クラスタ分析

非階層的方法と分析の評価

村田 昇

講義概要

- 第 1 回 : クラスタ分析の考え方と階層的方法
- **第 2 回 : 非階層的方法と分析の評価**

クラスタ分析の復習

クラスタ分析

- クラスタ分析 (**cluster analysis**) の目的
  個体の間に隠れている**集まり=クラスタ**を個体間の “距離” にもとづいて発見する方法
- 個体間の類似度・距離 (非類似度) を定義
  - 同じクラスタに属する個体どうしは似通った性質
  - 異なるクラスタに属する個体どうしは異なる性質
- さらなるデータ解析やデータの可視化に利用
- 教師なし学習の代表的な手法の一つ

クラスタ分析の考え方

- 階層的方法
  - データ点およびクラスタの間に **距離** を定義
  - 距離に基づいてグループ化
    - 近いものから順にクラスタを **凝集**
    - 近いものが同じクラスタに残るように **分割**
- 非階層的方法
  - クラスタの数を事前に指定
  - クラスタの **集まりの良さ** を評価する損失関数を定義
  - 損失関数を最小化するようにクラスタを形成

階層的クラスタリング

- 凝集的手続き
  1. データ・クラスタ間の距離を定義
     - データ点間の距離
     - クラスタ間の距離
  2. データ点およびクラスタ間の距離を計算
  3. 最も近い 2 つを統合し新たなクラスタを形成
  4. クラスタ数が 1 つになるまで 2-3 の手続きを繰り返す

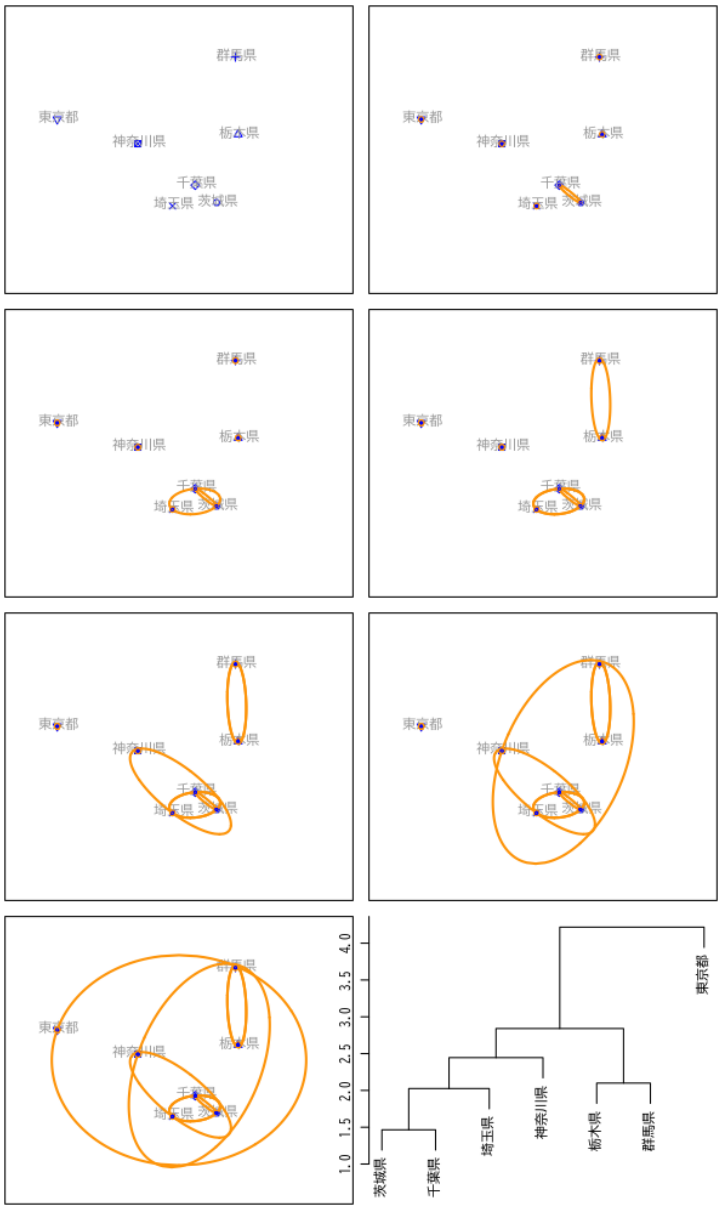
群馬県
東京都
神奈川県
栃木県
千葉県
埼玉県
茨城県
4.0
3.5
3.0
2.5
2.0
1.5
1.0
東京都
神奈川県
栃木県
群馬県
埼玉県
千葉県
茨城県

