

データサイエンス

第8回

~クラスター分析~

情報理工学系研究科
創造情報学専攻
中山 英樹

本日の内容

- レポート課題（補足）
- クラスター分析
 - 階層的クラスタリング
 - 非階層的クラスタリング

第一回レポート課題（回帰）

- Wine quality prediction dataset

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.

- 物理化学的特徴からワインの質（10段階）を予測（回帰）

- 説明変数

- 0 - red or white
- 1 - fixed acidity
- 2 - volatile acidity
- 3 - citric acid
- 4 - residual sugar
- 5 - chlorides
- 6 - free sulfur dioxide
- 7 - total sulfur dioxide
- 8 - density
- 9 - pH
- 10 - sulphates
- 11 - alcohol

- 目的変数

- 12 - quality (score between 0 and 10)



コンペティション形式

- ランキング

- スコアリングサーバ: <http://www.nlab.ci.i.u-tokyo.ac.jp/~nakayama/ds13/report1/index.php>
 - テストサンプルの推定結果を一行ずつ記載したテキストファイルを提出
 - ユーザ名を忘れずに（実名である必要はない。ただし同じユーザ名をずっと使用すること。複数の方法を比較したい場合は複数利用しても構わない。）
 - 提出した結果は上書きされる。最後のものだけ保存されるので注意。
 - 提出回数に制限はありませんが、**極端なアクセスはしないでください。**
- 現在は、提出されたテストデータ1000の推定結果のうち、所定の200サンプルでスコアリングしている
- **最終的なスコアは、締め切り(12/13)後に残りの800サンプルで算出**

課題詳細

- レポート内容
 - Wine quality predictionの問題に取り組み、以下の点を中心にA4用紙1~2枚程度にまとめよ。講義で扱っていない技術を用いても構わない。
- 1. 予測性能を向上させるための自分なりの工夫点と結果、考察（必須）
 - コードを載せる必要はない（もちろん、実装がポイントの場合は載せてかまわない）
- 2. その他、自由にデータを分析した結果（+α）
 - 学習データ数と性能、正則化パラメータの関係
 - 各特徴の寄与の分析
 - 赤ワインと白ワインを判別分析してみる …など
- 3. ここまでの講義の感想、要望など（必須）
 - フィードバックをください！
- その他、課題に関して補足・訂正がある場合はHPに掲載します

課題詳細

- 評価の方針
 - アイデアや試行錯誤の過程を重視
 - スコアが悪くても、しっかり考察してくれればOK
(もちろん良くなればプラスに評価しますが)
 - 面白い分析を期待します
- 提出先
 - 以下のアドレスへメールで提出すること（質問もこちらへ）
 - ds2013@nlab.ci.i.u-tokyo.ac.jp
 - 件名は「データサイエンスレポート課題1」
 - 氏名、学籍番号、所属を忘れずに
- 締め切り
 - 12月16日（月）
 - ただし、スコアリングサーバは13日に締め切り、最終結果発表

課題詳細

- スコアの良い人や、面白い分析をしてくれた人は18日の講義で簡単に紹介してもらおうかもしれません（未確定）
 - なるべくメールで事前に連絡&お願いします
 - みんなにとって利益になるかどうかの基準で選びます
- どうしても実装が難しい場合…（プログラミング未経験者の方など）
 - 1・2の代わりに、元論文を読み内容をまとめることでOKとします。
 - できれば挑戦して欲しいですが…

工夫できそうなところ

- 目的変数の解釈
 - 0~10の整数値。尺度は？
 - カテゴリとして扱ってみる？
 - 前処理を入れる？
- 回帰のオフセット
- カテゴリ変数の扱い（赤 or 白）
 - ダミー変数？
 - 分けてモデルを作る？
- 手法・パラメータチューニング
 - 正則化項の入れ方（リッジ回帰）
 - クロスバリデーション
- 原論文を読んでみる

クラスター分析の位置づけ

- 教師なし学習の一つ。特に、発見したい構造が**離散的**な場合
- データから何かを発見したい → 教師なし学習

説明変数	手法
量的データ(比尺度)	主成分分析、因子分析、LPP
量的データ(間隔尺度)	クラスター分析 、多次元尺度構成法、数量化Ⅳ類
質的データ	数量化Ⅲ類、対応分析

