

Maschinelles Lernen II

Vorlesung “Computerlinguistische Techniken”
Alexander Koller

22. Januar 2016

Heute

- Überwachtes Lernen:
 - ▶ Maximum-Entropy-Modelle
- Unüberwachtes Lernen:
 - ▶ Clustering


Maximum Entropy: Motivation

- Angenommen, wir entdecken beim POS-Tagging das neue Wort *zyxxy*. Wie ist die W.verteilung $P(c|w = zyxxy)$ über POS-Tags?
- Zunächst angenommen, wir wissen nichts über *zyxxy*. Es gibt 45 POS-Tags.

Maximum Entropy: Motivation

- Angenommen, wir entdecken beim POS-Tagging das neue Wort *zyxxy*. Wie ist die W.verteilung $P(c|w = zyxxy)$ über POS-Tags?
- Zunächst angenommen, wir wissen nichts über *zyxxy*. Es gibt 45 POS-Tags.

NN	JJ	NNS	VB	IN	PRP	CC	CD
1/45	1/45	1/45	1/45	1/45	1/45	1/45	1/45


 $P(c|w=zyxxy)$

(Beispiel nach Jurafsky & Martin)

Maximum Entropy: Motivation

- Jetzt haben wir ein bisschen annotierte Daten für *zyxxy*. Alle Instanzen waren mit NN, NNS, JJ oder VB annotiert.
 - ▶ d.h.: $P(c \in \{NN, NNS, JJ, VB\} \mid zyxxy) = 1$
- Wie ändert sich Ihre Schätzung der W.verteilung?

Maximum Entropy: Motivation

- Jetzt haben wir ein bisschen annotierte Daten für *zyxxy*. Alle Instanzen waren mit NN, NNS, JJ oder VB annotiert.
 - ▶ d.h.: $P(c \in \{NN, NNS, JJ, VB\} \mid zyxxy) = 1$
- Wie ändert sich Ihre Schätzung der W.verteilung?

NN	JJ	NNS	VB	IN	PRP	CC	CD
1/4	1/4	1/4	1/4	0	0	0	0

Maximum Entropy: Motivation

- Wir entdecken jetzt, dass *zyxxy* in 80% der Fälle als NN oder NNS annotiert war.
 - ▶ d.h.: $P(c=NN \vee c=NNS \mid zyxxy) = 0.8$
- Wie ändert sich Ihre Schätzung der W.verteilung?

Maximum Entropy: Motivation

- Wir entdecken jetzt, dass *zyxxy* in 80% der Fälle als NN oder NNS annotiert war.
 - ▶ d.h.: $P(c=NN \vee c=NNS \mid zyxxy) = 0.8$
- Wie ändert sich Ihre Schätzung der W.verteilung?

(NB: Alter Constraint $P(NN \vee JJ \vee NNS \vee VB \mid zyxxy) = 1$ immer noch erfüllt!)

Maximum Entropy: Motivation

- Wir entdecken jetzt, dass *zyxxy* in 80% der Fälle als NN oder NNS annotiert war.
 - ▶ d.h.: $P(c=NN \vee c=NNS \mid zyxxy) = 0.8$
- Wie ändert sich Ihre Schätzung der W.verteilung?

NN	JJ	NNS	VB	IN	PRP	CC	CD
4/10	1/10	4/10	1/10	0	0	0	0

(NB: Alter Constraint $P(NN \vee JJ \vee NNS \vee VB \mid zyxxy) = 1$ immer noch erfüllt!)

Maximum Entropy: Motivation

- Schließlich finden wir heraus, dass eines von 20 englischen Wörtern ein Verb ist.
 - ▶ d.h.: $P(c=VB) = 1/20$, über alle Wörter gerechnet
- Wie ändert sich Ihre Schätzung der W.verteilung?

Maximum Entropy: Motivation

- Schließlich finden wir heraus, dass eines von 20 englischen Wörtern ein Verb ist.
 - ▶ d.h.: $P(c=VB) = 1/20$, über alle Wörter gerechnet
- Wie ändert sich Ihre Schätzung der W.verteilung?

NN	JJ	NNS	VB	IN	PRP	CC	CD
4/10	3/20	4/10	1/20	0	0	0	0

(NB: Alte Constraints sind immer noch erfüllt!)

Maximum Entropy: Motivation

- Warum wählen Sie diese Verteilung:

NN	JJ	NNS	VB	IN	PRP	CC	CD
4/10	3/20	4/10	1/20	0	0	0	0

- Und nicht diese?

NN	JJ	NNS	VB	IN	PRP	CC	CD
8/10	3/20	0	1/20	0	0	0	0

Was haben wir gemacht?

- Wir haben drei *Features* definiert:
 - ▶ $f_1(c, w) = 1$ gdw $w = zyxy$ und $c \in \{NN, NNS, JJ, VB\}$
 - ▶ $f_2(c, w) = 1$ gdw $w = zyxy$ und $c \in \{NN, NNS\}$
 - ▶ $f_3(c, w) = 1$ gdw $c = VB$
- Features sind *Funktionen*, die gleichzeitig Klasse c und zu klassifizierendes Objekt w anschauen.

Was haben wir gemacht?

- Wir haben Erwartungswerte für die Features aus Beobachtungen im Korpus bestimmt.

$$E(f) = \sum_{w,c} P(w, c) \cdot f(w, c)$$

- $E(f_1) = P(\text{zyxxy}, \text{NN} \vee \text{NNS} \vee \text{JJ} \vee \text{VB}) = P(\text{zyxxy})$
 $E(f_2) = P(\text{zyxxy}, \text{NN} \vee \text{NNS}) = 0.8 * P(\text{zyxxy})$
 $E(f_3) = P(c = \text{VB}) = 1/20$
- NB: $P(w, c) = P(c | w) * P(w)$

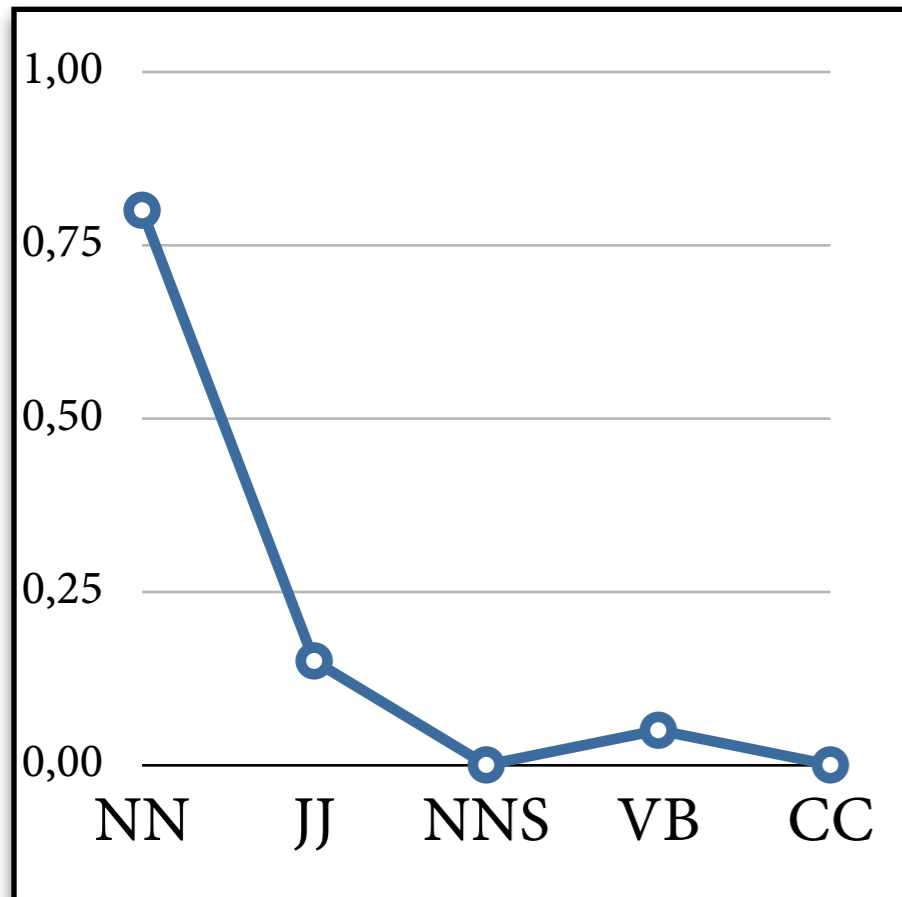
Maximum-Entropy-Modelle

- Wir schätzen jetzt eine W.verteilung \tilde{P} , so dass
 - ▶ $E_{\tilde{P}}(f) = \sum_{w,c} \tilde{P}(w,c) \cdot f(w,c) = E_P(f)$ für alle Features f
 - ▶ das Modell unter allen Modellen mit dieser Eigenschaft die wenigsten Zusatzannahmen macht.
- Formaler Begriff von “wenigste Zusatzannahmen”: maximale *Entropie*.