図 1: 凝集的手続きの例

非階層的方法

非階層的方法の手続き

- 対象の変数 : $\boldsymbol{X} = (X_1, X_2, \ldots, X_d)^\mathsf{T}$ (d 次元)
- 観測データ : n 個の個体の組

$$\{\boldsymbol{x}_i\}_{i=1}^n = \{(x_{i1}, x_{i2}, \ldots, x_{id})^\mathsf{T}\}_{i=1}^n$$

- 個体とクラスタの対応 C を推定

$$C(i) = (\text{個体 } i \text{ が属するクラスタ番号})$$

  - 対応 C の **全体の良さ** を評価する損失関数を設定
  - 観測データ $\{\boldsymbol{x}_i\}_{i=1}^n$ に最適な対応 $\{C(i)\}_{i=1}^n$ を決定

k-平均法の損失関数

- クラスタの個数 k を指定
- 2 つの個体 i, i' の **近さ=損失** を距離の二乗で評価

$$\|\boldsymbol{x}_i - \boldsymbol{x}_{i'}\|^2 = \sum_{j=1}^d (x_{ij} - x_{i'j})^2$$

- 損失関数 $W(C)$: クラスタ内の平均の近さを評価

$$W(C) = \sum_{l=1}^k \frac{1}{n_l} \sum_{i:C(i)=l} \sum_{i':C(i')=l} \|\boldsymbol{x}_i - \boldsymbol{x}_{i'}\|^2$$

k-平均法の性質

- クラスタ l に属する個体の平均

$$\bar{\boldsymbol{x}}_l = \frac{1}{n_l} \sum_{i:C(i)=l} \boldsymbol{x}_i, \quad (n_l \text{ はクラスタ } l \text{ に属する個体数})$$

- 損失関数 $W(C)$ の等価な表現

$$W(C) = 2 \sum_{l=1}^k \sum_{i:C(i)=l} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_l\|^2$$

- 最適な対応 C : クラスタ内変動の総和が最小

近似的な最適化

クラスタ対応の最適化

- 最適化 : 損失関数 $W(C)$ を最小とする C を決定
- 貪欲な C の探索
  - 原理的には全ての値を計算すればよい
  - 可能な C の数 : k^n 通り (有限個のパターン)
  - サンプル数 n が小さくない限り実時間での実行は不可能
- 近似的な C の探索
  - いくつかのアルゴリズムが提案されている
  - 基本的な考え方 : **Lloyd-Forgy のアルゴリズム**

    標本平均と変動の平方和の性質を利用

$$\bar{\boldsymbol{x}}_l = \arg\min_{\boldsymbol{\mu}} \sum_{i:C(i)=l} \|\boldsymbol{x}_i - \boldsymbol{\mu}\|^2 \quad \text{(クラスタ } l \text{ の標本平均)}$$

Lloyd-Forgy のアルゴリズム

1. クラスタ中心の初期値 $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k$ を与える
2. 各データの所属クラスタ番号 $C(i)$ を求める

$$C(i) = \arg\min_l \|\boldsymbol{x}_i - \boldsymbol{\mu}_l\|$$

3. 各クラスタ中心 $\boldsymbol{\mu}_l$ $(l = 1, 2, \ldots, k)$ を更新する

$$\boldsymbol{\mu}_l = \frac{1}{n_l} \sum_{i:C(i)=l} \boldsymbol{x}_i, \quad n_l = |\{\boldsymbol{x}_i | C(i) = l\}|$$

