:

- Wir wollen die Ähnlichkeit innerhalb der Gruppen maximieren.
- Wir wollen die Unterschiede zwischen den Gruppen maximieren, bzw. ihre Ähnlichkeit minimieren.

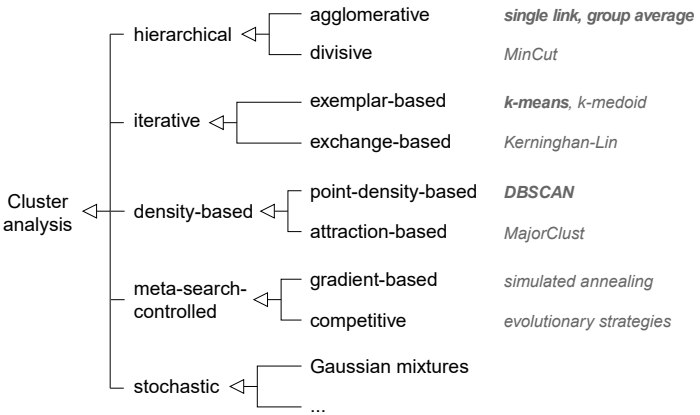
Grundablauf einer Clusteranalyse



- Merkmalsaufbereitung ist hier komplexer als bei einer Klassifikation, da man Skalenunterschiede zwischen Merkmalen vermeiden, jedoch innerhalb von Merkmalen Unterschiede nicht verwischen möchte.
- In der Praxis liegt man allerdings mit einer Standardisierung der Daten oft richtig. Man sollte jedoch, wenn es nicht gut funktioniert, auch andere Ansätze testen.
- Ziel eines Cluster-Algorithmus ist es, die Elemente x_j des Datenbestandes in Cluster C_i aufzuteilen.

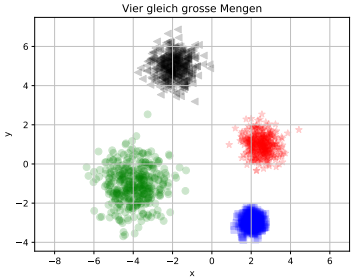
Taxonomie von Clusteralgorithmen

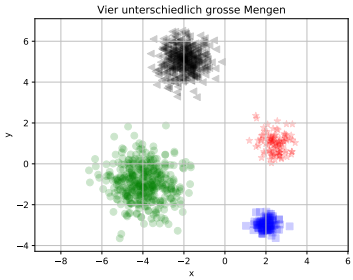
- Die Abbildung zeigt eine von Prof. Benno Stein vorgenommene Einteilungen von Cluster-Verfahren.
- Wir werden aus den drei populärsten Gruppen, den hierarchischen, den iterativen und den dichtebasierten Verfahren, jeweils einen Clusteralgorithmus vorstellen.

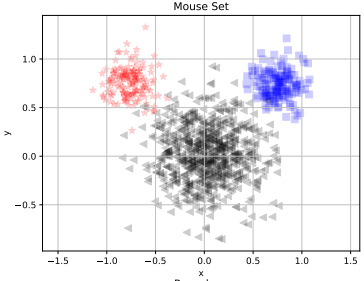


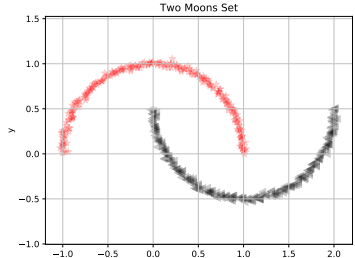
- Dabei konzentrieren wir uns wieder jeweils auf recht populäre Verfahren wie den DBSCAN und k-Means.