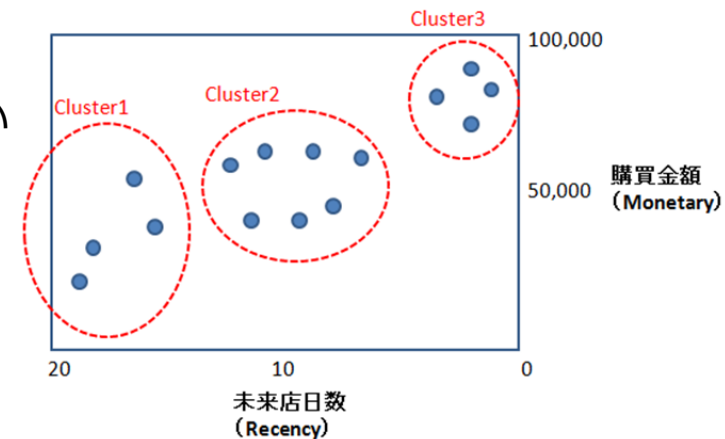
- データを使って何かを予測したい → 教師あり学習

目的変数	説明変数	手法
量的データ	量的データ	回帰分析
	質的データ	数量化Ⅰ類
質的データ	量的データ	判別分析
	質的データ	数量化Ⅱ類

クラスター分析（クラスタリング）

- データ間の類似度（距離）を定義し、
似ているものは同じまとまり（クラスタ）に、
似ていないものは異なるまとまりに分類する
 - 雑多なデータの代表点を抽出したい
 - 購買データに基づくユーザのグループ分け、ウェブ上の文書分類、etc.
 - Cluster = ブドウの房
- 目的変数なし（教師なし）の多変量解析
 - 各クラスタの意味が自動的に得られるわけではない
 - あくまである一つの視点から見た場合
- 結果の解釈が重要
 - 各クラスタの意味は後で解釈する

← 内的結合
← 外的分離



距離の定義

特徴ベクトル（説明変数）を $\mathbf{x} = (x_1, x_2, \dots, x_d)^T$ と表記する

- ユークリッド距離（L2距離）
$$\left[\sum_{k=1}^d (x_{ik} - x_{jk})^2 \right]^{1/2}$$

- ミンコフスキー距離（Lp距離）
$$\left[\sum_{k=1}^d |x_{ik} - x_{jk}|^p \right]^{1/p}$$

- マハラノビス距離
$$(\mathbf{x}_i - \bar{\mathbf{x}})^T \Sigma^{-1} (\mathbf{x}_j - \bar{\mathbf{x}})$$

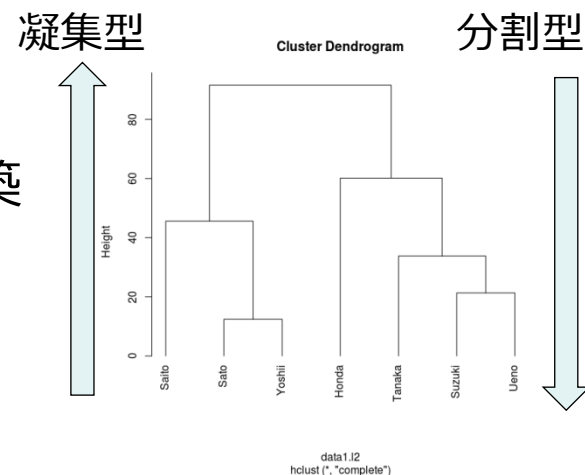
$\bar{\mathbf{x}}, \Sigma$ はそれぞれサンプルの平均・共分散行列
説明変数間に大きなスケールの差がある時に使う

- コサイン類似度
$$\frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x}_i\| \|\mathbf{x}_j\|}$$
- ...

クラスタリング手法の概要

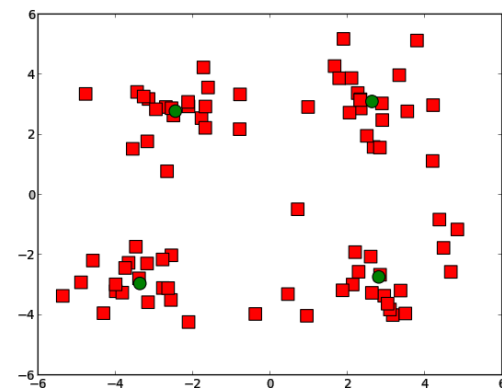
- 階層的手法（凝集型）

- データ一つ一つがクラスタである状態から出発し、ボトムアップに階層的な分類構造を構築
- 実際はそれほど使われない
 - サンプル数が多いと計算が大変 & 解釈が難しい