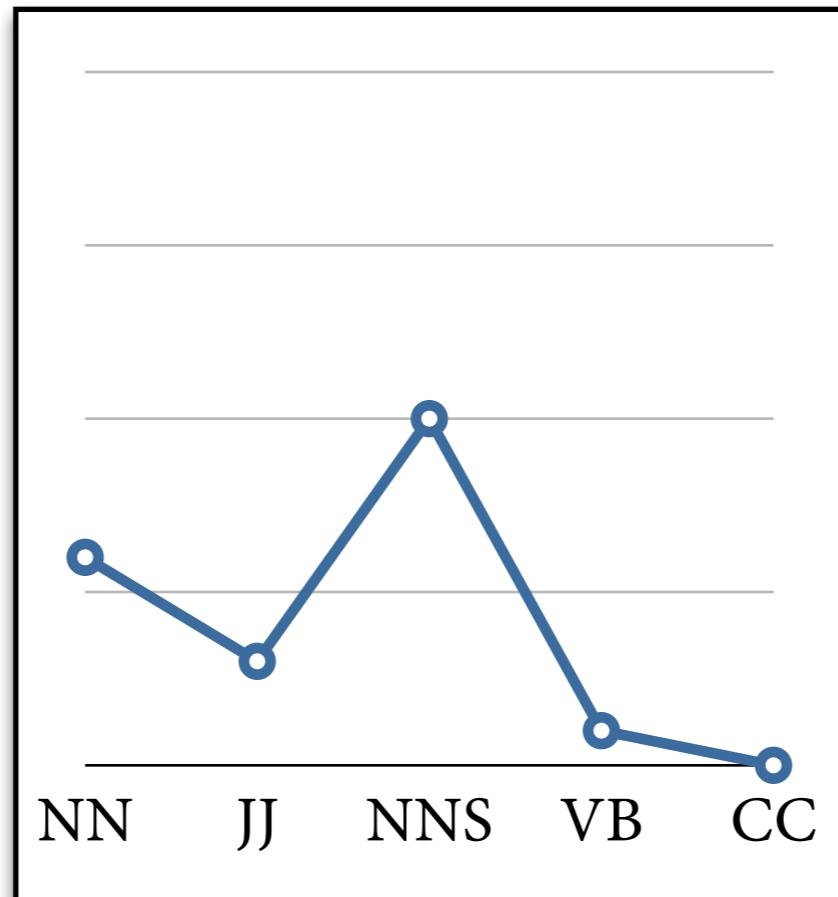
Entropie

- *Entropie* einer W.verteilung ist fundamentales Konzept der Informationstheorie: Vorhersehbarkeit einer Zufallsvariable.
- Definition:
$$H(P) = - \sum_z P(z) \cdot \log P(z)$$
- “Eindeutige” Verteilung hat $H(P) = 0$
($P(a) = 1$ für ein einziges a , $P(x) = 0$ sonst):
- Uniforme Verteilung über N mögliche Ereignisse:
 $H(P) = \log N$

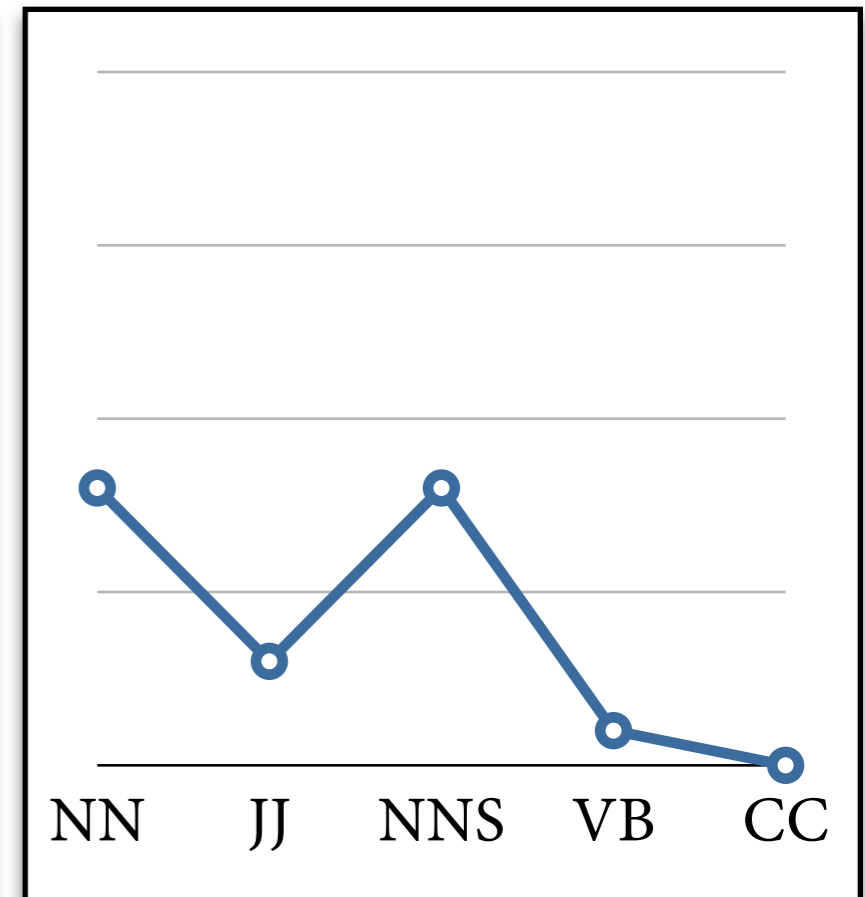
Entropie



$H = 0.61$



$H = 0.99$



$H = 1.02$

“peaky”

maximum
entropy

Featurefunktionen

- Annahme: Menge von Feature-Funktionen $f_i(w, c)$: gemeinsame Eigenschaften von w zu klassifizierendem Objekt w und Klasse c
- Eigenschaften völlig beliebig, gerne auch kompliziert. Müssen auch nicht statistisch unabhängig sein.
- Freie Wahl der Ft.funktionen ist eine zentrale Stärke des MaxEnt-Ansatzes.

Maximum-Entropy-Modelle

- Man kann zeigen, dass die W.verteilung maximaler Entropie immer folgende Form hat:

$$P(c|w) = \frac{e^{\theta \cdot f(c,w)}}{\sum_{c'} e^{\theta \cdot f(c',w)}}$$

wobei θ ein Vektor von Gewichten ist und

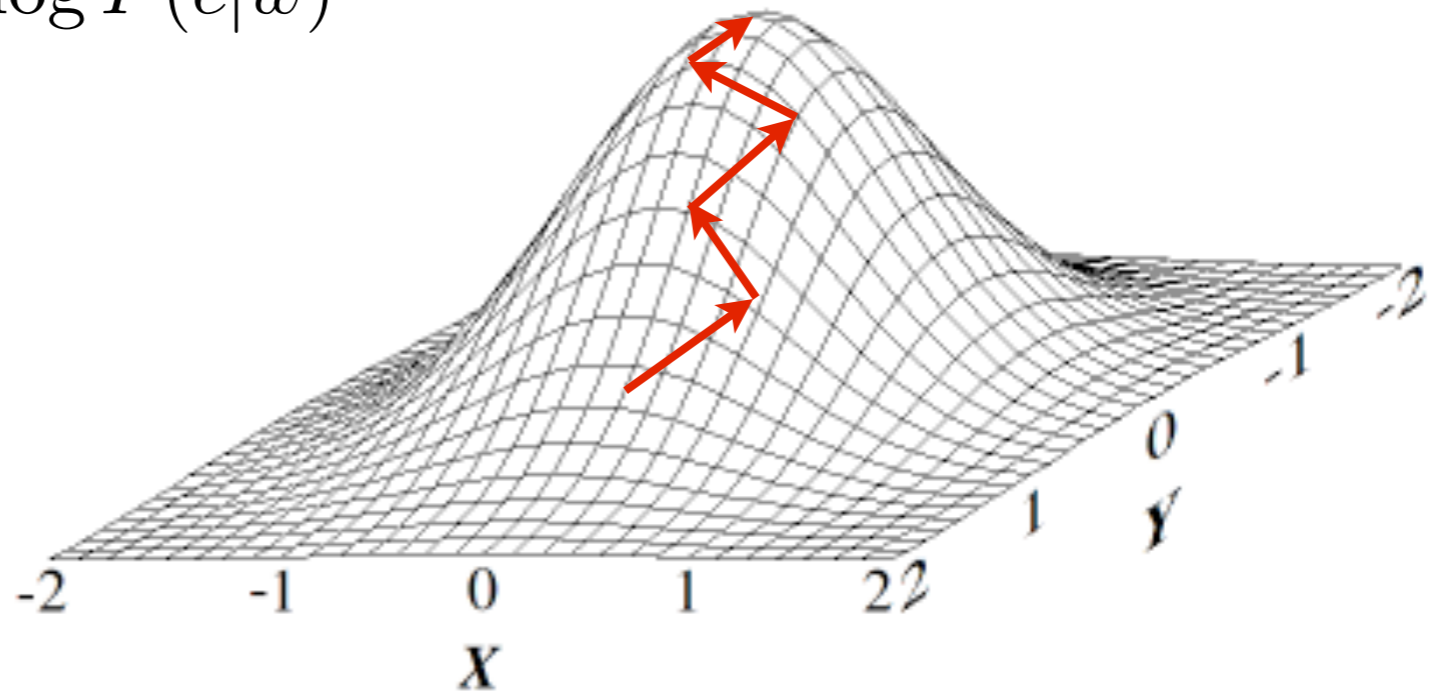
$$\theta \cdot f(c, w) = \sum_i \theta_i \cdot f_i(c, w).$$

- Training = Lernen von Parametern θ , die zu Beobachtungen passen.