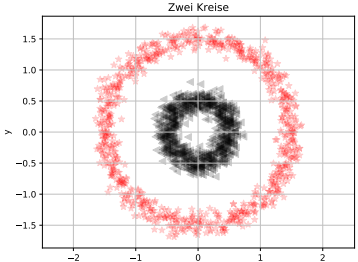
Testbeispiele für die Clusteralgorithmen

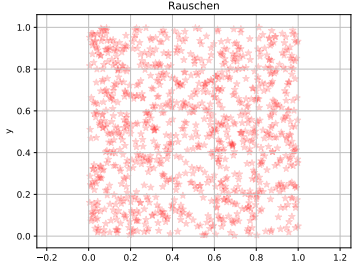






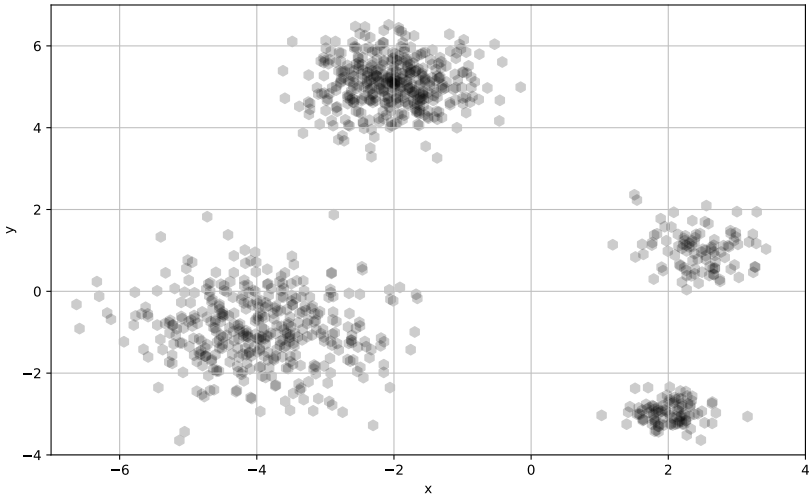






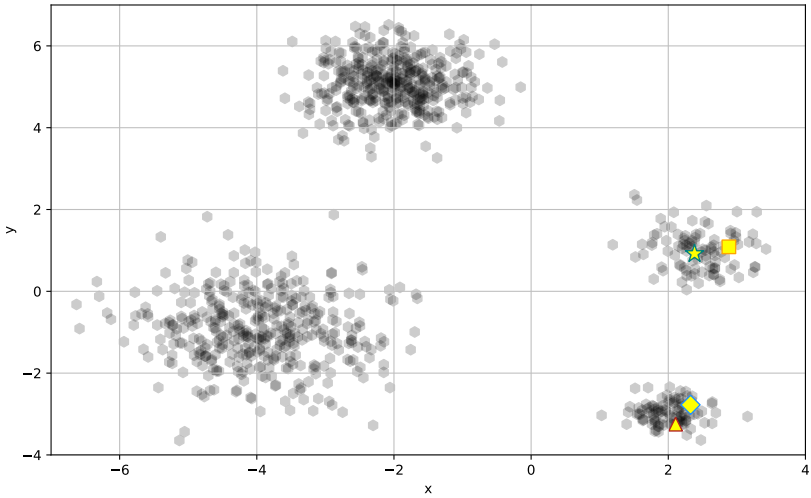
Ablauf von k-Means

1. Initialisiere k Repräsentanten μ_i für die Cluster.
2. Ordne jedes Element dem Cluster zu, bei welchem die Distanz zum Repräsentanten des Clusters am kleinsten ist.
3. Berechne durch Mittelwertbildung die neuen Repräsentanten μ_i der Cluster.



