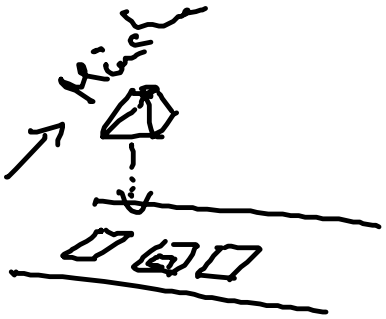
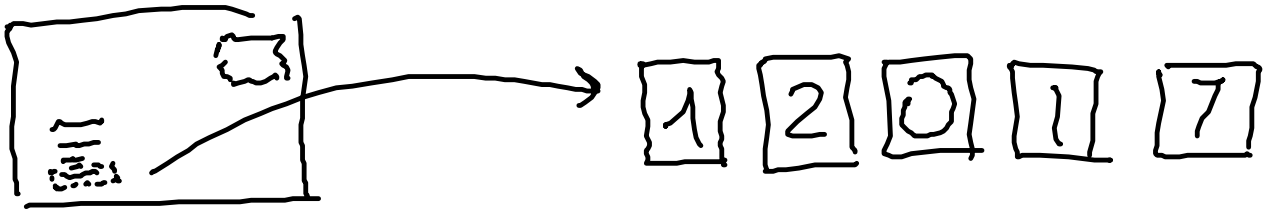
# Mustererkennung

Gesichter, Schilder, ... viele Anwendungsgebiete

Bsp: Briefe → Postleitzahlen

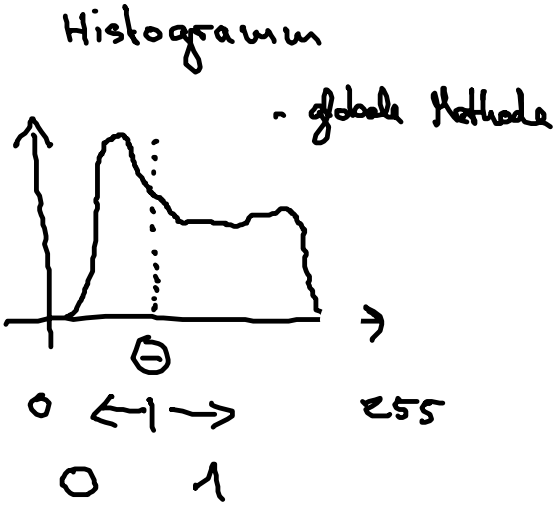



Vorverarbeitungsschritte:

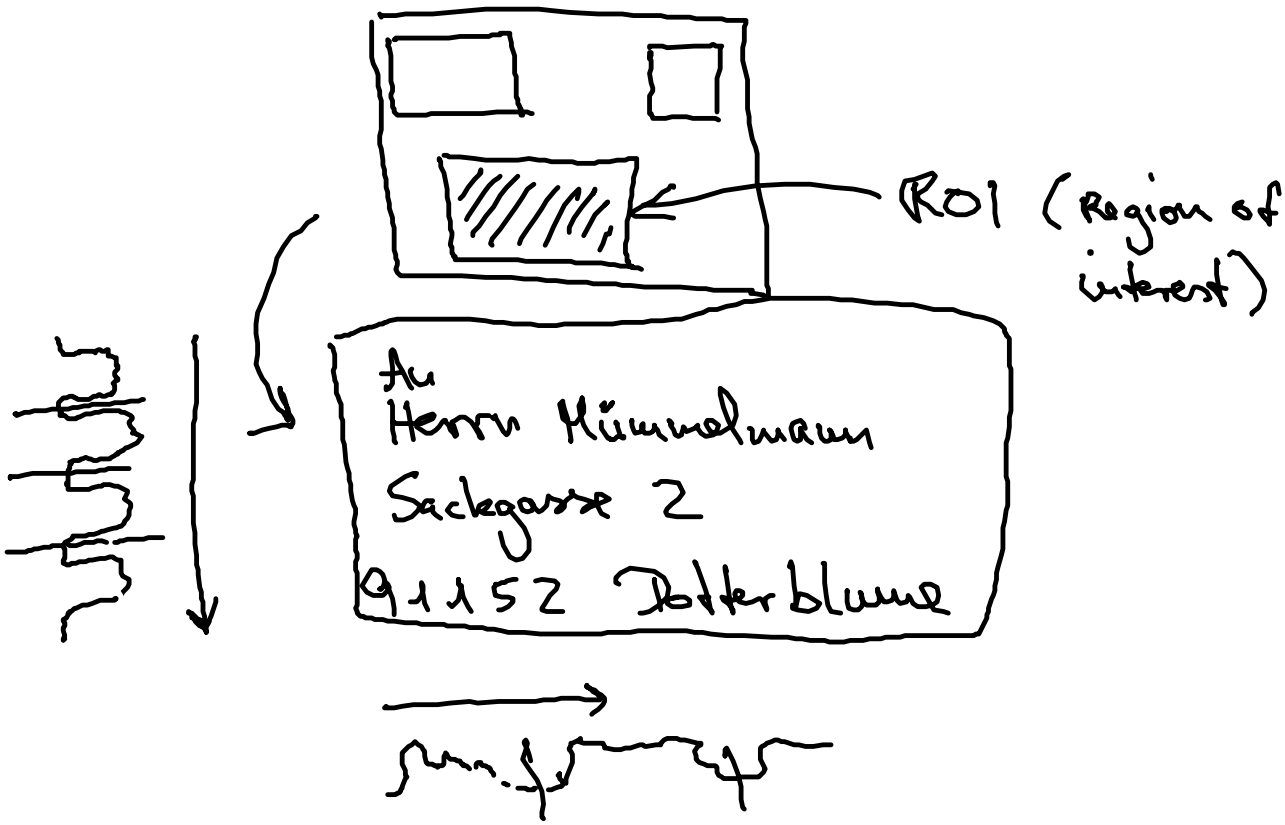



„Graubild"

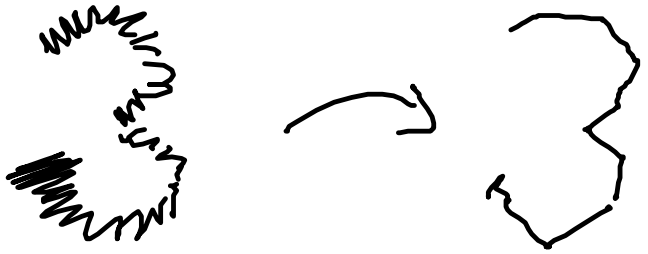
- Binarisierung:




- Textur Analyse:



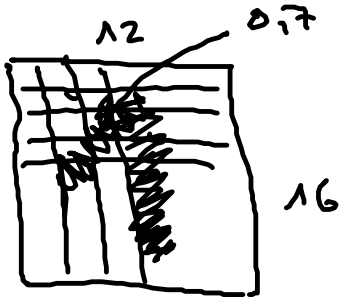

- Skelettierung

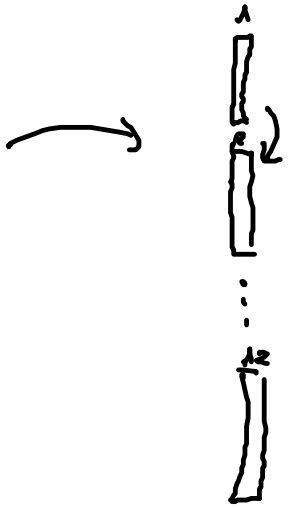


gegeben: vorverarbeitete Ziffern $\vec{x}_1, \dots, \vec{x}_n$

Trainingsdatenbank

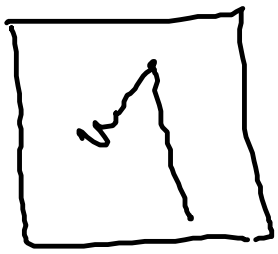
$\vec{x}_1$




192-dim Vektor

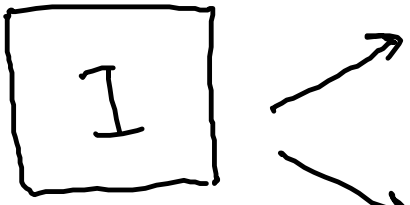
$\vec{x}_1$



Merkmale extrahieren

- # Linienzüge
- # Kreuzungen
- Winkel
- Abstände
- Längen
- Schwerpunkt
- ⋮

9 ↑ 6 ↓



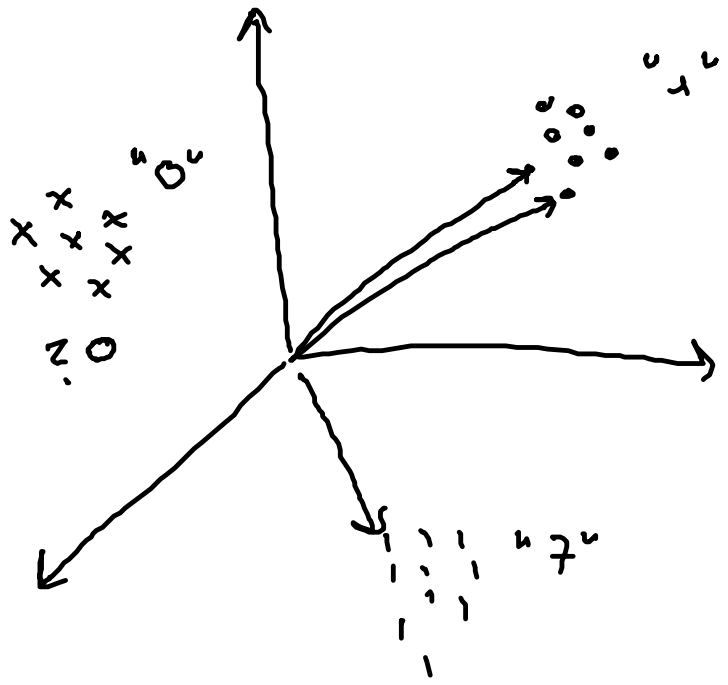

Rohdaten $\Rightarrow \vec{x} \in \mathbb{R}^n$

Merkmale $\Rightarrow \vec{x} \in \mathbb{R}^m$

$m \ll n$

Um ein intelligentes System zu bauen benötigen wir bekannte Daten (Trainingsmenge) inklusive des Klassentyps.

d.h. $(\vec{x}_1, \text{"0"}), (\vec{x}_2, \text{"5"}), \ldots$




Raum der Daten

0, 1, ..., 9

# k-nächste Nachbarn

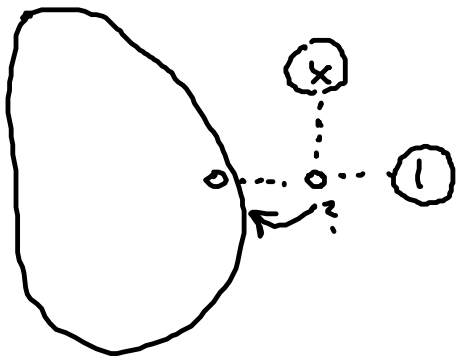
gesucht ist die Klassenzugehörigkeit von $\vec{y}$

wir betrachten alle $\vec{x}_1, \ldots, \vec{x}_n$ und berechnen den Abstand zu $\vec{y}$

$k = 3$


Klasse von $\vec{y}$ = Klasse vom nächsten Nachbarn

= k-nächste Nachbarn (Mehrheitsentscheid)

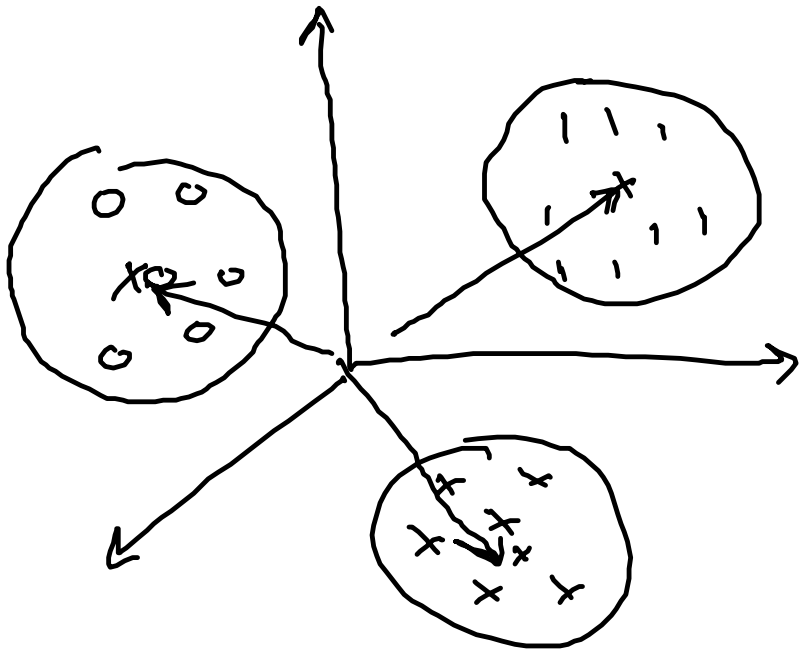


Nachteil: Laufzeit und Qualität abhängig von den Daten

## Clustering

„nicht überwachtes Lernen"

