知的情報処理

12. 仲間を集めるクラスタリング

櫻井彰人
慶應義塾大学理工学部

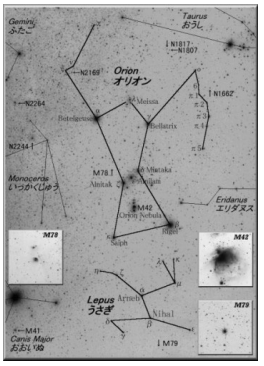
クラスタとは

- クラスタとは cluster のこと。
- cluster とは房のこと。
  - 葡萄の房
- 小さいものが複数個集まって、大きな一個の塊を構成するとき、この塊をクラスタという
  - あの、忌まわしきクラスタ爆弾の「クラスタ」もこれ

クラスタリング・クラスタ分析とは?

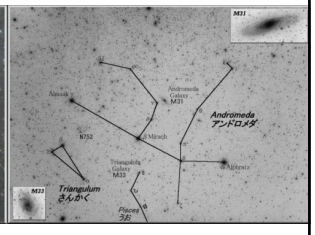
- クラスタ: 対象(データ)の集合
  - 同一クラスタ内では、似ている
  - 他のクラスタの対象とは似ていない
- クラスタ分析
  - 対象の集合を(複数個の)クラスタに集める
- クラスタリングには、普通、正解がない
  - 葡萄の房はつながっているか否かでわかるが、「似ている」か否かの判断には、多くの視点があるし、程度問題でもあるから
- 正解があっても、与えられていない or 人知には知りえない
- どんなときに使うか
  - スタンドアロンとして: 対象(データ)の分布に関する知見を得るため
  - 他のアルゴリズムの前処理として

星座: クラスタリングか?



http://homepage3.nifty.com/chokainomori-ao/gallery002.htm

星座

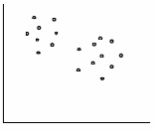


http://homepage3.nifty.com/chokainomori-ao/gallery002.htm

クラスタリングの評価: 均質性と分離性

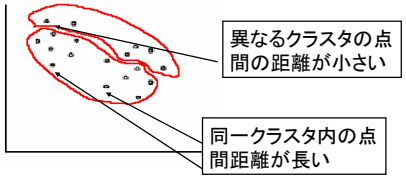
- **均質性:** 同一クラスタ内の要素は互いに類似している
- **分離性:** 異なるクラスタの要素は互いに異なっている
- …クラスタリングは容易ではない

こうしたデータが与えられたとき、クラスタリングアルゴリズムは2つのクラスタを見出す。次のように。



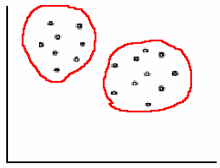
悪いクラスタリング

このクラスタリングは、均質性も分離性も満たしていない



よいクラスタリング

このクラスタリングなら、均質性と分離性を満たしている



よいクラスタリングとは?

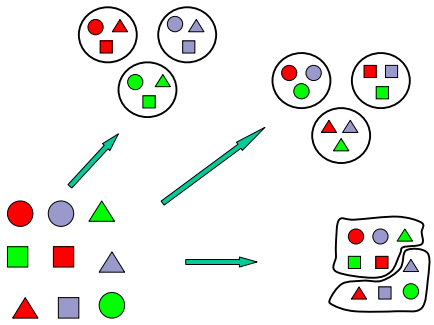
- よいクラスタ分析手法はよいクラスタを作る(はず)。よいクラスタとは
  - クラスタ内の類似性は高い
  - クラスタ間の類似性は低い
- クラスタリングの結果のよさは、用いる類似性尺度と手法のよさによる.
- クラスタリング手法のよさは、「隠れた」パターン(規則性)が発見できる能力で測ることができよう.

- なお、クラスタリングというのは、主観的であることに注意されたい
  - 正解がない。データが不足していて正解が決定できない場合だけでなく、本当に正解がない場合(見方によって変わる場合)がほとんど

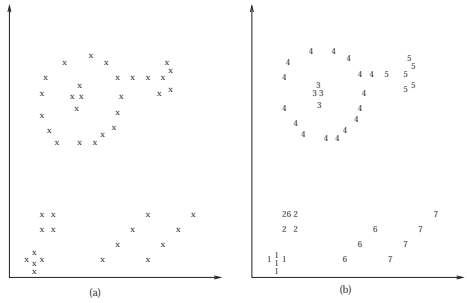
クラスタリングに対する要件

- スケーラビリティがある
- 異なった属性が扱える
- どんな形のクラスタでも発見できる
- パラメータを決めるために必要とされる領域知識の少ない
- ノイズや外れ値 (outlier) があってもよい
- データの入力順序に依存しない
- 次元が高くても大丈夫
- ユーザが制約を与えることができる
- 他の手法と組み合わせることができる