4. 中心が変化しなくなるまで 2,3 を繰り返す

アルゴリズムの性質

- 結果は **確率的**
  - 初期値 $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k$ に依存
  - アルゴリズムの成否は確率的なため，最適解が得られない場合もある
- 一般には複数の初期値をランダムに試して損失を最小とする解を採用する
- 平均の代わりにメドイド (medoid; 中心にある観測値) を用いる方法もある

$$\boldsymbol{x}_l^{\text{medoid}} = \arg\min_{\boldsymbol{x}_i} \sum_{i':C(i')=l} \|\boldsymbol{x}_i - \boldsymbol{x}_{i'}\|^2$$

事例

- 都道府県別好きなおむすびの具 (一部) での例

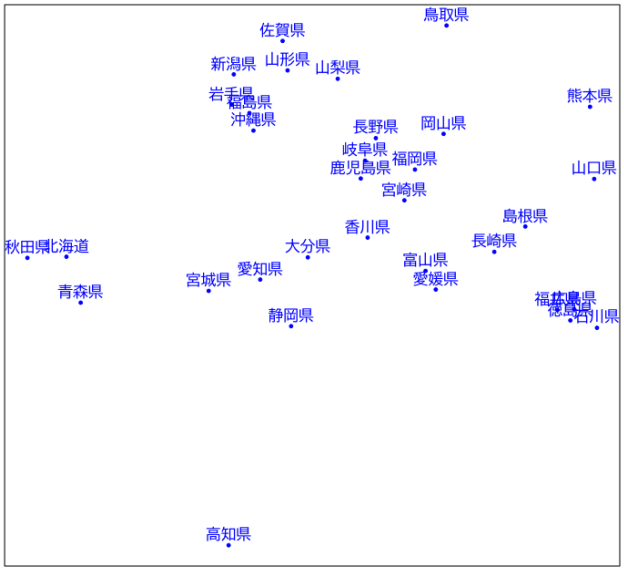

図 2: 非階層的クラスタリング

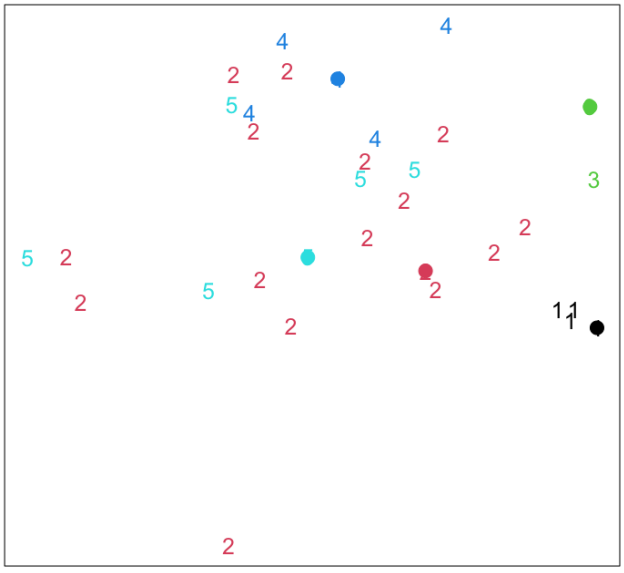

図 3: Lloyd-Forgy のアルゴリズム (その 1)

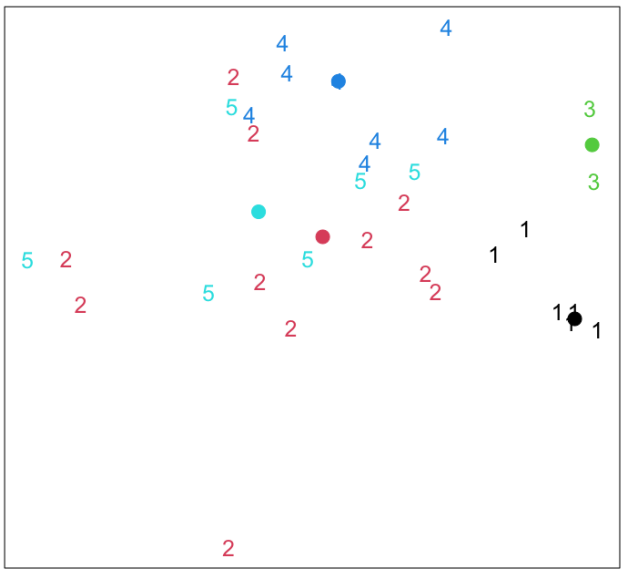

図 4: Lloyd-Forgy のアルゴリズム (その 2)