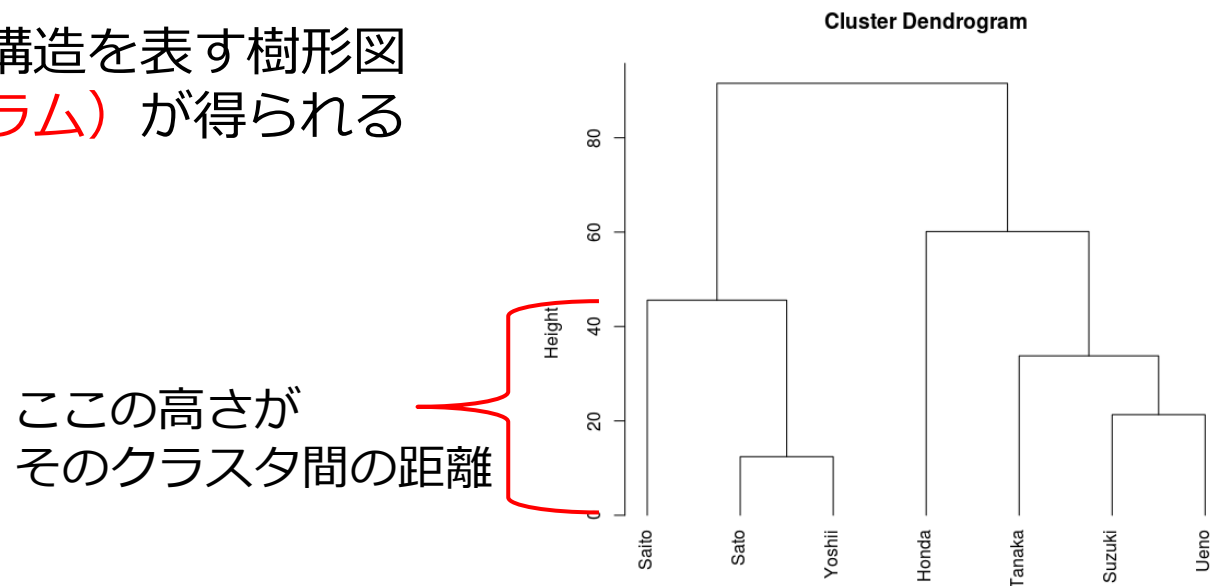
- 非階層的手法

- クラスタの良さを定義する目的関数を定義し、これを最適にする分割を探索する
- 一般的にはこちらがよく使われる



階層的クラスタリング

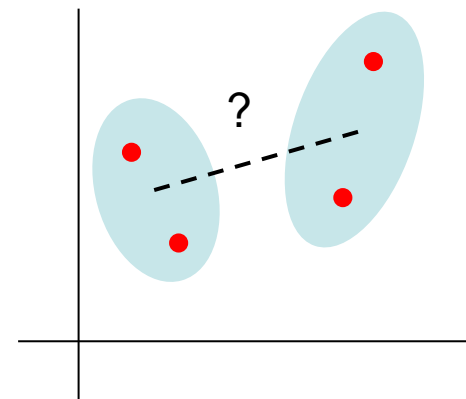
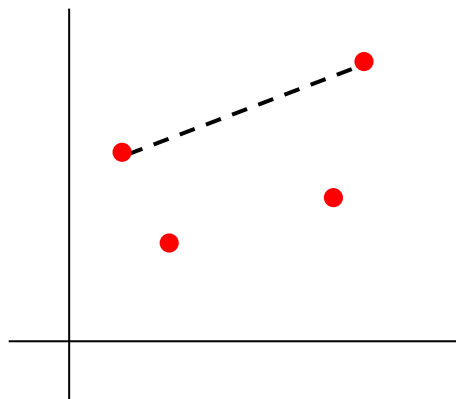
- 1. 各データを1つのクラスタとする
- 2. クラスタ間の距離（類似度）に基づいてクラスタを逐次的に併合する
- 3. 1つのクラスタになるまで併合を繰り返す
- データの階層構造を表す樹形図（デンドログラム）が得られる



data1.12
hclust(*, "complete")

階層的クラスタリングの手法

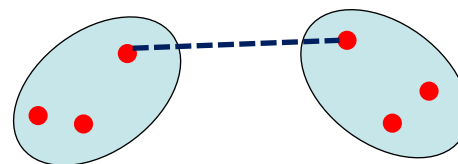
- 与えられているのはサンプル間の類似度
- クラスタ間の距離の測り方で変わる
 - 単連結法
 - 完全連結法
 - 群平均法
 - ウォード法
 - (重心法)
 - (メディアン法)



クラスタ間の距離の定義（１）

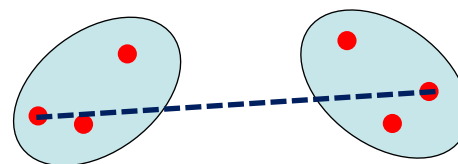
- 最短距離法（単連結法）
 - 二つのクラスタの個体のうち最も近い個体間の距離

$$d(C_1, C_2) = \min_{\mathbf{x}_1 \in C_1, \mathbf{x}_2 \in C_2} d(\mathbf{x}_1, \mathbf{x}_2)$$

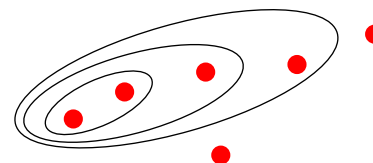


- 最長距離法（完全連結法）
 - 二つのクラスタの個体のうち最も遠い個体間の距離

$$d(C_1, C_2) = \max_{\mathbf{x}_1 \in C_1, \mathbf{x}_2 \in C_2} d(\mathbf{x}_1, \mathbf{x}_2)$$



- 計算コストが（比較的）小さい
（最小全域木）
- × 外れ値の影響を受けやすい
- × “チェイニング”が起こりやすい



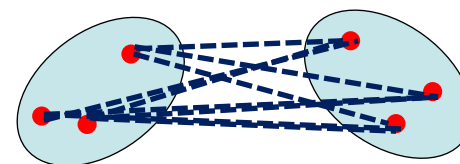
チェーン状のクラスタ

クラスター間の距離の定義（２）

- 群平均法

- 二つのクラスターのすべての
個体間の距離の平均

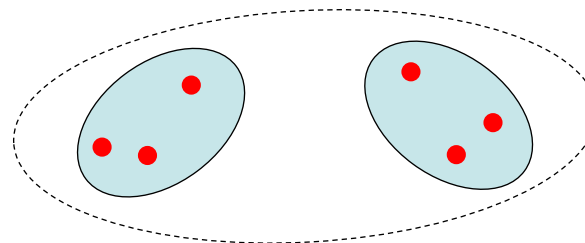
$$d(C_1, C_2) = \frac{1}{|C_1||C_2|} \sum_{\mathbf{x}_1 \in C_1} \sum_{\mathbf{x}_2 \in C_2} d(\mathbf{x}_1, \mathbf{x}_2)$$