そうはいっても難しい



意地悪な例ならいくらでも考えられる



意地悪ではない: 軸のスケールを変えると

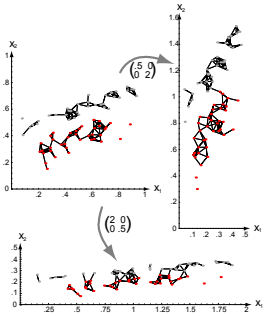


FIGURE 10.8. From: Richard O. Duda, Peter, E. Hart, and David G. Stork, Pattern Classification. Copyright c 2001 by John Wiley & Sons, Inc.

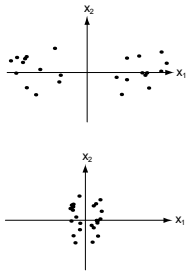


FIGURE 10.9. From: Richard O. Duda, Peter, E. Hart, and David G. Stork, Pattern Classification. Copyright c 2001 by John Wiley & Sons, Inc.

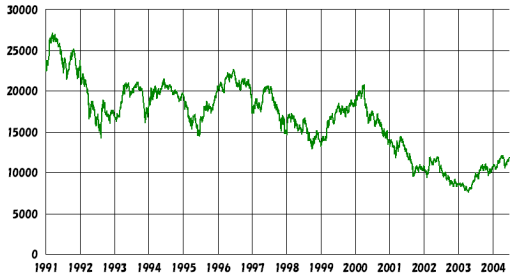
属性(特徴)選択とスケーリング

- 適切な属性(特徴)とスケーリングを選ぶことが必要
- 適切な属性(特徴)を選ぶことは難しい
  - クラスタリングの正解がわかっていれば、いろいろ工夫ができる。
  - しかし、そうでなければ(正解がわかっていないのが普通)コンピュータにはできない、というのが正しい
  - その分野の専門家しかできなかろう。
- スケーリング、より正確には、距離の定義が重要
  - 適切な距離が分かるということは、しかし、適切なクラスタリングが分かるということ
  - 従って、これも、難しい

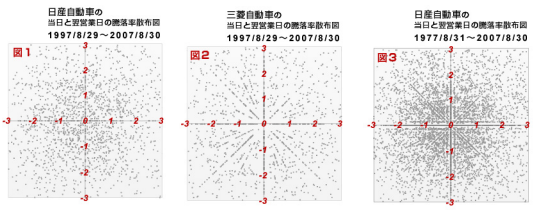
別の重要な(or 根源的な)問題

- そもそも、クラスタというのは存在するのか?
- 人間に「クラスタ」「仲間」「繰り返し性」が見えれば、クラスタが存在すると考えていいのか?
- 見えなければいけないのか?
  - 高次元空間のクラスタは見えないぞ

クラスタ(パターン)はあるか?



株価: クラスタはあるか?



株式市場に現れる謎の"羅針盤" 騰落率の放射模様は株価予測に役立つ?

http://www.nikkei.co.jp/needs/analysis/07/a070906.html

株価: クラスタはあるか?(もと)

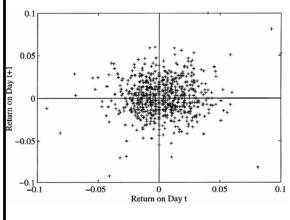


Figure 1. From Brealey and Myers-Weyerhaeuser. This is a reproduction of Figure 13-2 from Brealey and Myers (1991). Compare this to Figure 2 below. The horizontal axis shows return on Weyerhaeuser stock on a given day; the vertical axis shows return on the next day. Pluses (+) are used to symbolize the data. The data span the period January 2, 1986 to December 30, 1988. Out of 758 points, 6 lie outside the bounds of the plot.

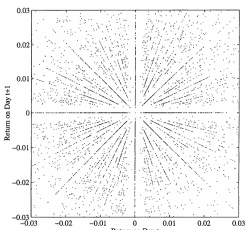


Figure 2. Compass Rose-Weyerhaeuser. This figure is identical to Figure 1, except that we extend the time series, symbolize the data with dots instead of pluses, and adjust the scale. The compass rose pattern appears clearly. The horizontal axis shows return on Weyerhaeuser stock on a given day; the vertical axis shows return on the next day. The data span the period December 6, 1963 to December 31, 1993. Out of 7566 points, 1212 lie outside the bounds of the plot.