IDEE: Daten werden zunächst nach Ähnlichkeiten zusammengefügt und anschließend entschieden, welche Klasse damit repräsentiert wird



# LBG-Algorithmus (k-means)

Daten: $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^m$ gehören zu k Klassen

gesucht sind k Prototypen $(\vec{p}_1, \ldots, \vec{p}_k)$

↳ (es ist oft besser die # Prototypen $> k$ zu wählen)

0) Initialisierung:

setze $\vec{p}_1, \ldots, \vec{p}_k$ zufällig

(auf k zufällige Daten verteilen)

1) Expection-Schritt

setze $\vec{x}_i \in$ Klasse(j), falls

$\vec{x}_i$ am nächsten zu $\vec{p}_j$ liegt

2) Maximization-Schritt

setze $\vec{p}_i$ auf den Schwerpunkt

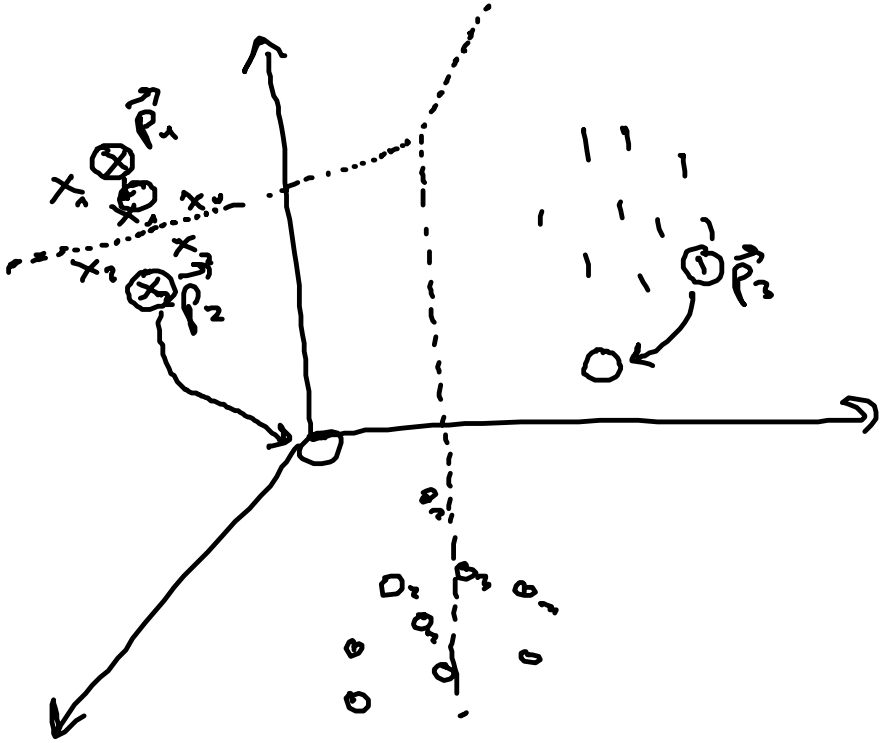
der Klasse(i) für $i = 1 \ldots k$

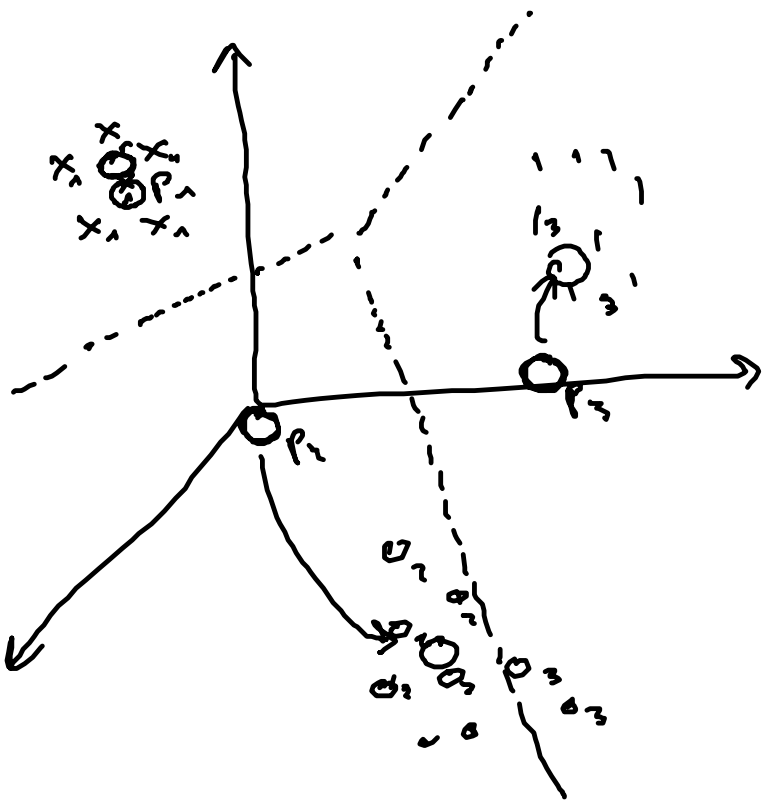
3) Abbruchkriterium

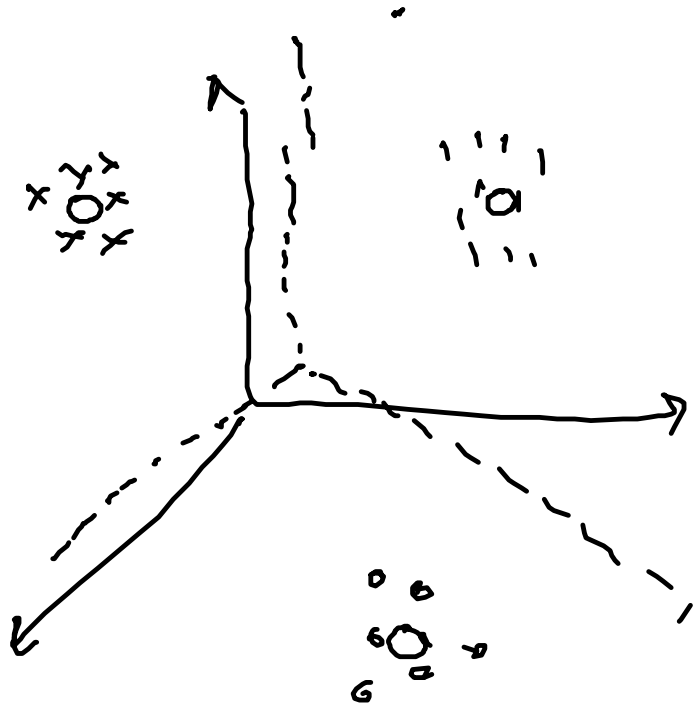