- ウォード法（最小分散法）

- 結合後の分散と、結合前のクラスターそれぞれの分散の和との差
- 判別規準と同じ

$$d(C_1, C_2) = \text{var}(C_1 \cup C_2) - (\text{var}(C_1) + \text{var}(C_2))$$

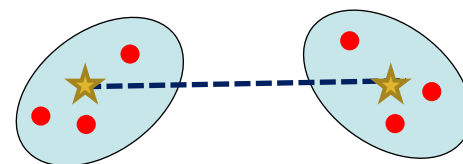


○外れ値に強く、実用性が高い

クラスター間の距離の定義 (3)

- 重心法
 - 二つのクラスターの重心間の距離

$$d(C_1, C_2) = d\left(\frac{1}{|C_1|} \sum_{\mathbf{x}_1 \in C_1} \mathbf{x}_1, \frac{1}{|C_2|} \sum_{\mathbf{x}_2 \in C_2} \mathbf{x}_2\right)$$



- 重み付重心法 (メディアン法)
 - 各クラスターの個体数の違いを考慮

(今はあまり使われない)

○ 計算コストが小さい

× クラスターの統合過程で、前段階よりも小さい距離となる“距離の逆転”が起こる場合がある

分析例

- まず距離行列を作る

```
> data1<-read.csv('seiseki.csv',header=F,row.names=1)
> data1.l2=dist(data1) #ユークリッド距離
> data1.l2
```

	Tanaka	Sato	Suzuki	Honda	Ueno	Yoshii
Sato	68.65858					
Suzuki	33.77869	81.11104				
Honda	60.13319	64.14047	52.67827			
Ueno	28.47806	60.75360	21.30728	47.10626		
Yoshii	63.37192	12.40967	75.66373	54.31390	56.38262	
Saito	67.88225	38.10512	87.53856	91.53142	67.72739	45.58509

```
> data1.l1=dist(data1,method="manhattan") #L1距離
> data1.l1
```

	Tanaka	Sato	Suzuki	Honda	Ueno	Yoshii
Sato	144					
Suzuki	47	173				
Honda	102	132	95			
Ueno	57	131	42	83		
Yoshii	136	22	165	110	123	
Saito	118	68	165	200	127	90

樹形図（デンドログラム）の表示

- 関数 hclust

```
> (data1.hc<-hclust(data1.l2))
```

Call:

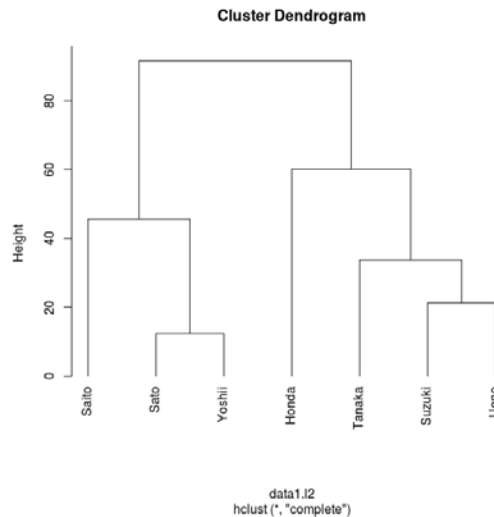
```
hclust(d = data1.l2)
```

Cluster method : complete

Distance : euclidean

Number of objects: 7

```
> plot(data1.hc,hang=-1)
```



```
> (data1.hc<-hclust(data1.l2,method="centroid"))
```

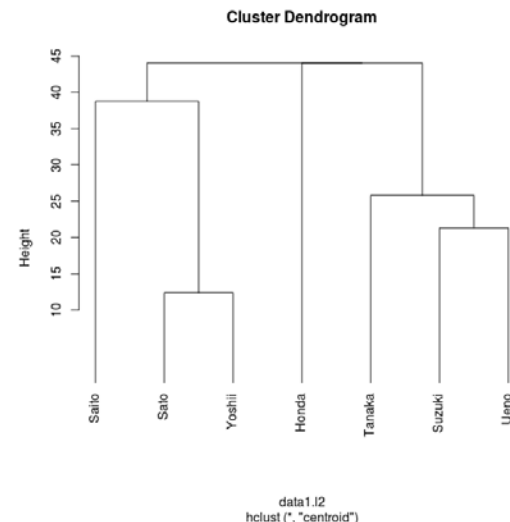
Call:

```
hclust(d = data1.l1, method = "ward")
```