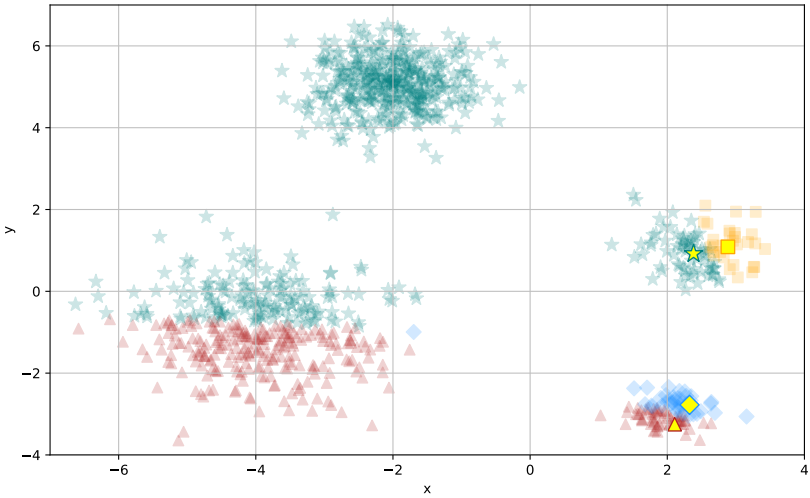
Ablauf von k-Means

1. Initialisiere k Repräsentanten μ_i für die Cluster.
2. Ordne jedes Element dem Cluster zu, bei welchem die Distanz zum Repräsentanten des Clusters am kleinsten ist.
3. Berechne durch Mittelwertbildung die neuen Repräsentanten μ_i der Cluster.



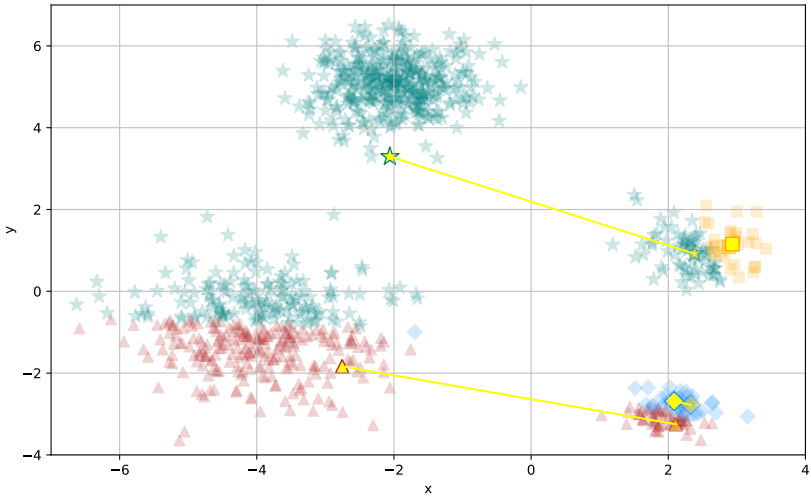
Ablauf von k-Means

1. Initialisiere k Repräsentanten μ_i für die Cluster.
2. Ordne jedes Element dem Cluster zu, bei welchem die Distanz zum Repräsentanten des Clusters am kleinsten ist.
3. Berechne durch Mittelwertbildung die neuen Repräsentanten μ_i der Cluster.



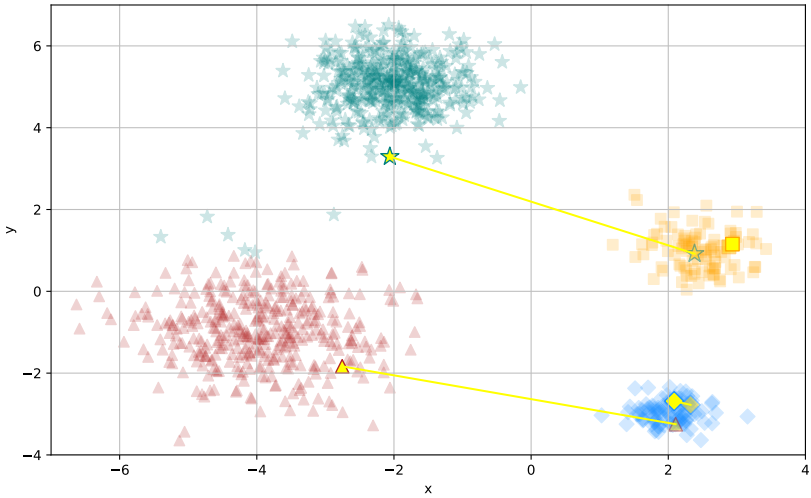
Ablauf von k-Means

1. Initialisiere k Repräsentanten μ_i für die Cluster.
2. Ordne jedes Element dem Cluster zu, bei welchem die Distanz zum Repräsentanten des Clusters am kleinsten ist.
3. Berechne durch Mittelwertbildung die neuen Repräsentanten μ_i der Cluster.