gehe zu (1) bis sich nichts

mehr ändert

$\vec{p}_1, \vec{p}_2, \vec{p}_3$

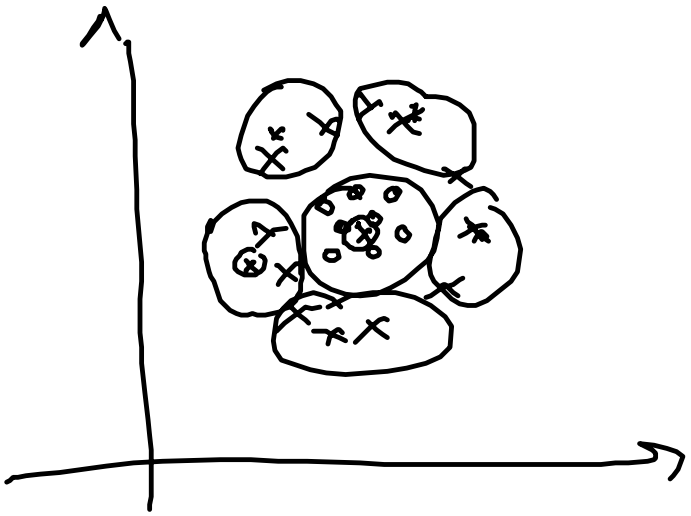






den wünschen wir uns

## Probleme



4) Klassen der Prototypen bestimmen

→ Mehrheitsentscheid

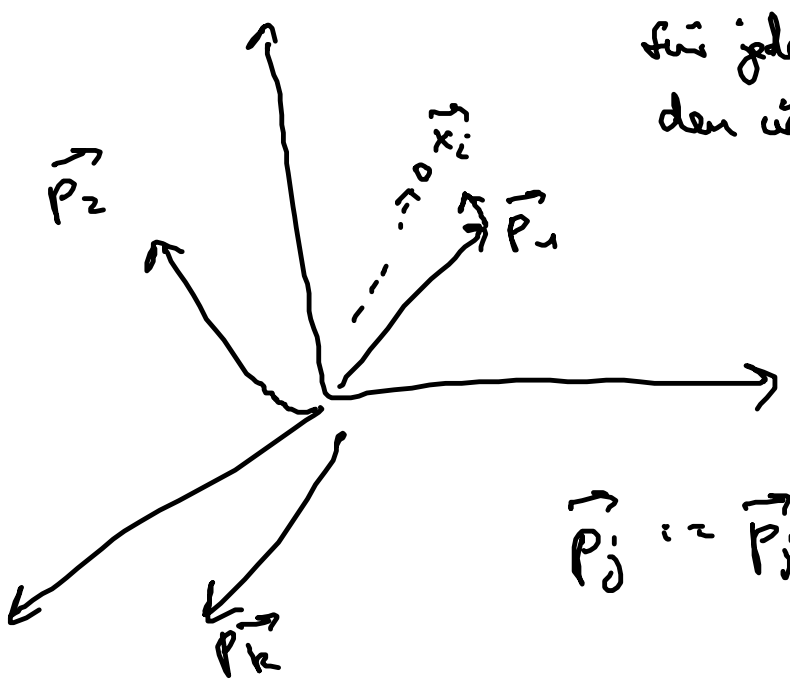
$\vec{p}_1, \ldots, \vec{p}_k$ sind „gelabelt"

Wenn wir nun ein Anfrage $\vec{y}$ haben, berechnen wir die Abstände zu den Prototypen und wählen den nächsten.

→ viel Zeit wird fürs Lernen benötigt

aber die Erkennung ist ausschlaggebend wesentlich schneller als k-means.

Online - Clustering




für jeden $\vec{x}_i$ justiere den nächsten Prototypen

$$\vec{p}_j := \vec{p}_j + \gamma\,(\vec{x}_i - \vec{p}_j)$$

Lernkonstante $\longrightarrow$ $\gamma \ll 1$

## Zwei Varianten von k-means:

„variables k-means"