Cluster method : ward

Distance : manhattan

Number of objects: 7



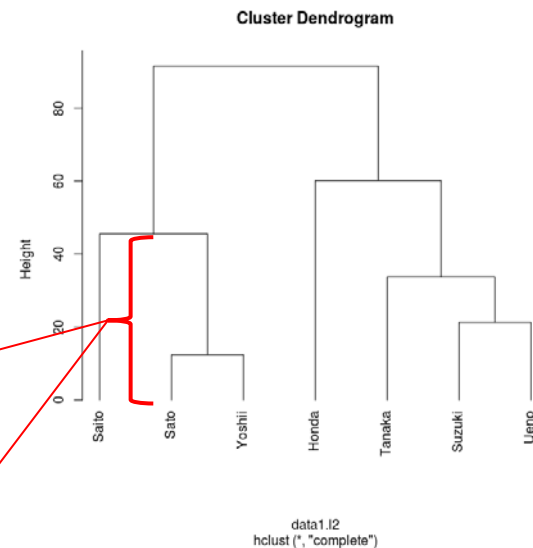
コーフェン行列

- 結合するタイミングでのクラス間距離を所属する各サンプルの距離にとった行列
- デンドログラムはコーフェン行列に基づいて作成される
 - 高さがコーフェン行列の各要素に対応

```
> data1.hc<-hclust(data1.l2)
```

```
> cophenetic(data1.hc)
```

	Tanaka	Sato	Suzuki	Honda	Ueno	Yoshii
Sato	91.53142					
Suzuki	33.77869	91.53142				
Honda	60.13319	91.53142	60.13319			
Ueno	33.77869	91.53142	21.30728	60.13319		
Yoshii	91.53142	12.40967	91.53142	91.53142	91.53142	
Saito	91.53142	45.58509	91.53142	91.53142	91.53142	45.58509



結果の妥当性

- コーフェン行列が元の距離行列と近いほどよとする
 - 相関係数（コーフェン相関係数）
> cor(data1.l2,cophenetic(data1.hc))
[1] 0.8944869
- あくまで一つの指標
 - 分析結果が妥当であることを保証するものではない

非階層的クラスタリング

- データの分割の良さを表すある評価関数によって、最適解を探索するアプローチ
- クラスタの数はあらかじめ与える手法が多い
 - 結果を解釈しながら、適当な数を探索
(またはAIC等を用いる)
- k-means
- Fuzzy c-means
- 混合分布モデル
- スペクトラルクラスタリング
- トピックモデル (pLSI, LDA)
- NMF (Non-negative Matrix Factorization)
- 自己組織化マップ
- Mean-shift クラスタリング
- ...

k-means法

- 各クラスタに所属するサンプルの、クラスタ中心点までの距離の和が最小となるように配置

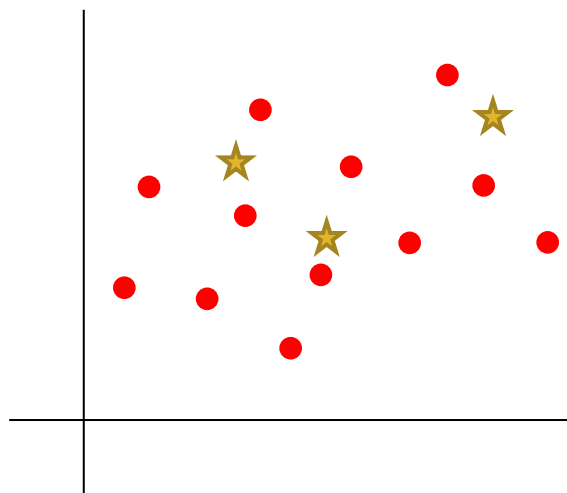
$$Err(\{C_k\}_{k=1}^K) = \sum_{k=1}^K \sum_{\mathbf{x}_i \in C_k} d(\bar{\mathbf{x}}_k, \mathbf{x}_i)$$

- クラスタ数 K は与える
- 通常、距離 d はユークリッド距離の二乗
- 実用上、非常によく用いられる手法
- c-means と呼ばれる

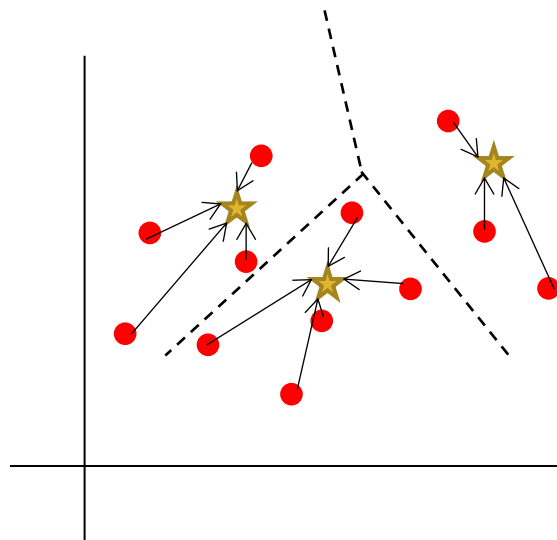
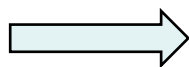
k-means法のアルゴリズム