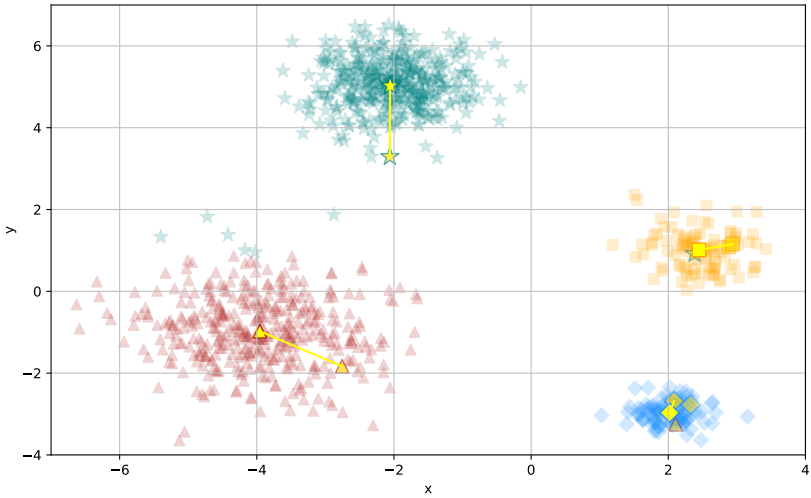
Ablauf von k-Means

1. Initialisiere k Repräsentanten μ_i für die Cluster.
2. Ordne jedes Element dem Cluster zu, bei welchem die Distanz zum Repräsentanten des Clusters am kleinsten ist.
3. Berechne durch Mittelwertbildung die neuen Repräsentanten μ_i der Cluster.



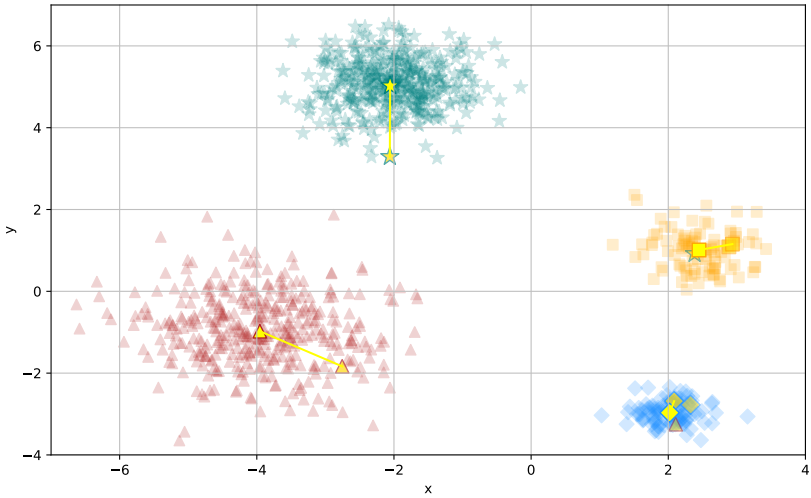
Ablauf von k-Means

1. Initialisiere k Repräsentanten μ_i für die Cluster.
2. Ordne jedes Element dem Cluster zu, bei welchem die Distanz zum Repräsentanten des Clusters am kleinsten ist.
3. Berechne durch Mittelwertbildung die neuen Repräsentanten μ_i der Cluster.