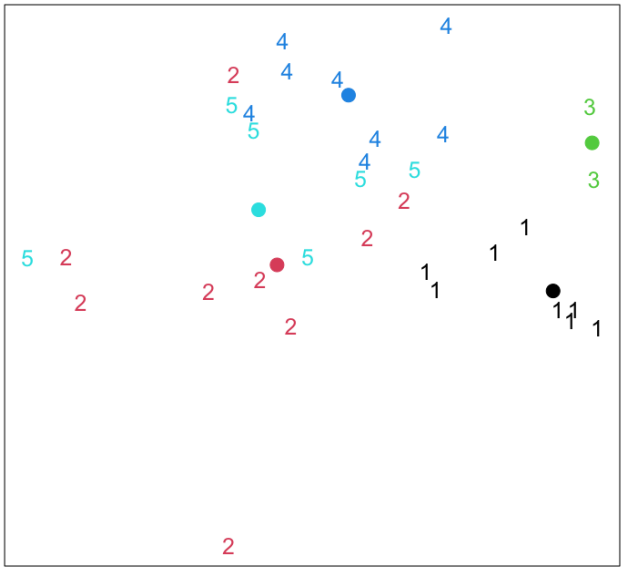

図 5: Lloyd-Forgy のアルゴリズム (その 3)

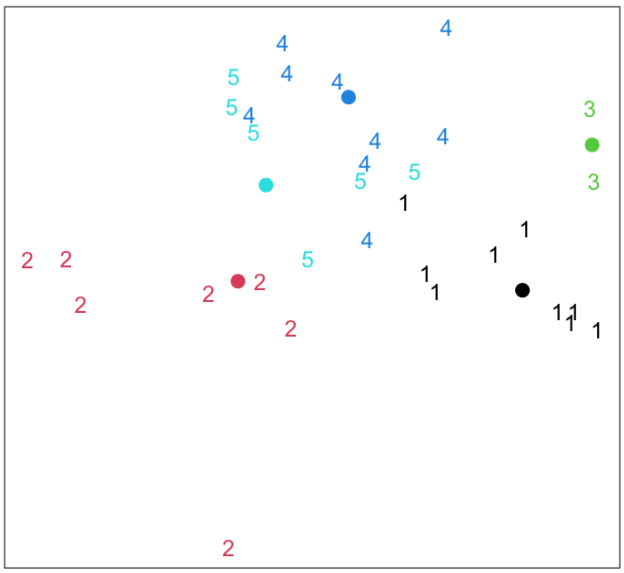

図 6: Lloyd-Forgy のアルゴリズム (その 4)

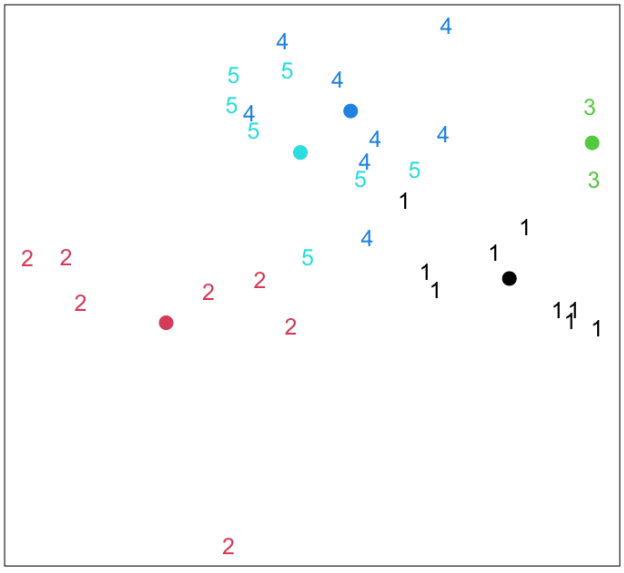

図 7: Lloyd-Forgy のアルゴリズム (その 5)