- 1. 全データ $\{\mathbf{x}_i\}_{i=1}^N$ から K 個のデータをランダムに選び、クラスタ中心点 $\{v_j\}_{j=1}^K$ の初期値とする。
- 2. 各データ \mathbf{x}_i と各クラスタの中心点 v_j との距離を求め、最も近い中心のクラスタに \mathbf{x}_i を割り当てる。
- 3. 割り振ったデータをもとに各クラスタの中心 v_j を再計算する。
計算は通常割り当てられたデータの各要素の算術平均が使用される。
- 4. 2～3を繰り返し、全ての \mathbf{x}_i のクラスタの割り当てが変化しなくなるか、目的関数の変化量が事前に設定した一定の閾値を下回った場合に、収束したと判断して処理を終了する。

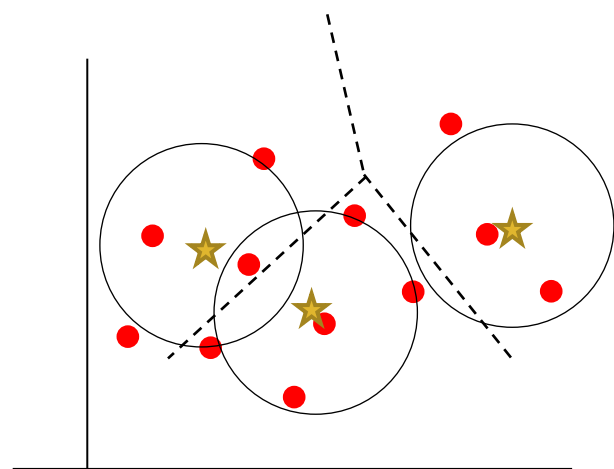
k-means法のアルゴリズム



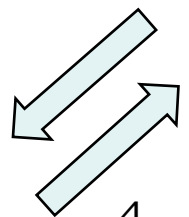
1. ランダムにクラスタ中心を選択



2. 各データを中心が最も近いクラスタへ割り当てる



3. 各クラスタの平均点を計算し、次の中心ベクトルとする



4. 収束するまで繰り返し

コード

```
def kMeans(dataSet, k, distMeas=distEclud, createCent=randCent):
    m = shape(dataSet)[0]
    clusterAssment = mat(zeros((m,2)))#create mat to assign data points
                                     #to a centroid, also holds SE of each point
    centroids = createCent(dataSet, k) #ランダムに初期化
    clusterChanged = True
    while clusterChanged:
        clusterChanged = False
        for i in range(m):#for each data point assign it to the closest centroid
            minDist = inf; minIndex = -1
            for j in range(k):
                distJI = distMeas(centroids[j,:],dataSet[i,:])
                if distJI < minDist:
                    minDist = distJI; minIndex = j
            if clusterAssment[i,0] != minIndex: clusterChanged = True
            clusterAssment[i,:] = minIndex,minDist**2
    print centroids
    for cent in range(k):#recalculate centroids
        ptsInClust = dataSet[nonzero(clusterAssment[:,0].A==cent)[0]]#get all the point in this cluster
        centroids[cent,:] = mean(ptsInClust, axis=0) #assign centroid to mean
    return centroids, clusterAssment
```

実行してみる

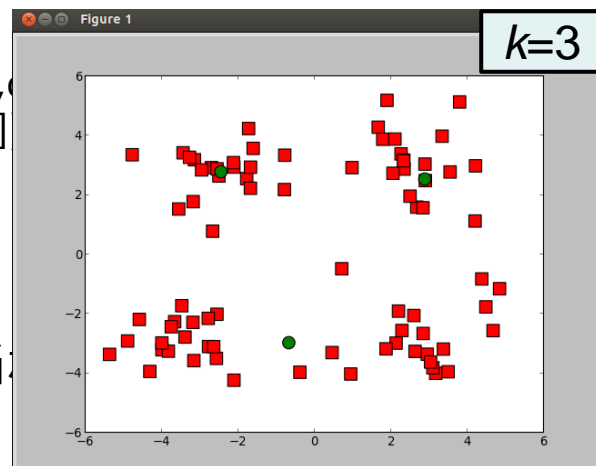
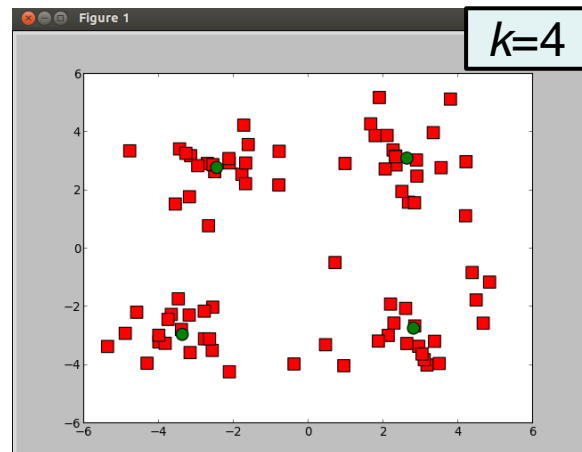
(test.py)

```
import numpy as np
import matplotlib.pyplot as plt
import kMeans
```

```
dataMat=np.mat(kMeans.loadDataSet('testSet.txt'))
dataMat.shape
myCentroids, clustAssign = kMeans.kMeans(dataMat,4) #k=4
```

```
fig = plt.figure()
ax = fig.add_subplot(111)
ax.scatter(np.array(dataMat[:,0]),np.array(dataMat[:,1]),s=150,c='r')
ax.scatter(np.array(myCentroids[:,0]),np.array(myCentroids[:,1]),s=150,c='g')
plt.show()
```