Crack, Timothy F., and Olivier Ledoit. 1996. "Robust Structure Without Predictability: The 'Compass Rose' Pattern of the Stock Market," Journal of Finance, 51: 751-762.

株価: クラスタはあるか？（もと）

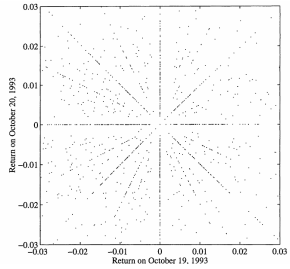


Figure 3. Compass Rose-2123 NYSE stocks. This plot shows one point only for each of 2123 NYSE stocks on a randomly selected pair of consecutive trading days; the structure is cross-sectionally coherent. The horizontal axis shows return on October 19, 1993; the vertical axis shows return on October 20, 1993. Out of 2123 points, 469 lie outside the bounds of the plot.

Crack, Timothy F., and Olivier Ledoit. 1996. "Robust Structure Without Predictability: The 'Compass Rose' Pattern of the Stock Market," Journal of Finance, 51: 751-762.

音楽: クラスタリングのある結果

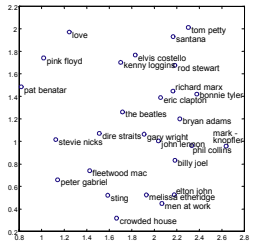
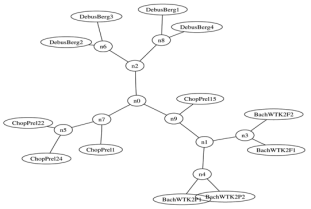


Figure 1: Artists embedded in a 2-D space. This is a small portion of the full space derived from the Erdösmeasure.

D. Ellis, B. Whitman, A. Berenzweig, S. Lawrence. The Quest for Ground Truth in Musical Artist Similarity *Proc. ISMIR-02*, Paris, October 2002.



R. Cilibrasi, P.M.B. Vitanyi, R. de Wolf, Algorithmic clustering of music based on string compression, Computer Music J., 28:4 (2004), 49-67.

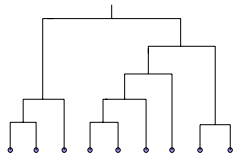
クラスタリング技法

- **階層的方法 hierarchical:** クラスタ間の関係が樹木状になるように構成する方法。根は全体。葉は（通常は）一要素からなるクラスタ。枝分かれが、クラスタの分割（逆にみるとクラスタの融合）に相当する
  - **凝集法 agglomerative:** 各要素がそれぞれ一つのクラスタを構成する状態から開始。近いクラスタを順々にまとめて大きなクラスタとしていく
  - **分割法 divisive: 全体を一つのクラスタとして開始。より小さなクラスタに分割していく**
- 非階層的方法 non- **hierarchical: 階層的でないもの**
  - モデル法: 統計的モデル（クラスタ＝分布）を考え、尤度最大化
    - 実際のアルゴリズムとしては、EMアルゴリズムが著名
      - Mathematica の ClusterAnalysis はこれに属する
  - 分割法: 分割の良否の評価関数の最適化問題と考える
    - k-means法
    - グラフ理論に基づく方法、spectral clustering等

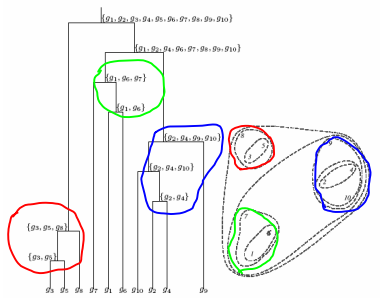
注: Dendrogram 樹形図

対象（データ）集合の融合（逆向きに見れば分割）の様子を2進木の形に表したもの。ただし、例えば下図であれば、縦軸が融合時の非類似度を表すようにする.

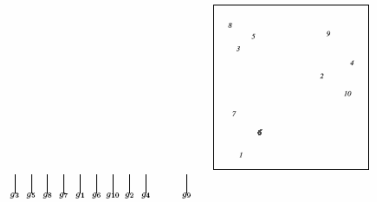
必要と考えるクラスタ数の枝があるところで、切断すれば、任意数クラスタへのクラスタリングが得られる.