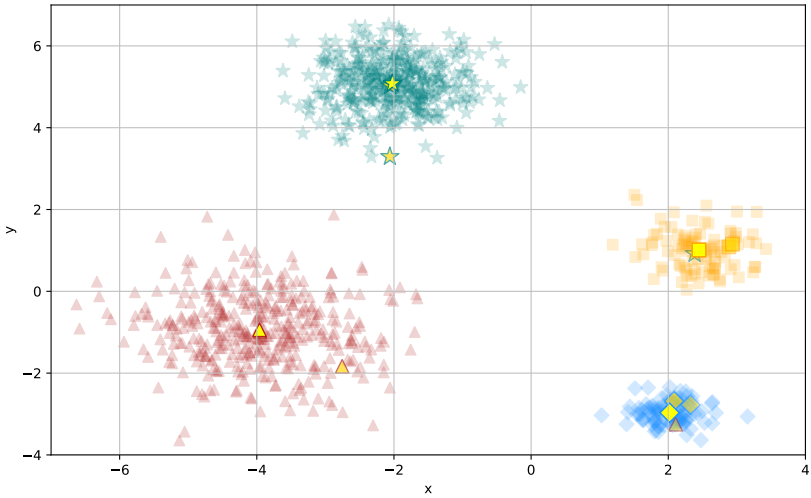
Ablauf von k-Means

1. Initialisiere k Repräsentanten μ_i für die Cluster.
2. Ordne jedes Element dem Cluster zu, bei welchem die Distanz zum Repräsentanten des Clusters am kleinsten ist.
3. Berechne durch Mittelwertbildung die neuen Repräsentanten μ_i der Cluster.



Ablauf von k-Means

1. Initialisiere k Repräsentanten μ_i für die Cluster.
2. Ordne jedes Element dem Cluster zu, bei welchem die Distanz zum Repräsentanten des Clusters am kleinsten ist.
3. Berechne durch Mittelwertbildung die neuen Repräsentanten μ_i der Cluster.



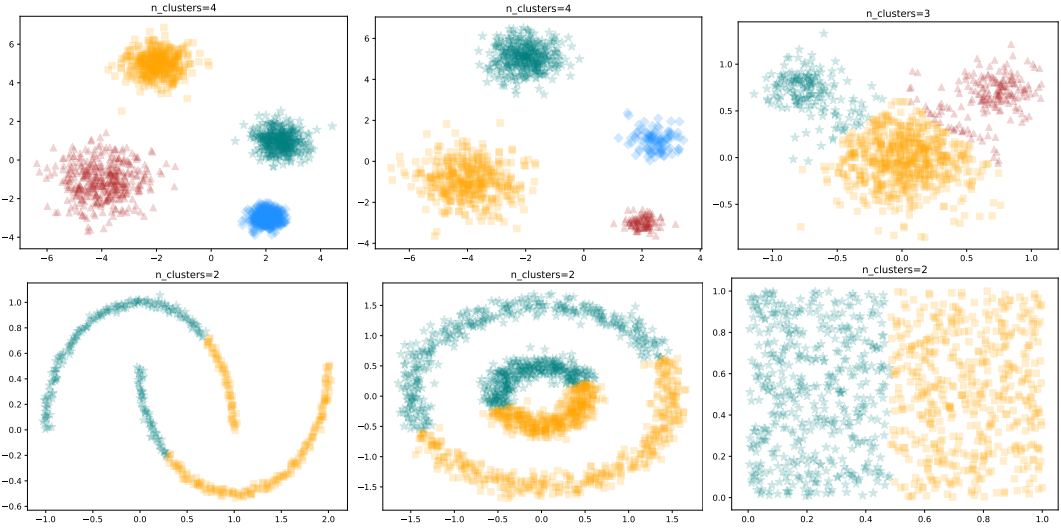
k-Means

- Die Startpunkte lagen im Beispiel unglücklich nah zusammen, trotzdem hat der Algorithmus recht wenig Schritte gebraucht.
- Die unterschiedliche Dichte der Gruppen hatte hier keinen Einfluss auf das Clustering.
- Das Kriterium für die Qualität dieser Aufteilung ist, dass die Summe der Abweichungen von den Cluster-Repräsentanten μ_i in der gewählten Distanzmetrik d minimal ist.
- Mathematisch entspricht dies der Optimierung der Funktion, hier bzgl. eines Minimums,

$$J = \sum_{i=1}^{k} \sum_{x_j \in C_i} d(x_j, \mu_i)$$

- Wichtig: k-Means baut Cluster in einer gewählten Norm um Repräsentanten. Das bedeutet die Cluster sind in dieser Norm immer konvex. Andere Formen sind nicht möglich.