Training

- Training mit iterativen Verfahren, typischerweise Gradient Ascent/Descent.
- Grundidee: Maximiere log-likelihood durch Änderung der Parameter in Richtung des steilsten Aufstiegs.

$$\sum_{(w,c) \in \text{Tr}} \text{freq}(c, w) \cdot \log P(c|w)$$



Beispiel

Nigeria ... prince ... bank	Spam
... Viagra ... Tabletten ...	Spam
Nigeria ... president ... bank	Spam
... Vorlesung ... fällt aus ...	kein Spam

$f_1(w, c) = 1$ gdw. w enthält "Nigeria" und $c = \text{Spam}$

$f_2(w, c) = 1$ gdw. w enthält "Vorlesung" und $c = \text{kein Spam}$ etc.

$\theta_1 = 1.01$ $\theta_2 = 2.22$ usw.

$P(\text{Spam} \mid \dots \text{Nigeria} \dots \text{Viagra} \dots) = 0.98$

Beispiele für Featuresets

- Übersetzung Englisch - Französisch (Berger et al. 96):
 - ▶ $ft(e,f) = 1$ gdw $e_{i+1} = \text{"Canada"}$ und $f_i = \text{"à"}$
 - ▶ $ft(e,f) = 1$ gdw "area" in e_{i+1}, \dots, e_{i+3} und $f_i = \text{"dans"}$
 - ▶ $ft(e,f) = 1$ gdw "increase" in e_{i-1}, \dots, e_{i-3} und $f_i = \text{"de"}$
- Training auf bilingualem Korpus (Protokolle des kanadischen Parlaments).

Beispiele für Featuresets

- POS-Tagging (Ratnaparkhi 1996)
verwendet Templates für Features:
 - ▶ $ft(w, p) = 1$ gdw $w_i = X$ und $p = Y$
 - ▶ $ft(w, p) = 1$ gdw w_i endet in X (z.B. “-ing”) und $p = Y$
 - ▶ $ft(w, p) = 1$ gdw w_i enthält Großbuchstaben und $p = Y$
- Erzeugt alle Instanzen (z.B. “ w_i endet in -ing und $p = VBG$ ”) und wirft zu seltene weg.
- 96.6% tagging accuracy auf WSJ

Log-lineare Modelle

- Maximum-Entropy-Modelle heißen auch *log-lineare* Modelle, weil $\log P$ linear (in den Features) ist.

$$\begin{aligned}\log P(c|w) &= \log \frac{e^{\theta \cdot f(c,w)}}{\sum_{c'} e^{\theta \cdot f(c',w)}} \\ &= \log \frac{e^{\theta \cdot f(c,w)}}{Z(w)} \\ &= \theta \cdot f(c,w) - \log Z(w)\end{aligned}$$

- Statt Likelihood kann man den Lerner auch eine andere Zielfunktion optimieren lassen; dann nicht mehr Maximum Entropy, aber immer noch log-linear.

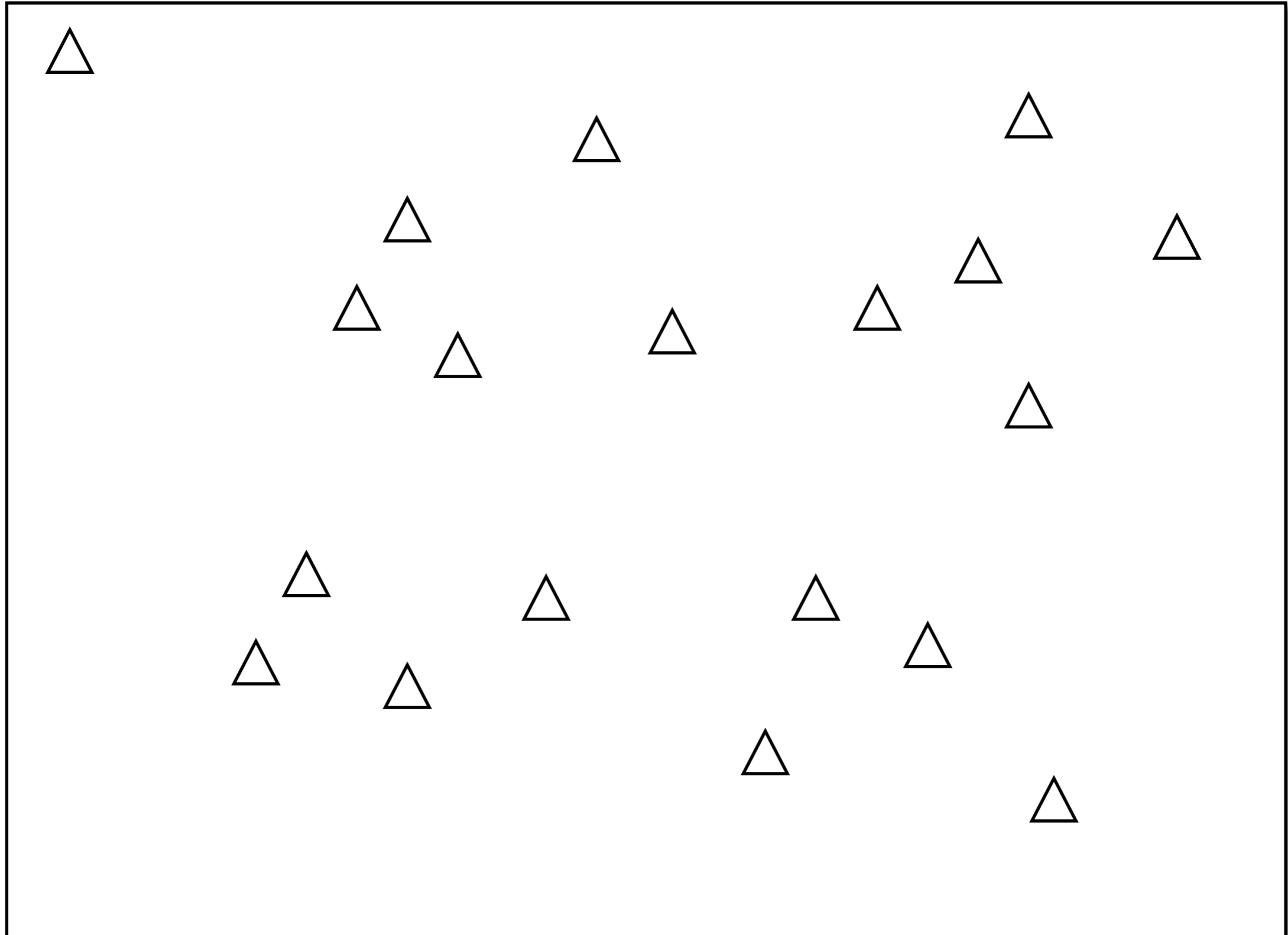
Vor- und Nachteile

- Featurefunktionen extrem mächtig: jede berechenbare Eigenschaft von (w, c) .
- Robust gegenüber statistischen Abhängigkeiten.
- MaxEnt ist *diskriminatives* Modell: Berechnet nur $P(c | w)$, nicht direkt $P(c, w)$ (anders als HMM u.ä.).
- Größerer Trainingsdaten- und Rechenzeitbedarf als Naive Bayes.