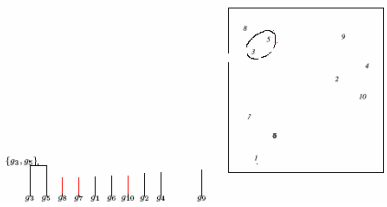
階層的クラスタリング



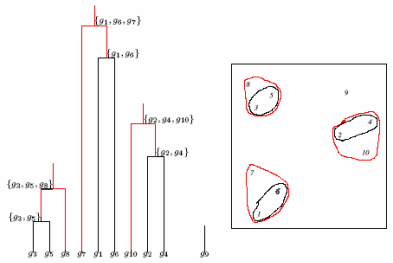
階層的クラスタリング: 凝集法



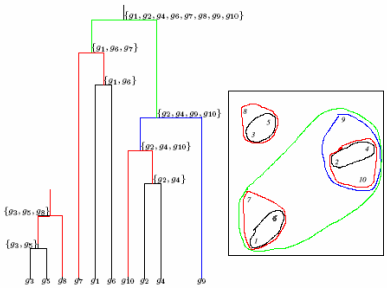
階層的クラスタリング: 凝集法



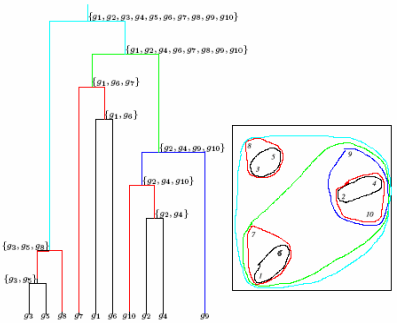
階層的クラスタリング: 凝集法



階層的クラスタリング: 凝集法

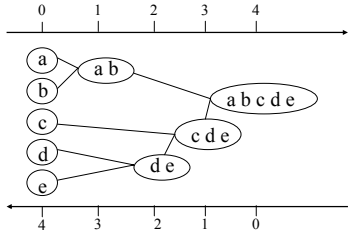


階層的クラスタリング: 凝集法



階層的クラスタリング

- 距離行列をクラスタリングの基準に用いる．この方法では、クラスタ数 ***k*** を入力パラメータとしない。



凝集法: アルゴリズム

本アルゴリズムは、点の個数 n と nxn の距離行列（各点間の距離を要素とする。対称行列となる）を入力とする

1. 凝集法 (***d***, *n*)
2. 各要素からなるクラスタを作る。*n* 個とする
3. 各クラスタが一つの頂点（葉）であるような木 ***T*** を作る
4. **while** 2個以上のクラスタがある
5. 最も**近い**二つのクラスタを見つける。C_1 と C_2 とする
6. C_1 と C_2 を一緒にして新クラスタ ***C*** を作る。要素数 $|C_1| + |C_2|$
7. ***C*** **から他のすべてのクラスタへの距離を計算する**
8. ***T*** に新たな頂点 ***C*** を付加し、頂点 C_1 及び C_2 に結ぶ
9. クラスタ間の距離を保持する表 ***d*** から C_1 と C_2 に関する情報を削除する
10. 新クラスタ ***C*** と他のクラスタとの距離を、表 ***d*** に付け加える）
11. ***T*** を値として終了

距離の定義が異なれば、異なるクラスタ・クラスタ構造が得られる

クラスタ間距離

- 最小（最近）距離:
$$d_{\min}(C_i, C_j) = \min_{p \in C_i, p' \in C_j} d(p, p')$$
- 最大（最遠）距離:
$$d_{\max}(C_i, C_j) = \max_{p \in C_i, p' \in C_j} d(p, p')$$
- 平均距離:
$$d_{average}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{p' \in C_j} d(p, p')$$
- 中心（重心）距離:
$$d_{mean}(C_i, C_j) = d(m_i, m_j)$$
- Ward距離
$$d_{Ward}(C_i, C_j) = e(C_i \cup C_j) - e(C_i) - e(C_j)$$
$$e(C) = \sum_{x \in C} (d(x, c))^2 \quad (c \text{は} C \text{の中心})$$

クラスタ間の距離：再掲

- $d_{min}(C, C^*)$ = min $d(x,y)$
  全要素 $x \in C$ と $y \in C^*$
  - 二つのクラス間距離は、すべての要素対の距離の**最小**値
- $d_{max}(C, C^*)$ = max $d(x,y)$
  全要素 $x \in C$ と $y \in C^*$
  - 二つのクラス間距離は、すべての要素対の距離の**最大**値
- $d_{avg}(C, C^*) = (1 / |C^*||C|) \sum d(x,y)$
  全要素 $x \in C$ と $y \in C^*$
  - 二つのクラス間距離は、すべての要素対の距離の**平均**値