k-Means auf Testfällen



Laufzeit

- Die Laufzeit von k-Means ist

$$\mathcal{O}(n \cdot r \cdot k \cdot i), \text{ wobei}$$

  n die Anzahl der Samples ist,
  r steht für die Dimension des Vektorraums, also die Anzahl der Merkmale,
  k ist die Zahl der Cluster, die k-Means finden soll und
  i die Anzahl der Iterationen, welche bis zur Konvergenz benötigt werden.

- Für sinnvolle Einsatzfälle ist i dabei eine eher kleine Zahl im Bereich unter hundert.
- Für eine feste Fragestellung ist
  - r konstant und
  - die Anzahl der gesuchten Cluster k konstant.
- Der Algorithmus hat also bzgl. der Datenbankgröße n eine **lineare Komplexität**.

Repräsentanten als Besonderheit

- Im Allgemeinen arbeiten Clusteralgorithmen auf einer Gesamtmenge und teilen diese in einzelne Cluster auf.
- Wenn der Cluster für ein neues Element bestimmt werden soll, muss entweder der Clusteralgorithmus von neuem durchgeführt werden oder ein ML-Verfahren (z. B. k-NN) eine Klassifikation mit den Cluster-Labels durchführen.
- Bei k-Means gibt es die Repräsentanten μ_i, die man dazu nutzen kann um eine Vorhersage zu treffen, in welchem Cluster C_i ein neues Element x wohl wäre.
- Hierzu wird die Distanz zu jedem Repräsentanten μ_i berechnet, d. h. $d(x, \mu_i)$ und dann die Clusterzuordnung mit der kleinsten Distanz zurück geliefert.
- Daher gibt es ausnahmsweise wie bei den überwachten Verfahren die Möglichkeit eine `predict` bereitzustellen, die neue Elemente einem Cluster zuweist. Bei anderen Clusteralgorithmen gibt es dies i.d.R. nicht!
- Ausblick: Es gibt Algorithmen wie z.B. BIRCH, die mit Streaming-Daten umgehen können.

Bessere Start-Repräsentanten

- k-Means ist sehr abhängig von den Startwerten. Deren Wahl kann die Anzahl der Iterationen aber auch die gefunden Cluster beeinflussen.
- Für robustere Startbedingungen wurde **k-Means++** entworfen.
- Der Unterschied liegt ausschließlich in der Initialisierung der Repräsentanten:

Initialisierung in k-Means++

1. Wähle den ersten Repräsentanten μ_1 zufällig.
   2. Berechne für jeden Eintrag x den Abstand $D(x)$ zum nächstgelegenen bereits gewählten Repräsentanten.
   3. Wähle zufällig einen neuen Datenpunkt als neuen Repräsentanten. Hierbei nutzt man jedoch keine uniforme Wahrscheinlichkeit, sondern gewichtet diese proportional zu $D(x)^2$.
4. Kehre zu Schritt 2 zurück, bis k Repräsentanten gewählt wurden.
5. Führe nun den bekannten k-Means durch.