Daten: $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^m$ gehören zu k Klassen gesucht l Prototypen $(\vec{p}_1, \ldots, \vec{p}_l)$ und l mit bestmöglicher Erkennungsrate (Qualität versus Effizenz)

$l = k - 1$

while ( qualitätsunzureichend) {

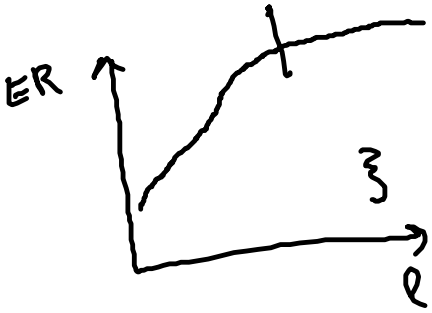
$l$++ ;

# Cluster

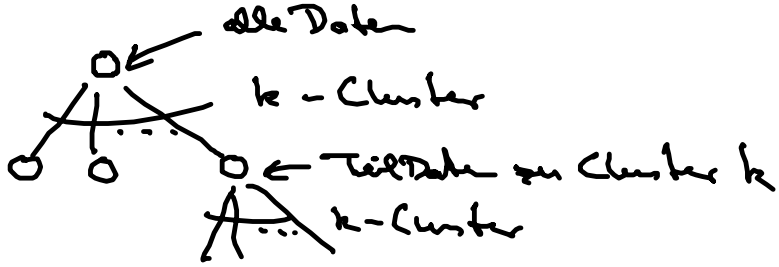
$(\vec{p}_1, \dots \vec{p}_l) \leftarrow$ k-means $(\vec{x}_1, \dots, \vec{x}_n, l)$

ER $\leftarrow$ prüfe $\vec{x}_1, \dots, \vec{x}_n$ mit $\vec{p}_1, \dots, \vec{p}_l$

wenn ER sich nicht mehr signifikant ändert $\Rightarrow$ qualitätsunzureichend




"rekursives k-means"




Daten : $\vec{x}_1, \dots, \vec{x}_n \in \mathbb{R}^\epsilon$

Tiefe $l$, $t$ (Tiefe)

k-means_rec ( $\vec{x}_1 \dots, \vec{x}_n, l, t$)

if ($t$==0)

return

(0)
(1)
(2)
(3)

k-means

for $i=1$ to $l$

k_means_rec(Daten von $\vec{p}_i$, $l$, $t-1$)