距離と類似度と非類似度

- 類似度（similarity）: 大きい = 距離が近い
  - ちょっと変わった例: 文字列の編集距離(edit distance)
- 非類似度（dissimilarity）: 小さい = 距離が近い

点間の距離

- 数学でポピュラーなのはミンコフスキー距離:
$$d(i,j) = \sqrt[q]{(|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + ... + |x_{ip} - x_{jp}|^q)}$$
  ただし $i = (x_{i1}, x_{i2}, ..., x_{ip})$ と $j = (x_{j1}, x_{j2}, ..., x_{jp})$ は p-次元データ、q はある正数（通常は整数）
- 特別な場合（コンピュータではこっちがポピュラー）: $q = 1$ のとき, マンハッタン距離と呼ばれる
$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + .. + |x_{ip} - x_{jp}|$$

点間の距離

- $q = 2$ ならばユークリッド距離:
$$d(i,j) = \sqrt{(|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + ... + |x_{ip} - x_{jp}|^2)}$$
  - 距離の定義
    - $d(i,j) \geq 0$
    - $d(i,i) = 0$
    - $d(i,j) = d(j,i)$
    - $d(i,j) \leq d(i,k) + d(k,j)$
- 他には、荷重付きの距離、Pearson の積率相関係数等々

補足: 距離らしくない距離

- Compression distance（圧縮距離）というものもある
- データを圧縮するアルゴリズムに基づく
  - 例えば、テキストAを、テキストB（の統計情報）を知って圧縮すると、知らないで圧縮するより短くできたとする。そうすると、テキストAとテキストBは似ている、すなわち、距離はかなり近いと考えられる。
  - これは、数学的な奥行きが非常に深い概念である

階層的クラスタリング: 使う「距離」

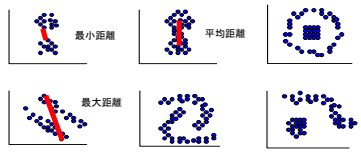
- single-link クラスタリング (または connectedness/minimum 法) : 二つのクラスタ間の距離を、要素間の最小距離で定義する. 類似性という言葉で表現するなら、二つのクラスタの類似性は、最もよく似た要素対間の類似性であると定義することになる(ちょっと苦しいぞ).
- complete-link クラスタリング (または the diameter/maximum 法): 二つのクラスタ間の距離を、要素間の最大距離で定義する. 類似性という言葉で表現するなら、二つのクラスタの類似性は、最も異なった要素対間の類似性であると定義することになる(ちょっと苦しいぞ).
- average-link クラスタリング : 二つのクラスタ間の距離を、全要素間の距離の平均値で定義する.

クラスタ間の距離: 補足

■single-link クラスタリング (または connectedness/minimum 法) : 二つのクラスタ間の距離を、要素間の最小距離で定義する. 類似性という言葉で表現するなら、二つのクラスタの類似性は、最もよく似た要素対間の類似性であると定義することになる(ちょっと苦しいぞ).

■complete-link クラスタリング (または the diameter/maximum 法):二つのクラスタ間の距離を、要素間の最大距離で定義する. 類似性という言葉で表現するなら、二つのクラスタの類似性は、最も異なった要素対間の類似性であると定義することになる(ちょっと苦しいぞ).

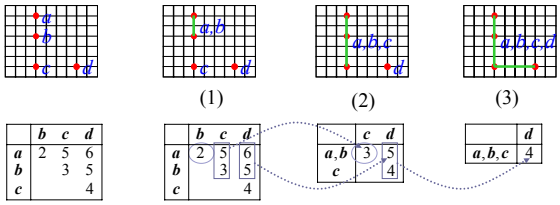
■average-link クラスタリング : 二つのクラスタ間の距離を、全要素間の距離の平均値で定義する.



❑Single-Link / 最近隣
❑Complete-Link / 最遠隣(?)
❑Average 平均.
❑ Centroids 中心.