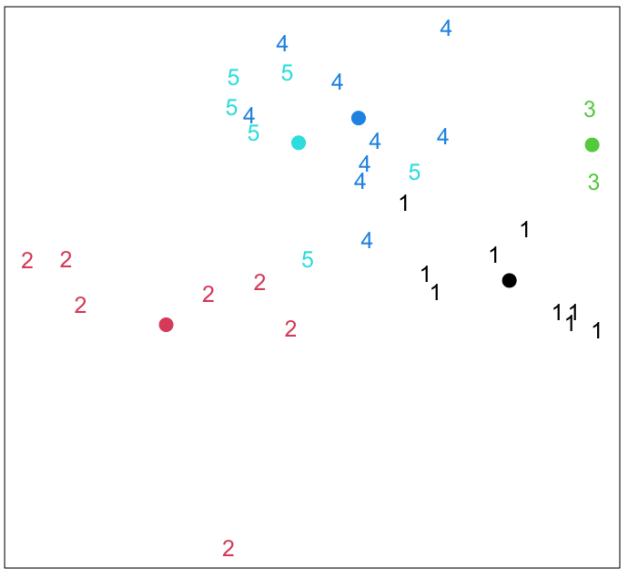

図 8: Lloyd-Forgy のアルゴリズム (その 6)

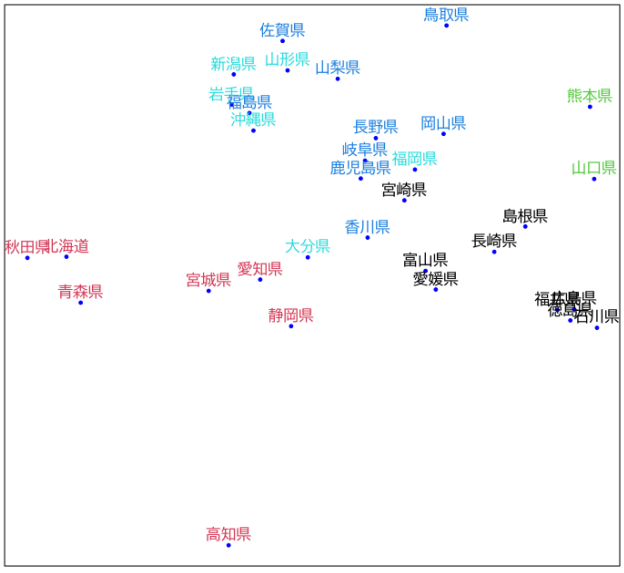

図 9: クラスタリングの結果

実習

R：k-平均法

- 関数 kmeans()

```
kmeans(x, centers, iter.max = 10, nstart = 1,
       algorithm = c("Hartigan-Wong", "Lloyd", "Forgy",
                     "MacQueen"), trace = FALSE)
#' x: データフレーム
#' centers: クラスタ数
#' iter.max: 最大繰り返し数
#' nstart: 初期値の候補数
#' algorithm: 最適化法の指定．既定値は 'Hartigan-Wong'
```

  - 結果は変数のスケールにも依存
    - 例えば測定値の単位により異なる
    - 必要ならば主成分分析の場合と同様にデータの標準化を行う

R：2 次元でのクラスタ表示

- 関数 ggfortify::autoplot() (ggplot 系)

```
autoplot(object, data = NULL, colour = "cluster", ...)
#' object: stats::kmeans(), cluster::pam() などの返値
#' data: クラスタリングに用いたデータ (kmeans の場合に必要)
#' 詳細は '?ggfortify::autoplot.kmeans()' を参照
```