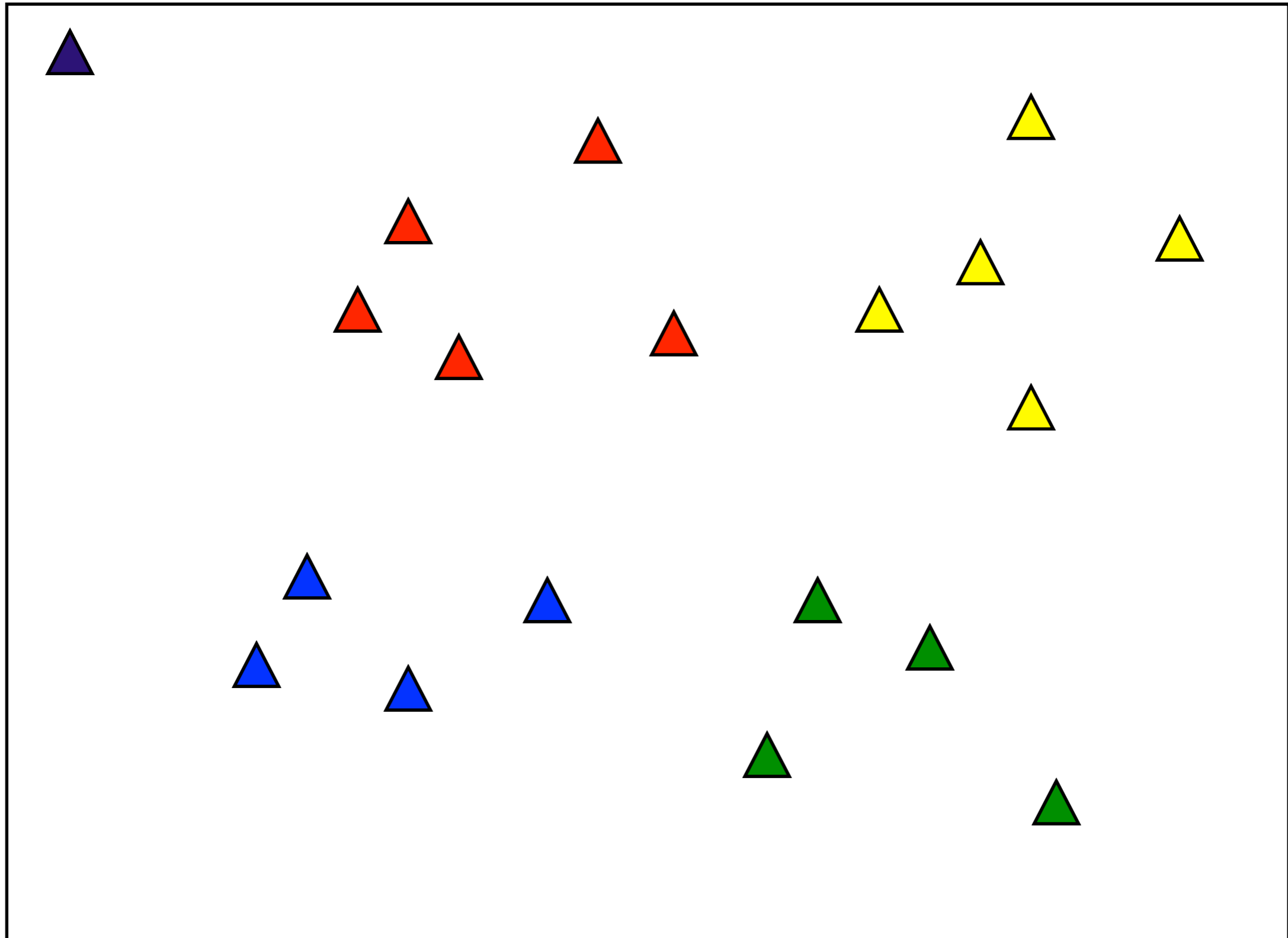
Clustering

- Unüberwachtes Lernen:
 - ▶ Trainingsdaten sind nicht mit Klassen annotiert
 - ▶ Wir wollen die Klassen selbst herausfinden
- Beispiel: Semantische Ähnlichkeit
 - ▶ gegeben: Ähnlichkeitsmaß auf Paaren von Wörtern
 - ▶ gesucht: Klassen von ähnlichen Wörtern
- Ein Lösungsansatz: Clustering.

Clustering



Clustering



Semantische Ähnlichkeit

- Problemstellung: Wie kann man automatisch Wörter in Gruppen zusammenfassen?
 - ▶ “Near-synonymy”: {error, mistake, blunder, faux pas, ...}
 - ▶ Gattungen: {Äpfel, Birnen, Kirschen, ...}
 - ▶ Verbklassen: {essen, konsumieren, trinken, saufen, ...}
- Gegeben ist ein Ähnlichkeitsmaß.
 - ▶ Numerische Features, z.B. aus distributionellem Ansatz: “Wie oft kommt im Korpus in einem Fenster von 5 Wörtern das Wort essen vor?”
 - ▶ Daraus numerisches Ähnlichkeitsmaß definieren: in etwa wie bei k-nearest-neighbor.

Clustering

- Gegeben:
 - ▶ Menge von Objekten
 - ▶ Ähnlichkeitsmaß
- Gesucht: Zuordnung von Objekten zu Clustern, so dass
 - ▶ Objekte im gleichen Cluster ähnlich sind
 - ▶ Objekte in verschiedenen Clustern unähnlich sind

Clustering

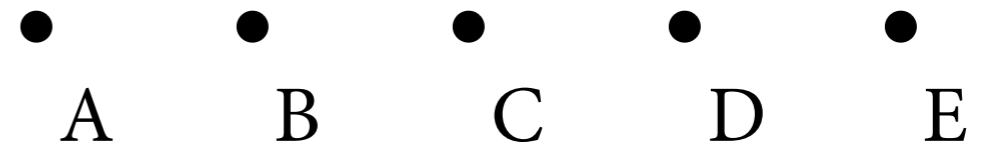
- Grundlegende Ansätze:
 - ▶ *agglomerativ / hierarchisch*: mit feinkörnigen Clustern anfangen und sie nach und nach verschmelzen
 - ▶ *k-means-Clustering*: festgelegte Anzahl k von *Centroids* berechnen; jeder Cluster besteht aus Objekten, die dem gleichen Centroid am ähnlichsten sind

Agglomeratives Clustering

- Form von hierarchischem Clustering:
Große Cluster bestehen rekursiv aus kleinen.
- Ablauf:
 - ▶ Initialisierung: ein eigenes Cluster für jedes Objekt
 - ▶ Schritt: zwei ähnlichste Cluster verschmelzen
 - ▶ Ende: nur ein Cluster bleibt übrig
- Verschiedene Ansätze, um Ähnlichkeit von Clustern aus Ähnlichkeit der Element zu berechnen.

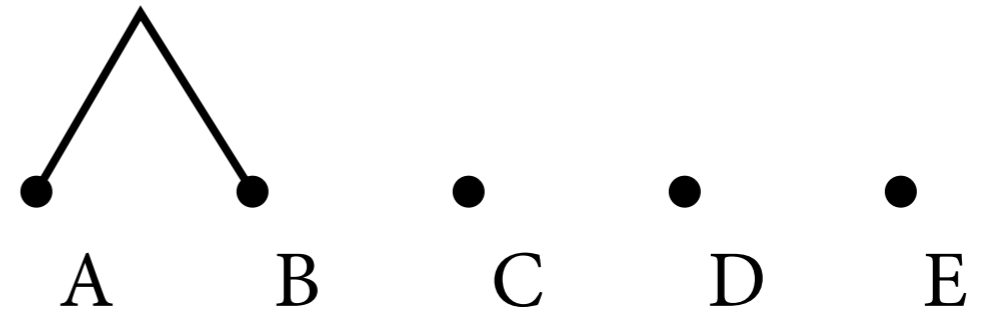
Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			



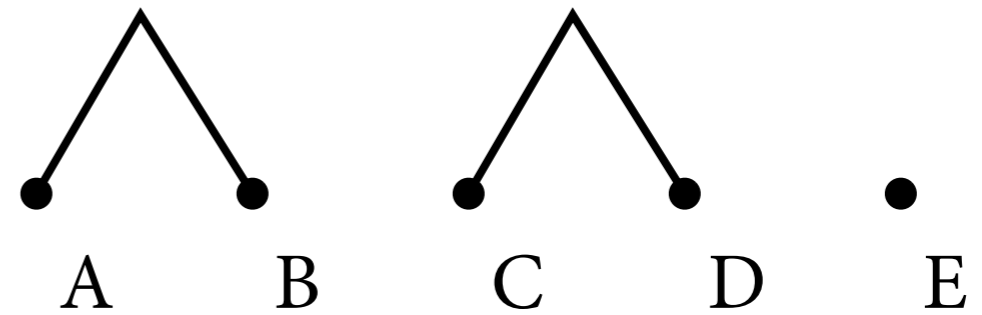
Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			