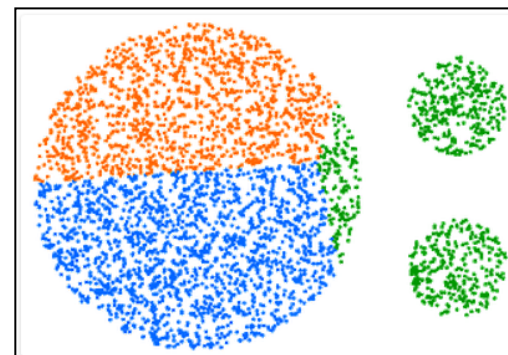
- この例だと 3 ~ 4 回のループで収束
- クラスタ数 k を変えて何回か試してみる。いつも妥当



k-means法の注意点

- k-meansの解は**局所最適解**しか保証されない
 - 初期値に強く依存
 - 通常、何度かk-meansを試行（毎回ランダムにk個の点を選びなおし初期値とする）し、目的関数が最小となった場合を採用する
- 分散の等しい超球状のクラスタを前提とする
 - 各クラスタに属するデータ数はおおむね等しいことを仮定
 - データとクラスタ数によっては必ずしも適切な結果にならない！

目的関数



k-means の改良手法（主に初期値依存性に対処）

- k-means++ [Arthur & Vassilvitskii, 2007]
 - 初期値の取り方を工夫した改良手法
 - なるべく離れるように最初のクラスタ中心を配置
- Bisecting k-means
 - データ全体を内包する一つの大きなクラスタからスタートし、階層的にk-means ($k=2$)を行い各クラスタを二分割していく方法
 - オリジナルのk-meansよりもよい解が得られる場合がある
 - 計算コストが小さい
 - 階層構造が得られる
- 配布プログラムのbiKmeansを参照

おまけ：k-meansとPCA

- K-means Clustering via Principal Component Analysis [Ding & He, ICML2004]
- $k=2$ の場合、k-meansの結果はPCAの最大固有値の軸（第一主成分）の正負と一致

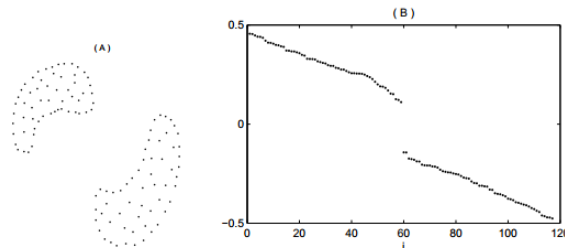


Figure 1. (A) Two clusters in 2D space. (B) Principal component $v_1(i)$, showing the value of each element i .

- $k>2$ の場合、k-meansで得られる k 個のクラスタ中心ベクトルが張る空間が、PCAの上位 $(k-1)$ 個の固有ベクトルが張る空間と一致
 - PCAで $k-1$ 次元へ圧縮してからk-meansをかけてもそれほど変わらない
- PCAは潜在的にクラスタリングをしていると解釈できる
 - より一般には、スペクトラルクラスタリングと呼ばれる（カーネルPCAの特殊な場合）

Fuzzy c-means法

- k-means (=c-means) の改良版
- 各事例 \mathbf{x}_i を複数のクラスタへ重みをつけて割り当てる

$$Err(\{u_{ki}\}, \{\boldsymbol{\mu}_k\}_{k=1}^C) = \sum_{k=1}^C \sum_{i=1}^N (u_{ki})^m d(\boldsymbol{\mu}_k, \mathbf{x}_i)$$

$$u_{ki} \in [0,1] \quad \sum_{k=1}^C u_{ki} = 1 \quad (i = 1, \dots, N) \quad \text{メンバシップ係数}$$

m : 割り当てのファジーさを決めるパラメータ
(C と共に与える)

- Fuzzy : 曖昧な～

Fuzzy c-means法のアルゴリズム

- 1. 全データのメンバシップ $\{u_{ki}\}$ をランダムに初期化
- 2. 現在の $\{u_{ki}\}$ を用いて、各クラスタの中心点 μ_k を計算

$$\mu_k = \frac{\sum_{i=1}^N (u_{ki})^m \mathbf{x}_i}{\sum_{i=1}^N (u_{ki})^m}$$

- 3. $\{\mu_k\}$ を用いて、データ \mathbf{x}_i のクラスタ k への割り当てを更新

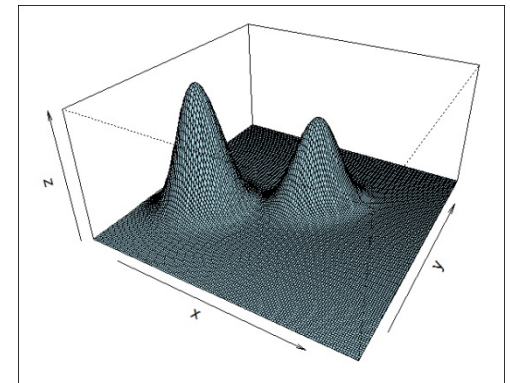
$$u_{ki} = \left[\sum_{j=1}^c \left(\frac{d(\mathbf{x}_i, \mu_k)}{d(\mathbf{x}_i, \mu_j)} \right)^{\frac{1}{m-1}} \right]^{-1}$$