Fuzzy-C-Means

- Während k-Means zeitlich im Bereich der späten fünfziger und sechziger Jahre aufkam, wurde etwa ein Jahrzehnt später eine Fuzzy-Variante durch J. C. Dunn vorgestellt.
- Während man in der klassischen Logik diese Aussage nur mit Wahr (1) und Falsch (0) beantworten kann, können wir für die Fuzzy-Logik hier einer Aussage einen Wahrheitswert zwischen 0 und 1 zuweisen.
- Mit dieser Fuzzy-Aussage zum Wahrheitsgehalt geht auch die Fuzzy-Zugehörigkeit mit einer Menge einher.
- Der Rhein wird also vielleicht nur mit einem Wahrheitswert von z. B. 0.6 zur Menge der langen Flüsse gehören.

- Annahmen: n Datensätze und C Cluster sollen gebildet werden
- Für den Fuzzy-Ansatz benötigen wir für jeden der n Datensätze somit C Werte.
- Damit ergibt sich für die Aussagen in welchem Maße der Datensatz zu einem Cluster zugehörig ist eine Matrix $W \in \mathbb{R}^{c \times n}$.
- Diese Matrix enthält Werte von 0 bis 1. Jeder Eintrag w_{ij} gibt somit den Grad an, dem sich der Datensatz j dem Cluster i zugehörig fühlt.
- Alle Basisideen von k-Means bleiben erhalten und wir minimieren nun eine Funktion

$$J = \sum_{i=1}^{C} \sum_{j=1}^{n} (w_{ij})^m d(x_j, \mu_i) \ .$$

- Die primäre Änderung ist der Faktor w_{ij}^m und seine Interpretation.
- Durch ihn fließt die Fuzzy-Zugehörigkeit so in das Funktional ein.
- Komplexer ist die Rolle von m, dem **Fuzzifier**.

Fuzzifier

$$J = \sum_{i=1}^{C} \sum_{j=1}^{n} (w_{ij})^m d(x_j, \mu_i) \ .$$

- Für den **Fuzzifier** gilt zunächst $m \geq 1$.
- Seine Wahl verändert, wie scharf die Zugehörigkeit zu Clustern gewertet wird.
- Je größer m wird, desto stärker werden jedoch Werte kleiner Eins reduziert.
- Ein großes m führt also zu unschärferen Clustern.
- Wählt man $m = 1$, erhält man nachdem der Algorithmus konvergiert ist, scharfe Mitgliedschaften wie schon beim k-Means.
- Werte größer als 3 sind unüblich und wenig erfolgversprechend.
- Sollte kein spezieller Grund durch Expertenwissen vorliegen, wird daher in der Regel die Mitte, also $m = 2$, als Ansatz gewählt.

- Für ein festes m ist der Algorithmus zum Auffinden des Minimums ähnlich zu dem von k-Means.
- Wir erhalten nur **eine weitere Nebenbedingung**. Bei k-Means hatten wir nur die Bedingung: Die Cluster sind nicht-leer.
- Im Fuzzy-Ansatz ist diese alte Bedienung durch die folgende Forderung modelliert:

$$\sum_{j=1}^{n} w_{ij} > 0 \text{ für alle } i = 1, \ldots, C$$

- Die neue Nebenbedingung ist, dass die Summe der Zugehörigkeiten zu den Clustern für jeden Datensatz 1 ist, das bedeutet

$$\sum_{i=1}^{C} w_{ij} = 1 \text{ für alle } j = 1, \ldots, n\,. \tag{1}$$

- Damit bezieht eine Bedingung sich auf die Zeilen der Matrix W und die andere auf die Spalten.

Vorgehen zur Lösung des Minimierungsproblems:

1. Initialisiere k Repräsentanten μ_i für die Cluster
2. Berechne für jedes Element bzgl. jedes Clusters ein Maß der Zugehörigkeit mittels:

$$w_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_i - \mu_j\|}{\|x_i - \mu_k\|} \right)^{\frac{2}{m-1}}} = \frac{1}{\|x_i - \mu_j\|^{\frac{2}{m-1}} \cdot \sum_{k=1}^{C} \|x_i - \mu_k\|^{\frac{-2}{m-1}}} \tag{2}$$