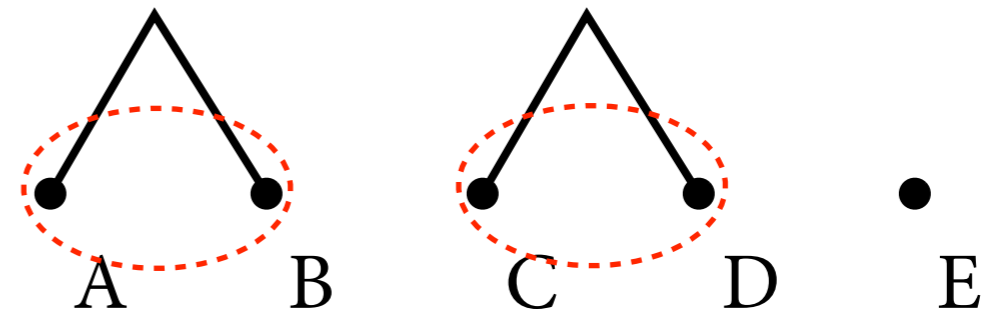
Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			



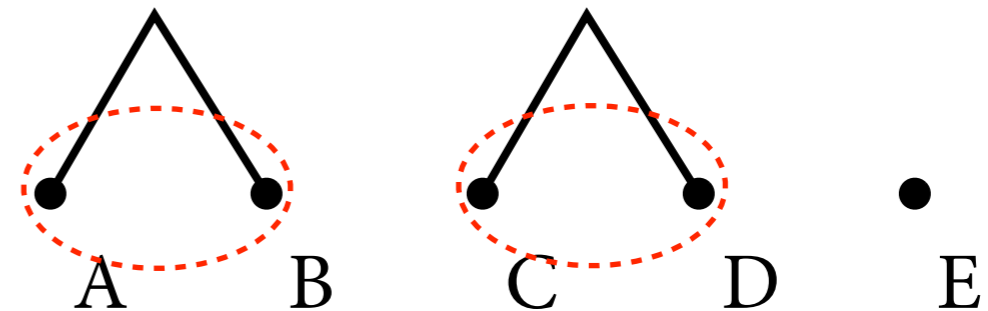
Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			



Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

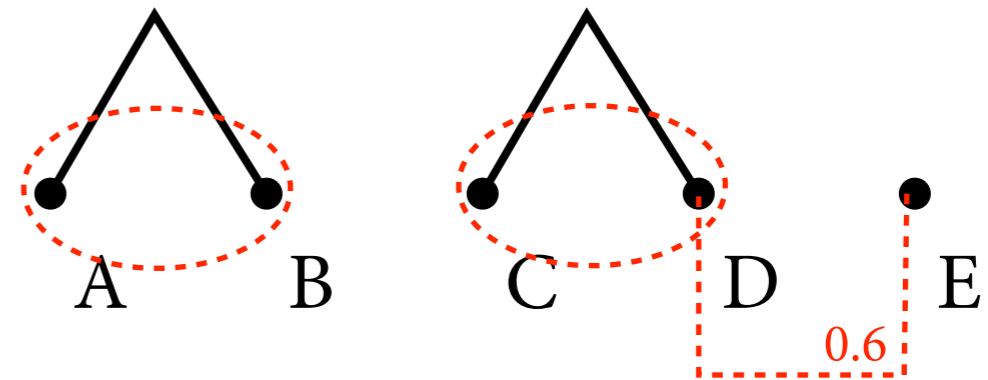


Single-Link Clustering:

Ähnlichkeit von Clustern = maximale Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

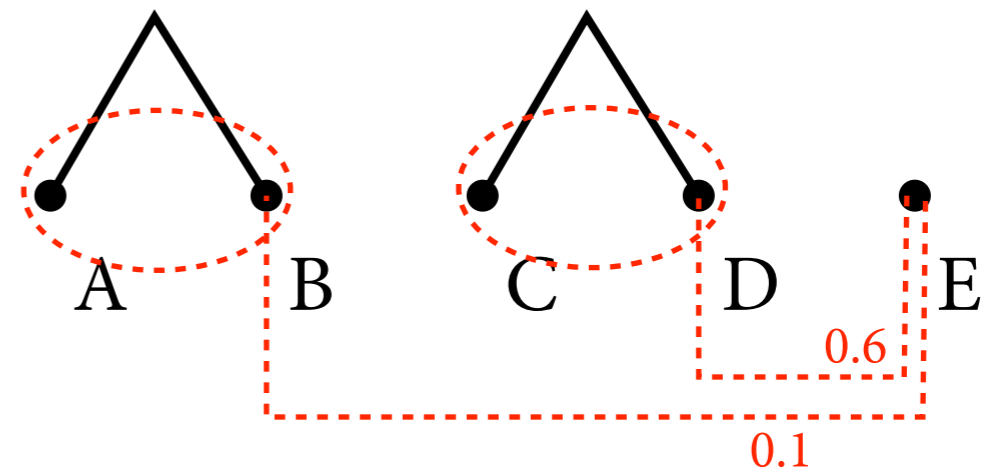


Single-Link Clustering:

Ähnlichkeit von Clustern = maximale Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

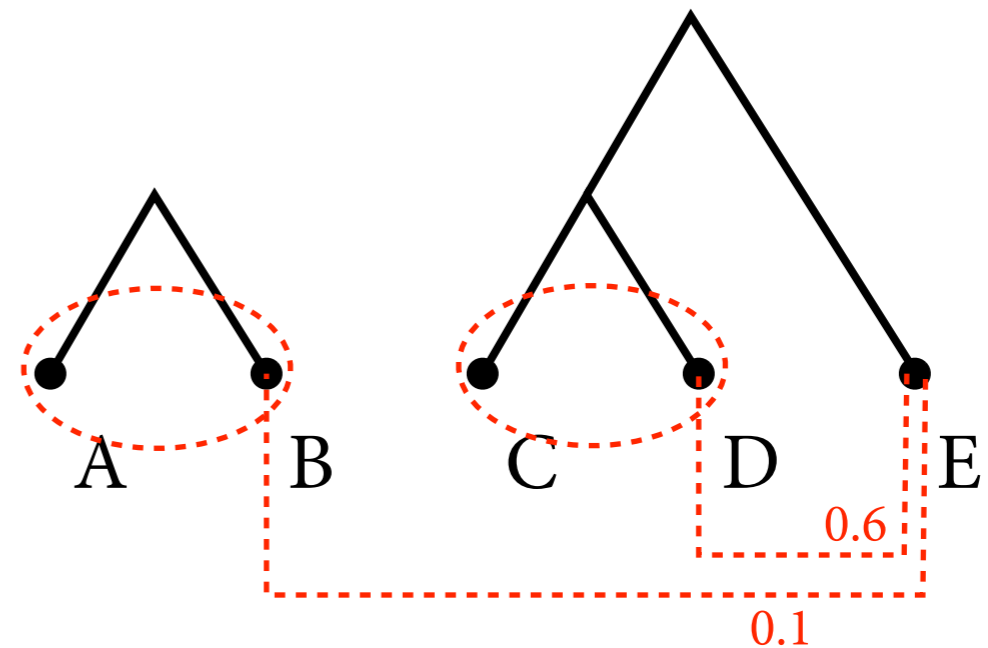


Single-Link Clustering:

Ähnlichkeit von Clustern = maximale Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

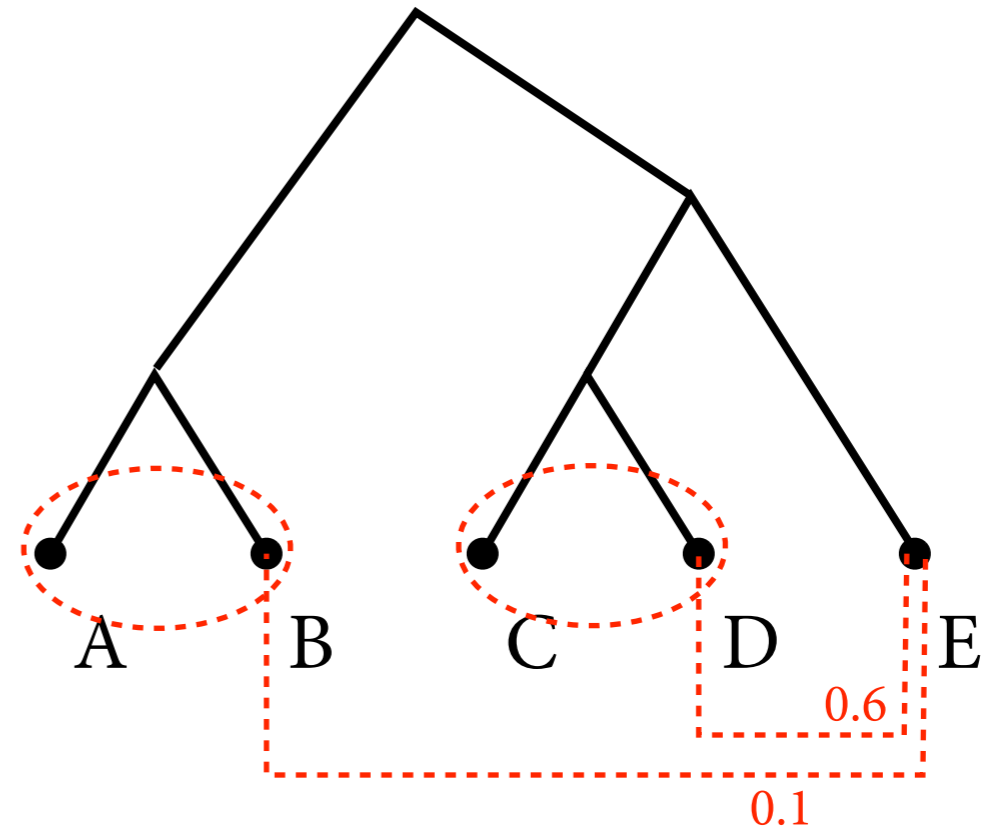


Single-Link Clustering:

Ähnlichkeit von Clustern = maximale Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

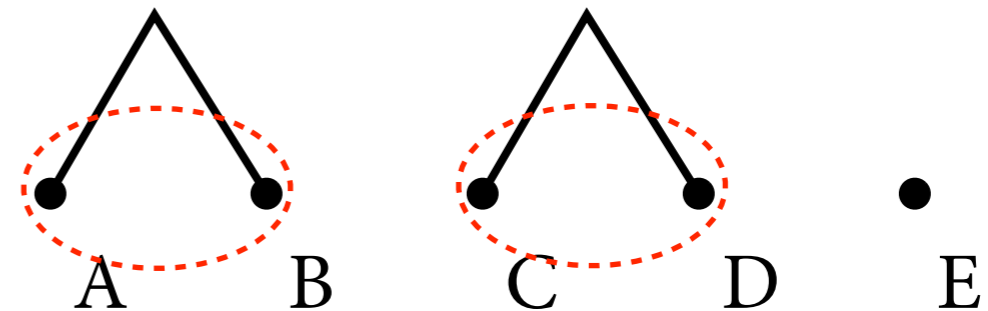


Single-Link Clustering:

Ähnlichkeit von Clustern = maximale Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

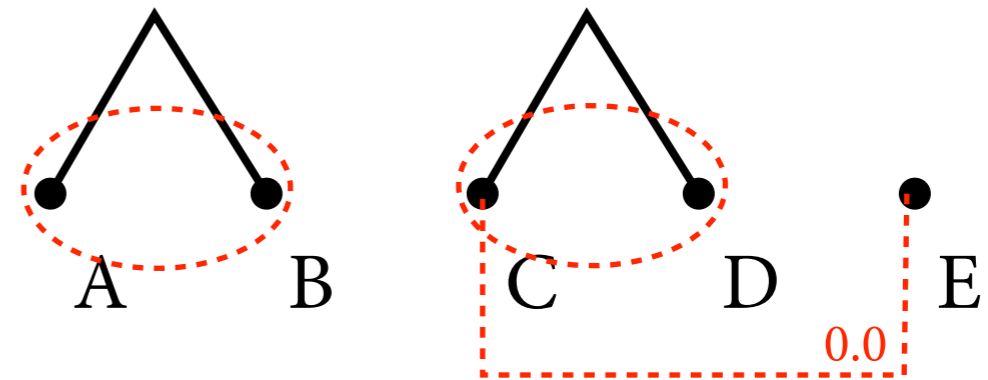


Complete-Link Clustering:

Ähnlichkeit von Clustern = minimale Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

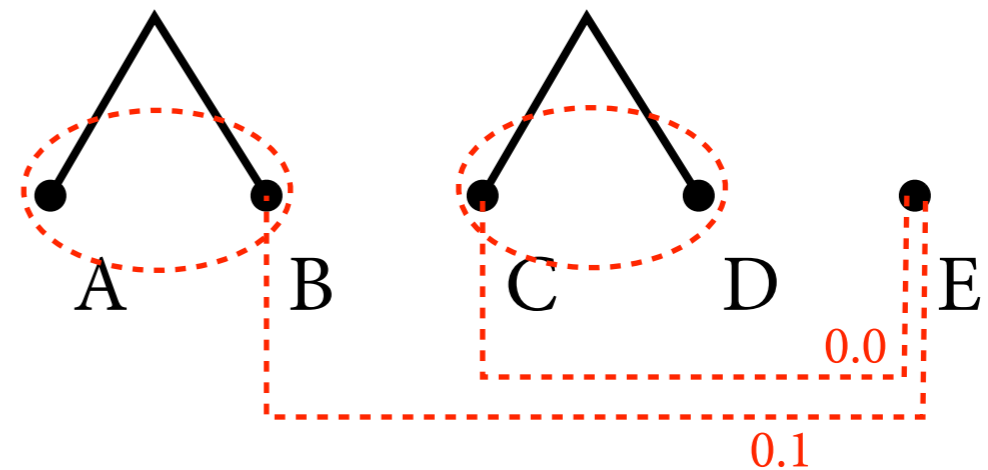


Complete-Link Clustering:

Ähnlichkeit von Clustern = minimale Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

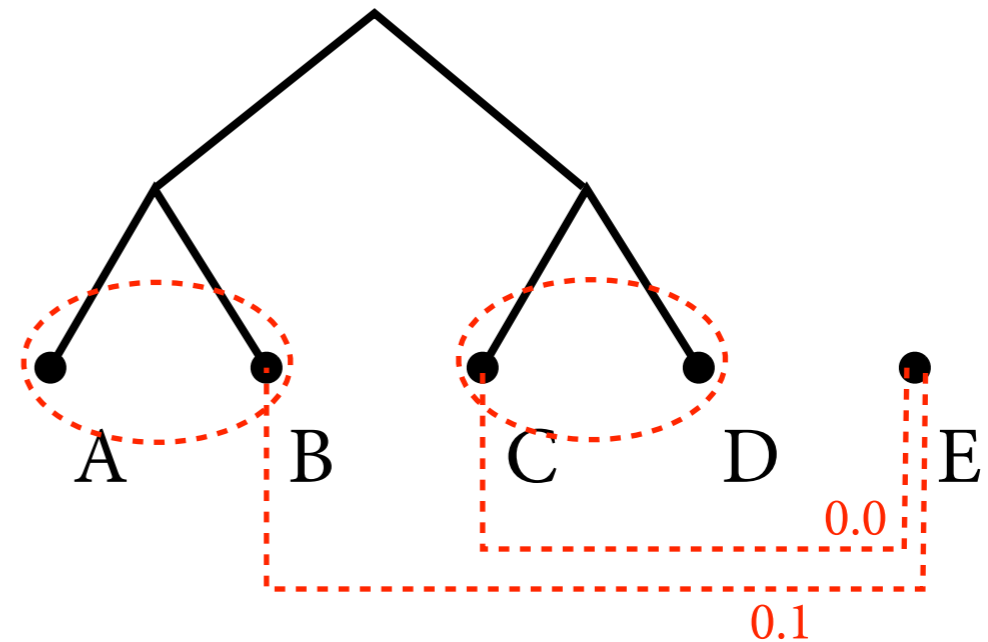


Complete-Link Clustering:

Ähnlichkeit von Clustern = minimale Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