Single-Link クラスタリング

ユークリッド距離



	b	c	d
a	2	5	6
b		3	5
c			4

	b	c	d
a	2	5	6
b		3	5
c			4

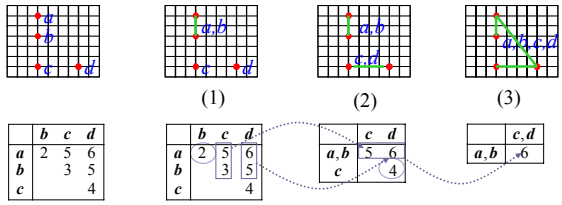
	c	d
a,b	3	5
c		4

	d
a,b,c	4

距離行列

Complete-Link クラスタリング

ユークリッド距離



	b	c	d
a	2	5	6
b		3	5
c			4

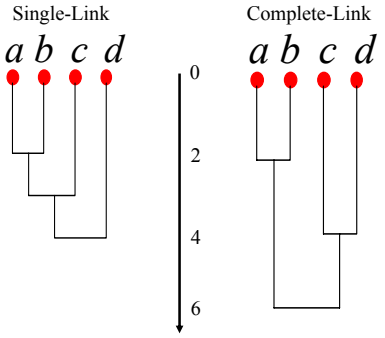
	b	c	d
a	2	5	6
b		3	5
c			4

	c	d
a,b	5	6
c		4

	c,d
a,b	6

距離行列

デンドログラム(樹形図)で比較



距離データの構造

- 非対称距離

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

- 対称距離

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

k-means クラスタリング: 定式化

- **入力**: n 個の点からなる集合 $\boldsymbol{V}$, パラメータ k
- **出力**: k 個の点 (クラスタの中心) からなる集合 $\boldsymbol{X}$ で、すべての可能な $\boldsymbol{X}$ のなかで、二乗歪（squared error distortion）$d(\boldsymbol{V},\boldsymbol{X})$ が最小のもの

二乗歪

- 点 v と点集合 X が与えられたとき, v から X への距離

$$d(v, X)$$

は、v から X の最近接点へのユークリッド距離であるとする.

- n 個の点からなる集合 $V=\{v_1 \ldots v_n\}$ と点集合 X が与えられたとき, 二乗歪 **Squared Error Distortion** は次のように定義される

$$d(V,X) = \sum_{1 \le i \le n} d(v_i, X)^2 \ / \ n$$

1-Means クラスタリング: やさしい場合

- **Input**: n 個の点からなる集合 $\boldsymbol{V}$
- **Output**: ある一個 の点 $\boldsymbol{x}$ (クラスタ中心) で、$\boldsymbol{x}$ のすべての可能な選び方のなかで、二乗歪 $d(\boldsymbol{V},\boldsymbol{x})$ が最小となるもの

1-Means クラスタリングはやさしい問題.

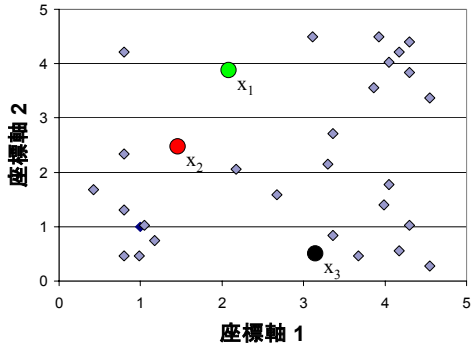
しかし, 中心（クラスタ数）が2個以上となると、とたんに難しい問題 (NP-完全) となる.