- 4. 2～3を繰り返し、パラメータの変化が十分小さくなれば処理を終了。

混合分布モデルを用いたクラスタリング

- データの背後に存在する確率モデルを推定し、クラスタリングへ利用する
- 混合正規分布 (Gaussian Mixture Model, GMM)
 - 複数の正規分布を足し合わせた確率モデル

$$p(\mathbf{x}; \Theta) = \sum_{k=1}^K \alpha_k N(\mathbf{x}; \boldsymbol{\mu}_k, \Sigma_k) \quad \alpha_k \in [0,1] \quad \sum_{k=1}^K \alpha_k = 1$$
$$N(\mathbf{x}; \boldsymbol{\mu}_k, \Sigma_k) = \frac{1}{(2\pi)^{d/2} |\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k)\right)$$



- クラスタリングのための手法ではないが、実用上便利なので道具としてしばしばもちいられる。
- 混合する正規分布の一つ一つをクラスタと解釈
- k-meansはこれの特殊な場合

混合正規分布の最尤推定

- 対数尤度
 - 最大化したいが、解析的には解けない

$$\log L(\Theta) = \sum_{i=1}^N \log \left\{ \sum_{k=1}^K \alpha_k N(\mathbf{x}_i; \boldsymbol{\mu}_k, \Sigma_k) \right\}$$

- EMアルゴリズムという枠組で最適化が可能
 - 観測不可能な潜在変数に確率モデルが依存する場合の最尤推定法
 - Eステップ(expectation)、Mステップ(maximization)を交互に繰り返す
 - Eステップ：現在推定されている潜在変数の分布に基づき、モデル尤度の期待値を計算する
 - Mステップ：Eステップで求めた期待値を最大化するようにパラメータ（潜在変数含む）を求める
- 対数尤度が単調増加することが証明されている

混合正規分布の場合

- 潜在変数=各データがどの正規分布に属しているか

- クラスタリングの文脈でまさに求めたいもの
- \mathbf{x}_i が j 番目の分布に属する確率を $q(i, j)$ と表記する

- 1. $\{\alpha_j^0, \boldsymbol{\mu}_j^0, \Sigma_j^0\}_{j=1}^K$ をランダムに初期化 ($t=0$)

- 2. Eステップ: 現在のパラメータを固定し、潜在変数の分布を更新

$$\hat{q}(i, j)^{t+1} = \frac{\alpha_j^t N(\mathbf{x}_i; \boldsymbol{\mu}_j^t, \Sigma_j^t)}{\sum_{k=1}^K \alpha_k^t N(\mathbf{x}_i; \boldsymbol{\mu}_k^t, \Sigma_k^t)} \quad (\text{尤度の期待値が陰に求まる})$$

事後確率の比

- 3. Mステップ: 現在の潜在変数分布に基づき、尤度を最大化

$$\alpha_j^{t+1} = \frac{\sum_{i=1}^N \hat{q}^{t+1}(i, j)}{\sum_{i=1}^N \sum_{k=1}^K \hat{q}^{t+1}(i, k)} \quad \boldsymbol{\mu}_j^{t+1} = \frac{\sum_{i=1}^N \hat{q}^{t+1}(i, j) \mathbf{x}_i}{\sum_{i=1}^N \hat{q}^{t+1}(i, j)} \quad \Sigma_j^{t+1} = \frac{\sum_{i=1}^N \hat{q}^{t+1}(i, j) (\mathbf{x}_i - \boldsymbol{\mu}_j^{t+1})(\mathbf{x}_i - \boldsymbol{\mu}_j^{t+1})^T}{\sum_{i=1}^N \hat{q}^{t+1}(i, j)}$$

j 番目の分布に属する
サンプル数の比

各サンプルの j 番目の分布への寄与で重みづけた
平均と分散

- 4. 収束するまで 2、3 を繰り返し

まとめ

- クラスタリング
 - 与えられたデータ集合をいくつかのまとまりに分割
 - 外的分離と内的結合
- 階層的クラスタリング（凝集型）
 - データ一つ一つがクラスタの状態から始めて、逐次的にクラスタを併合
 - 群平均法かワード法
- 非階層的クラスタリング
 - クラスタの良さを定義する目的関数を最適にする分割を探索する
 - ハード（排他的なクラスタ）： k-means法
 - ソフト（複数クラスタへ重み付け）：混合正規分布など
- 注意すべきポイント
 - 適切な距離を使う
 - 得られたクラスタは絶対的・客観的なものではない
 - 自分で結果をみて解釈を与えることが重要