3. Berechne durch gewichtete Mittelwertbildung die neuen Repräsentanten μ_i der Cluster

$$\mu_i = \sum_{j=1}^{n} \underbrace{\frac{(w_{ij})^m}{\sum_{j=1}^{n} (w_{ij})^m}}_{=\omega_j} x_j = \frac{1}{\sum_{j=1}^{n} (w_{ij})^m} \sum_{j=1}^{n} (w_{ij})^m x_j \tag{3}$$

$$w_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_i - \mu_j\|}{\|x_i - \mu_k\|} \right)^{\frac{2}{m-1}}} = \frac{1}{\|x_i - \mu_j\|^{\frac{2}{m-1}} \cdot \sum_{k=1}^{C} \|x_i - \mu_k\|^{\frac{-2}{m-1}}}$$

- Jedoch ist schon die linke Formulierung der Gewichte vielleicht nicht direkt intuitiv zugänglich und man fragt sich beim Lesen: was passiert hier eigentlich?
- Man nimmt den Abstand zu einem Zentrum μ_j und teilt diesen durch die Summe der Abstände zu allen Zentren.
- Ein Ziel ist es sicherzustellen, dass die Summe aller dieser Gewichte eben entsprechend Gleichung (1) 1 ergibt.
- Da die Summe für alle gleich ist, haben wir sonst den üblichen Effekt:

Großer Abstand $\rightsquigarrow$ Kleines Gewicht

Kleiner Abstand $\rightsquigarrow$ Großes Gewicht

Zwei Beispiel mit $m = 2$ und drei Clusterzentren

Beispiel 1

Stellen wir uns ein Gewicht vor, bei dem der Abstand zu μ_j sehr klein ist, also ε, und der Abstand zu den beiden anderen jeweils 1. Was erhalten wir dann als Wert?

$$\frac{1}{\left(\frac{\varepsilon}{\varepsilon}\right)^2 + \left(\frac{\varepsilon}{1}\right)^2 + \left(\frac{\varepsilon}{1}\right)^2} \approx 1$$

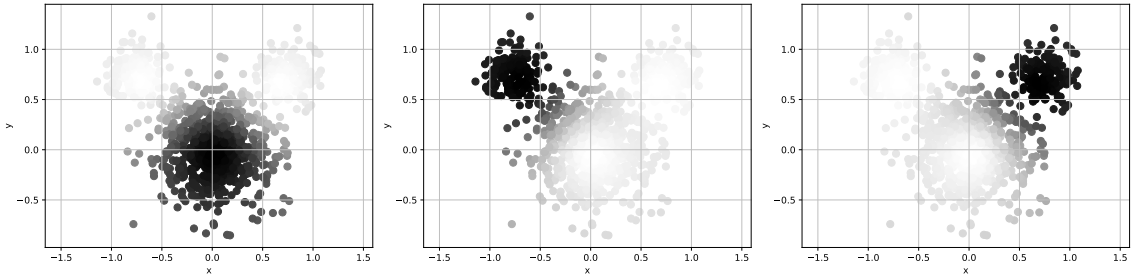
Beispiel 2

Nun noch der Fall, in dem der Datensatz zu allen Zentren den gleichen Abstand, sagen wir $1/2$ hat. Hier ergibt sich mit wie gewünscht eine Gleichverteilung für alle Zentren:

$$\frac{1}{\left(\frac{1/2}{1/2}\right)^2 + \left(\frac{1/2}{1/2}\right)^2 + \left(\frac{1/2}{1/2}\right)^2} = \frac{1}{3}$$

Beispiel Mouse Set

- Die Information des Fuzzy-Ansatzes kann man auch nutzen, um z. B. Randbereiche abzutrennen oder die Gewichte in der späteren Verarbeitung von Daten neu zu bewerten.
- Werden die Gewichte lediglich zur Clusterung verwendet, ist es i. d. R. günstiger, den normalen k-Means zu verwenden. Die Ergebnisse sind meistens vergleichbar.



Wahrscheinlichkeit, zu einem speziellen Cluster zu gehören