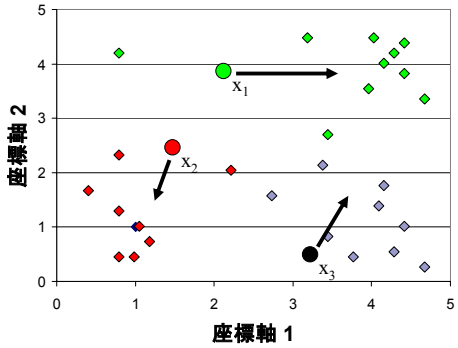
効率のよい発見的方法の一つに Lloyd アルゴリズムがある

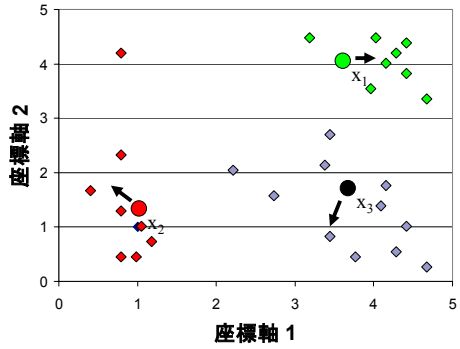
K-Means クラスタリング: Lloyd アルゴリズム

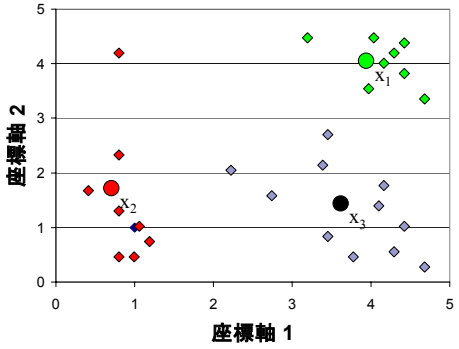
1. Lloyd アルゴリズム
2. 任意に k 個のクラスタ中心を選ぶ
3. **while** クラスタ中心が変更された
4. 各データ点をクラスタ C_i に割り当てる
   割り当てるクラスタは、最も近いクラスタ中心のクラスタ
   $(1 \le i \le k)$
5. すべての点を割り当て終わったら,
   各クラスタの重心を新たなクラスタ中心とする
   すなわち、新たなクラスタ中心は、クラスタ C それぞれに
   $\sum v \ / \ |C|$
   全 $v \in C$

*このアルゴリズムは局所解に陥ることがある.









クラスタリング demo

http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html

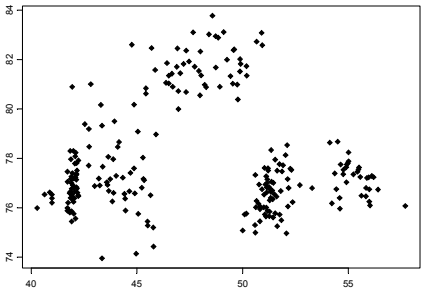
http://know-center.tugraz.at/forschung/wissenserschliessung/downloads_demos/fuzzy_k_means_and_k_means_clustering_demo

hacdemo.jnlp 必要なモジュールは web からダウンロードされます

kmeansdemo.jnlp 同上

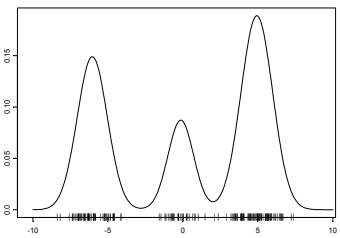
http://www.cs.washington.edu/research/imagedatabase/demo/kmcluster/

モデル法の例: 統計的な枠組み



5または6個のクラスタがありそう(?)
それを見出し、各点を各クラスタに分ける

モデル法:

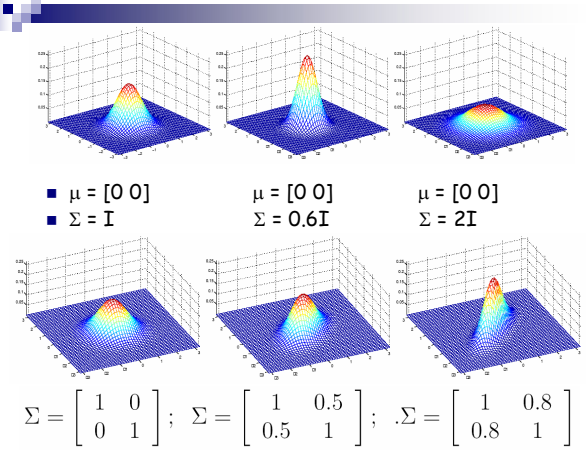


前提をおく:

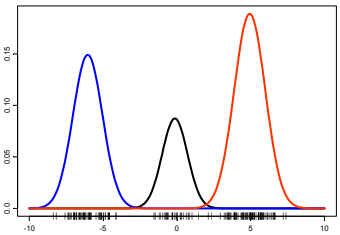
- 観測値はある混合分布からのサンプルである

$$p(x) = \sum \pi_g p_g(x)$$

- 各クラスタは混合分布の個々の分布に対応する



モデル法



フィッティング:
π_g と $p_g(x)$ のパラメータ(平均、分散等)を推定する

モデル法

ガウス混合分布(ガウス分布の重み付き線形和)をフィッティングする
各xはいずれかの分布(クラスタ)に属すると考える

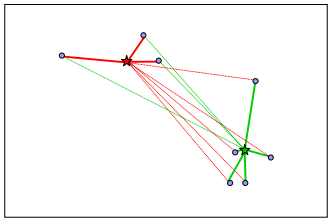
- 対数尤度を最大化する π_g、$p_g()$ を求める。

$$\log L(\pi_1,\ldots,\pi_G, p_1,\ldots,p_G \mid x_1,\ldots,x_n) = \log \prod_{i=1}^{n} \sum_{g=1}^{G} \pi_g p_g(x_i)$$