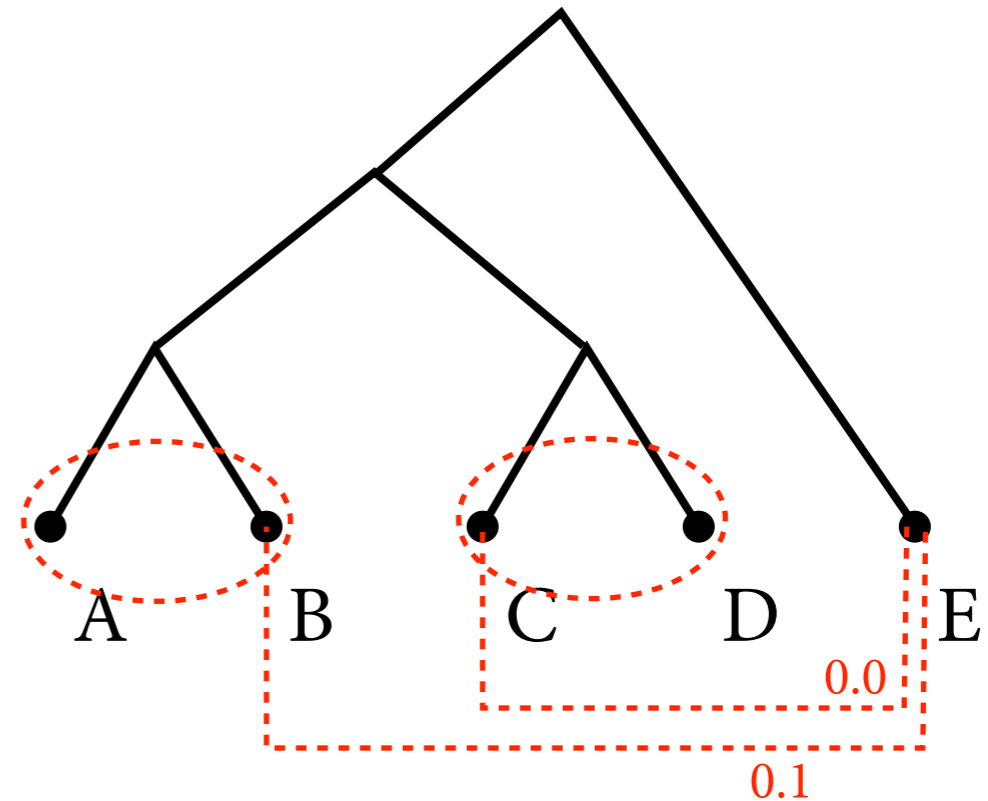


Complete-Link Clustering:

Ähnlichkeit von Clustern = minimale Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

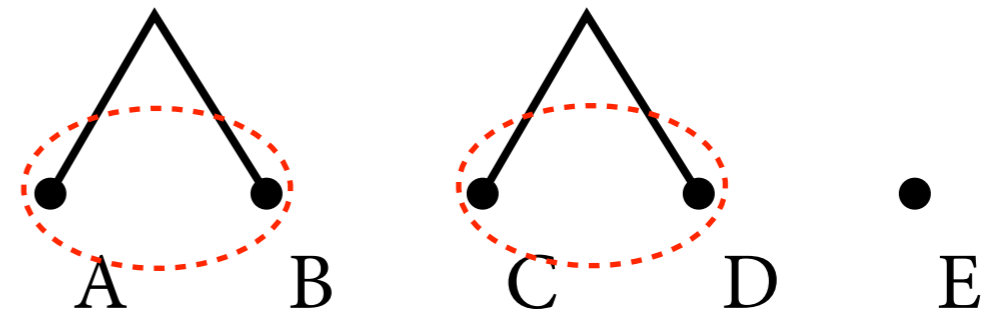


Complete-Link Clustering:

Ähnlichkeit von Clustern = minimale Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

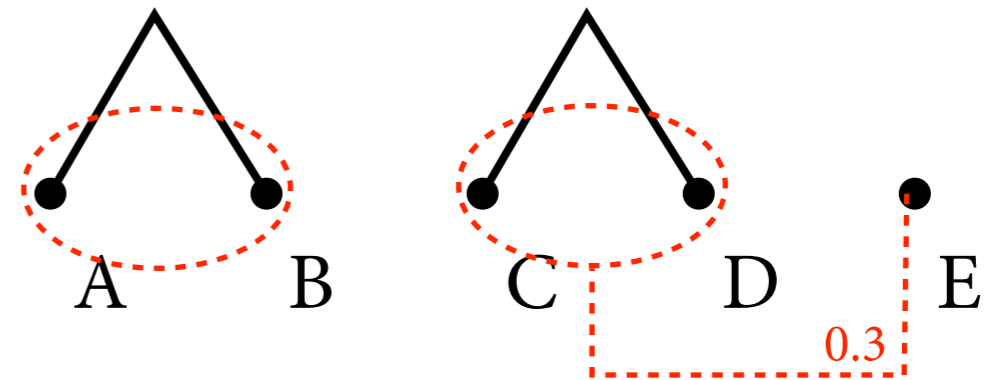


Average-Link Clustering:

Ähnlichkeit von Clustern = durchschnittliche Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

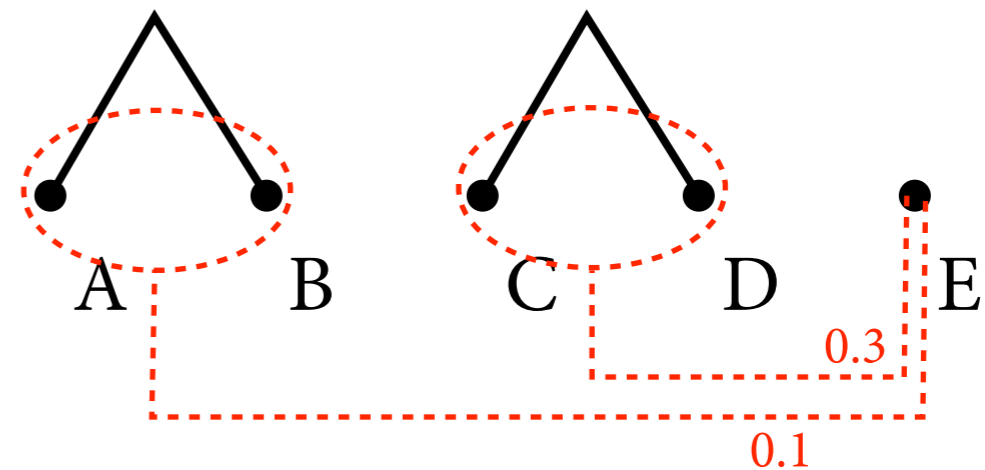


Average-Link Clustering:

Ähnlichkeit von Clustern = durchschnittliche Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

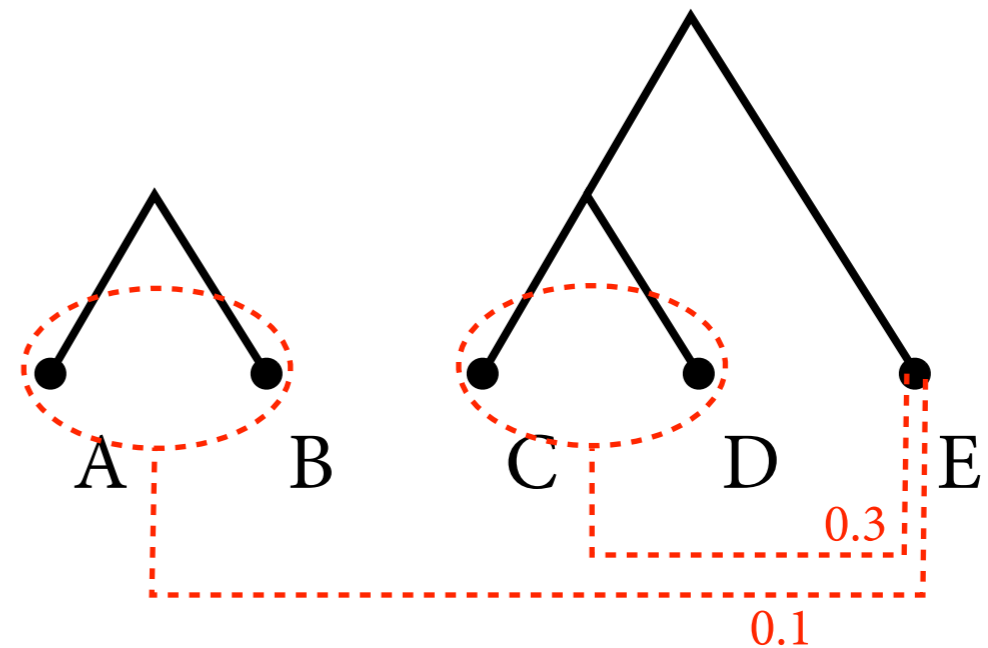


Average-Link Clustering:

Ähnlichkeit von Clustern = durchschnittliche Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