- 関数 cluster::clusplot() を利用することもできる

練習問題

- データセット japan_social.csv を用いて以下を確認しなさい

```
js_df <- read_csv("data/japan_social.csv") |>
  column_to_rownames(var = "Pref") |> # 'Pref' を行名に変換
  select(-Area) # 地方名は除く
```

  - 関数 kmeans() を用いて各変数平均 0，分散 1 に標準化 (関数 scale() を利用) したデータを 7 クラスタに分割しなさい
  - 各クラスタ内の県名を表示しなさい
  - 2 次元散布図に各クラスタを表示しなさい
- データセット omusubi.csv でも確認しなさい

R：k-メドイド法

- 関数 cluster::pam()

```
pam(x, k, diss = inherits(x, "dist"),
    metric = c("euclidean", "manhattan"),
    medoids = if(is.numeric(nstart)) "random",
    nstart = if(variant == "faster") 1 else NA,
    stand = FALSE, cluster.only = FALSE,
    do.swap = TRUE,
    keep.diss = !diss && !cluster.only && n < 100,
    keep.data = !diss && !cluster.only,
    variant = c("original", "o_1", "o_2", "f_3", "f_4", "f_5", "faster"),
    pamonce = FALSE, trace.lev = 0)
#' x: データフレーム，または距離行列
#' k: クラスタの数
```

```
#' metric: 距離の指定 (x がデータフレームの場合)
#' stand: 標準化 (平均 0, 絶対偏差 1)
```

練習問題

- データセット `japan_social.csv` を用いて以下を確認しなさい
  - 関数 `pam()` を用いて各変数平均 0, 絶対偏差 1 に標準化したデータを 7 クラスタに分割しなさい
  - 各クラスタ内の県名を表示しなさい
  - 2 次元散布図に各クラスタを表示しなさい
- データセット `omusubi.csv` でも確認しなさい

クラスタ構造の評価

階層的方法の評価

- 評価の対象
  - データ $\boldsymbol{x}_i$ と最初に統合されたクラスタ C の距離

$$d_i = D(\boldsymbol{x}_i, C)$$

  - 最後に統合された 2 つのクラスタ C', C'' の距離

$$D = D(C', C'')$$

- **凝集係数** (agglomerative coefficient)

$$AC = \frac{1}{n} \sum_{i=1}^{n} \left(1 - \frac{d_i}{D}\right)$$

凝集係数の性質

- 定義より

$$0 \leq AC \leq 1$$

  - 1 に近いほどクラスタ構造が明瞭
- banner plot: 各 $(1 - d_i/D)$ を並べた棒グラフ
- banner plot の面積比として視覚化

非階層的方法の評価

- 評価の対象
  - $\boldsymbol{x}_i$ を含むクラスタ C^1 と $\boldsymbol{x}_i$ の距離

$$d_i^1 = D(\boldsymbol{x}_i, C^1 \setminus \boldsymbol{x}_i)$$

  - 一番近いクラスタ C^2 と $\boldsymbol{x}_i$ の距離

$$d_i^2 = D(\boldsymbol{x}_i, C^2)$$

- **シルエット係数** (silhouette coefficient)

$$S_i = \frac{d_i^2 - d_i^1}{\max(d_i^1, d_i^2)}$$

シルエット係数の性質

- 定義より

$$-1 \leq S_i \leq 1$$

  - 1 に近いほど適切なクラスタリング
- 全体の良さを評価するには S_i の平均を用いる
- 距離の計算を適切に行えば階層的方法でも利用可

実習

R：凝集係数

- 関数 `cluster::agnes()` の返値の情報

```
object[["ac"]] # 凝集係数の取得 (object$ac でも良い)
object[["height"]] # デンドログラムの高さ (結合時のクラスタ距離)
object[["order.lab"]] # デンドログラムの並び順のラベル
#' object: cluster::agnes() の返値
```