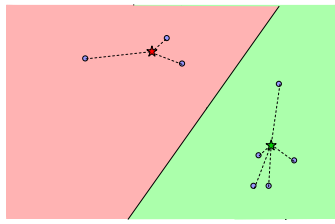
$$= \sum_{i=1}^{n} \log\left(\sum_{g=1}^{G} \pi_g p_g(x_i) \right)$$

- EMアルゴリズムを用いる
  - 個々の点x_iが個々の分布p_gに属する確率を計算する
    - $\pi_g p_g(x_i)$/**正規化定数 を計算する**
  - これらの確率をもとに、尤度を最大化する
    - π_g , p_g **のパラメータを最尤推定する**
  - という反復計算。一般には、局所最適解に収束。

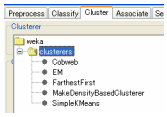
EM-アルゴリズムの実行イメージ



K-Means クラスタリングとの比較



モデル法の長所と短所



- 長所
  - 統計的基盤、理論的に健全
    - 信頼性等が計算可能。検定もできる
- 短所
  - 計算が大変
  - 世の中のものに、「分布」を予め仮定するのが大変
    - 理論的妥当性に疑問がある場合が多い

再: クラスタリング方法の分類

- 階層的クラスタリング:
  - ある基準に従って、階層的に対象(データ)の集合を分割していく
- 非階層的クラスタリング
  - モデル法: 各クラスタに(統計的)モデルを仮定し、データに最もあうように、パラメータを決定する
  - 分割的クラスタリング: いくつかの分割を作成し、ある基準に従い評価する
  - 濃度に基づく: 濃度・密度・結合度に基づく
  - グリッド法: 多層の粒度構造に基づく

では、文書分類はできるか？

- 実は、できない！
- クラスタリングするには、各文書をなんらかの数値ベクトルで表現しないといけない。
- どうやる？

文書クラスタリング

- 普通は次のように行う
  - 文書を単語に分ける。
    - 例: 明日は晴れでしょう。→ 明日+は+晴れ+で+しょう+。
  - どこにでもある単語（ at, is, a,… ）ははずす
    - なぜでしょうか？
  - 余りにも特殊な単語ははずす
    - なぜでしょうか？
  - 文書ごとに文書の特徴ベクトルを作る。各単語に番号を振り、その番号を場所とするベクトルである。各要素は、当該文書中に出現する当該単語の回数（または、0/1）
  - 文書を点、この特徴ベクトルをその点の座標と考えて、クラスタリングを行う

文書クラスタリング: 困ること

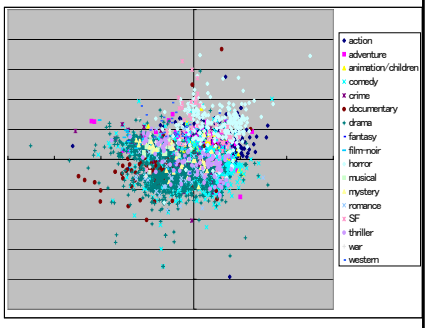
- ベクトルの次元が極めて高くなる
  - 短い文書（実験を行う newsgroup への投稿記事でも）かつ少ない文書数でもすぐ数千、数万になる。
  - すなわち、数千次元、数万次元。
  - 人間が図で見れるのは多くて3次元
  - 実は、「高次元」というのはいろいろな問題を引き起こす要因である。
    - 計算に時間がかかる
    - （いくらメモリが 1Gとか2Gとかいっても）メモリ不足
    - 実は、ベクトルの距離の定義が難しくなる。ユークリッド距離は適さなくなってしまう

次元の呪い: これは実問題の問題

- Curse of dimentionality という
- コンビニの売り上げデータを考えよう
- 一回の売り上げを一個のベクトルであらわす
  - Excel の表なら、横1行が一回の売り上げ。
- ベクトルの各要素は、各商品に対応する
  - Excel の表なら、縦1行がある商品一個に対応
- ある一年を考えるだけでも、数千・数万行が必要。
  - 次元が数千・数万のベクトル
- 数年分を管理しようと思えば、数万・数十万行
  - 次元が数万・数十万のベクトル

高次元の実データ例 MovieLens

- 全ユーザ
- 数量化3類適用後



高次元の実データ例 MovieLens

- 女性、18歳～24歳
- 数量化3類適用後