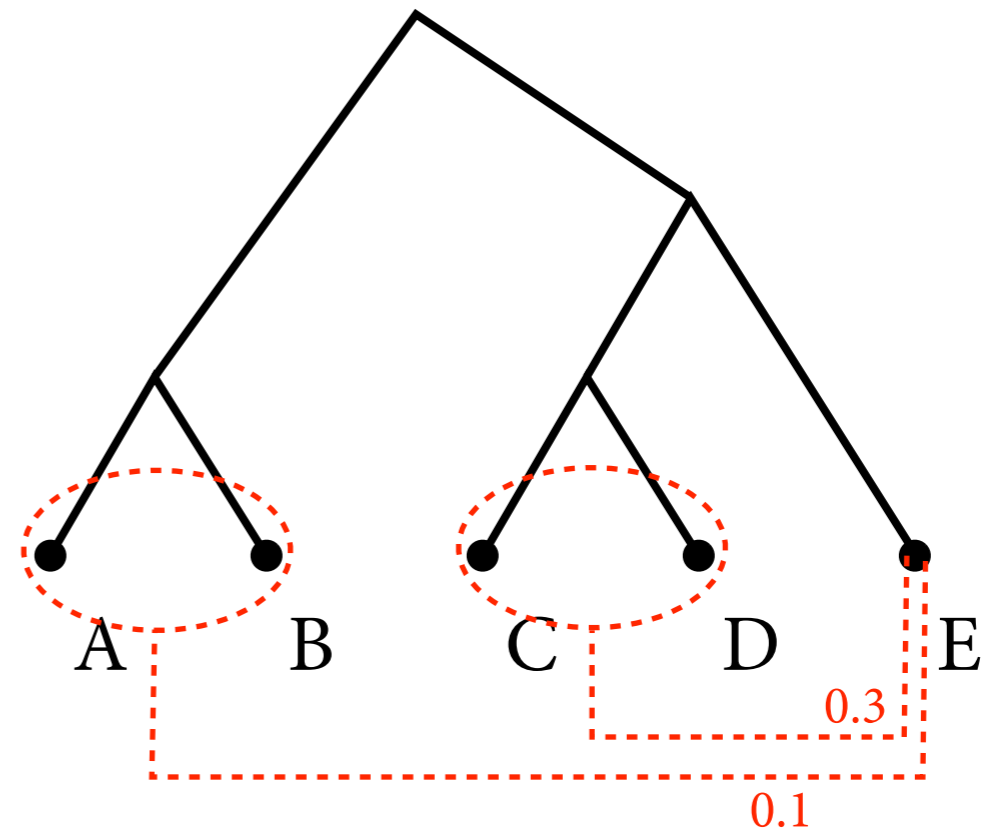


Average-Link Clustering:

Ähnlichkeit von Clustern = durchschnittliche Ähnlichkeit der Elemente

Agglomeratives Clustering

	E	D	C	B
A	0.1	0.2	0.2	0.8
B	0.1	0.1	0.2	
C	0.0	0.7		
D	0.6			

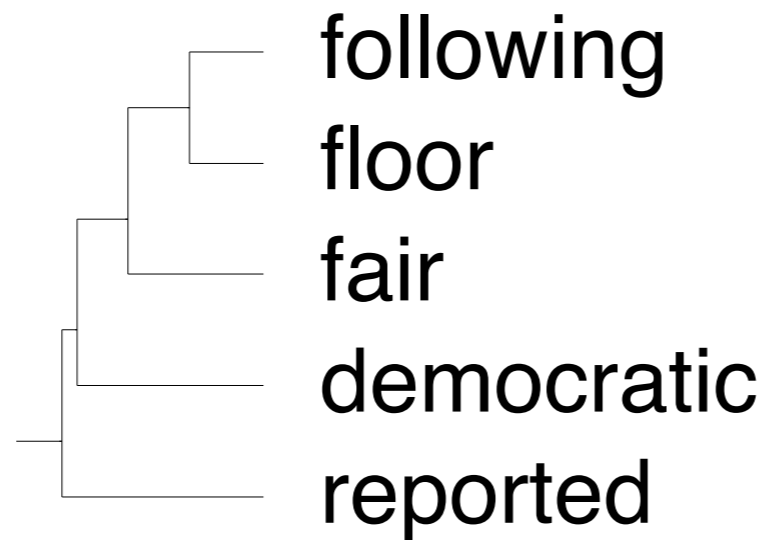
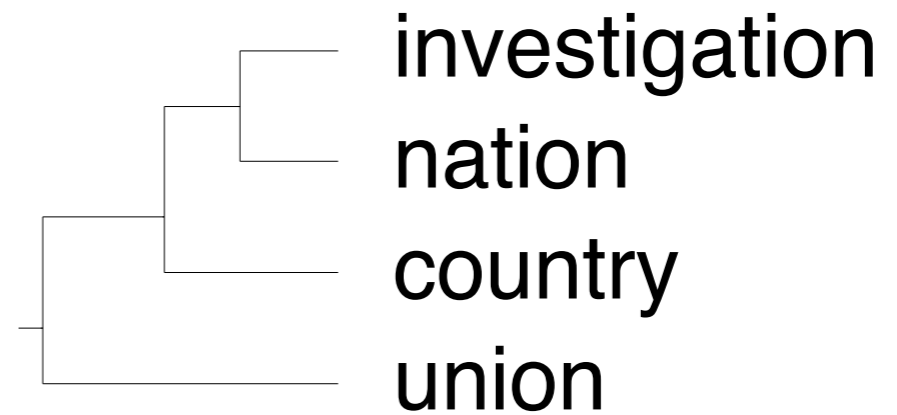
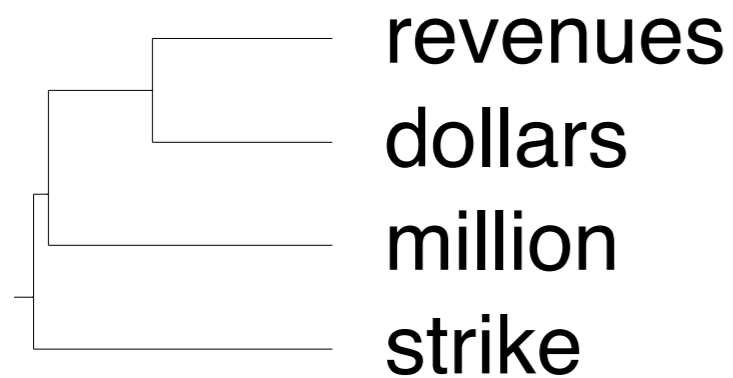


Average-Link Clustering:

Ähnlichkeit von Clustern = durchschnittliche Ähnlichkeit der Elemente

Ähnlichkeitsklassen

Beispielanwendung: Hierarchie von ähnlichen Wörtern.



Ähnlichkeitsklassen

Friday Monday Thursday Wednesday Tuesday Saturday Sunday weekends Sundays
people guys folks fellows CEOs chaps doubters commies unfortunates blokes
down backwards ashore sideways southward northward overboard aloft downwards adrift
water gas coal liquid acid sand carbon steam shale iron
great big vast sudden mere sheer gigantic lifelong scant colossal
American Indian European Japanese German African Catholic Israeli Italian Arab
mother wife father son husband brother daughter sister boss uncle
machine device controller processor CPU printer spindle subsystem compiler plotter
John George James Bob Robert Paul William Jim David Mike
feet miles pounds degrees inches barrels tons acres meters bytes
had hadn't hath would've could've should've must've might've
that tha theat
head body hands eyes voice arm seat eye hair mouth

(Brown et al. 1992)

Ähnlichkeitsklassen

- (4) *abholen* 'pick up', *ansehen* 'watch', *bestellen* 'order', *erwerben* 'purchase', *holen* 'fetch', *kaufen* 'buy', *konsumieren* 'consume', *verbrennen* 'burn', *verkaufen* 'sell'
- (5) *anschauen* 'watch', *erhoffen* 'wish', *vorstellen* 'imagine', *wünschen* 'wish', *überlegen* 'think about'
- (6) *danken* 'thank', *entkommen* 'escape', *gratulieren* 'congratulate'
- (7) *beschleunigen* 'speed up', *bilden* 'constitute', *darstellen* 'illustrate', *decken* 'cover', *erfüllen* 'fulfil', *erhöhen* 'raise', *erledigen* 'fulfil', *finanzieren* 'finance', *füllen* 'fill', *lösen* 'solve', *rechtfertigen* 'justify', *reduzieren* 'reduce', *senken* 'lower', *steigern* 'increase', *verbessern* 'improve', *vergrößern* 'enlarge', *verkleinern* 'make smaller', *verringern* 'decrease', *verschieben* 'shift', *verschärfen* 'intensify', *verstärken* 'intensify', *verändern* 'change'
- (8) *ahnen* 'guess', *bedauern* 'regret', *befürchten* 'fear', *bezweifeln* 'doubt', *merken* 'notice', *vermuten* 'assume', *weißen* 'whiten', *wissen* 'know'

Diskussion

- Ähnlichkeitsklassen nützlich in Anwendungen.
 - ▶ Clustering ist Standardverfahren
 - ▶ Finden oft erstaunlich nichttriviale Klassen
- Einschränkungen:
 - ▶ Features zu flach
 - ▶ Ähnlichkeitsmaß zu flach
 - ▶ Wie viele Cluster?

Zusammenfassung

- Maschinelles Lernen:
 - ▶ überwacht: Klassen sind vorgegeben
(Memory-Based, Naive Bayes, Maximum Entropy)
 - ▶ unüberwacht: Lerner findet Klassen selbst
(z.B. Clustering)
- ML ist aus moderner CL nicht wegzudenken.
 - ▶ Es lohnt sich, separate Vorlesung dazu zu besuchen.