  - これらを利用して banner plot を描くことができる
  - 関数 `summary(object)` はこれらの情報をまとめて表示する
- 視覚化のための関数 (graphics 系)

```
#' cluster::plot.agnes() 系統樹および凝集係数の表示
plot(x, ask = FALSE, which.plots = NULL, main = NULL,
     sub = paste("Agglomerative Coefficient = ",round(x$ac, digits = 2)),
     adj = 0, nmax.lab = 35, max.strlen = 5, xax.pretty = TRUE, ...)
#' x: cluster::agnes() の返値
#' which.plots: 1 (banner plot), 2 (dendrogram)
#' banner plot の表示には以下の cluster::bannerplot() が呼出される
bannerplot(x, w = rev(x$height), fromLeft = TRUE,
           main=NULL, sub=NULL, xlab = "Height",  adj = 0,
           col = c(2, 0), border = 0, axes = TRUE, frame.plot = axes,
           rev.xax = !fromLeft, xax.pretty = TRUE,
           labels = NULL, nmax.lab = 35, max.strlen = 5,
           yax.do = axes && length(x$order) <= nmax.lab,
           yaxRight = fromLeft, y.mar = 2.4 + max.strlen/2.5, ...)
```

練習問題

- データセット japan_social.csv を用いて以下を検討しなさい
  - 関数 agnes() を用いて階層的クラスタリングを行いなさい
    - 標準化: 行う
    - データ距離: ユークリッド距離，およびマンハッタン距離
    - クラスタ距離: 群平均法
  - 凝集係数を用いて 2 つのデータ距離の評価を行いなさい
  - 凝集係数が低いいくつかのデータを削除して評価しなさい

R：シルエット係数

- 関数 cluster::pam() の返値の情報

```
object[["silinfo"]] # シルエット係数に関する様々な情報
object[["silinfo"]][["widths"]] # 各データのシルエット係数
object[["silinfo"]][["clus.avg.widths"]] # 各クラスタのシルエット係数の平均
object[["silinfo"]][["avg.width"]] # シルエット係数の平均
#' object: cluster::agnes() の返値
```

  - 関数 summary(object) はこれらの情報をまとめて表示する

- 補助的な関数

```
#' シルエット係数の取得
silhouette(x, ...)
#' x: cluster::pam() などの返値
silhouette(x, dist, dmatrix, ...)
#' x: stats::cutree() などの返値
#' dist/dmatrix: 距離行列または解離度を表す行列など
```

- 視覚化のための関数 (ggplot 系)

```
#' ggfortify::autoplot.silhouette() シルエット係数の表示
autoplot(object,
  colour = "red", linetype = "dashed", size = 0.5, bar.width = 1, ...)
#' object: cluster::silhouette() の返値
#' colour/linetype/size: reference line の設定
```

- 視覚化のための関数 (graphics 系)

```
#' cluster::plot.partition() 2次元クラスタおよびシルエット係数の表示
plot(x, ask = FALSE, which.plots = NULL,
     nmax.lab = 40, max.strlen = 5, data = x$data, dist = NULL,
     stand = FALSE, lines = 2,
     shade = FALSE, color = FALSE, labels = 0, plotchar = TRUE,
     span = TRUE, xlim = NULL, ylim = NULL, main = NULL, ...)
#' x: cluster::pam() などの返値
#' which.plots: 1 (cluster plot), 2 (silhouette plot)
#' silhouette plot の表示には以下の cluster::plot.silhouette() が呼出される
plot(x, nmax.lab = 40, max.strlen = 5,
     main = NULL, sub = NULL, xlab = expression("Silhouette width "* s[i]),
     col = "gray",  do.col.sort = length(col) > 1, border = 0,
     cex.names = par("cex.axis"), do.n.k = TRUE, do.clus.stat = TRUE, ...)
#' x: cluster::silhouette() の返値
```

練習問題

- データセット `omusubi.csv` を用いて以下を検討しなさい
  - Hellinger 距離を用いて距離行列を計算しなさい

    $\boldsymbol{p}, \boldsymbol{q}$ を確率ベクトルとして定義される確率分布の距離

    $$d_{\text{hel}}(\boldsymbol{p}, \boldsymbol{q}) = \frac{1}{\sqrt{2}} d_{\text{euc}}(\sqrt{\boldsymbol{p}}, \sqrt{\boldsymbol{q}})$$

  - クラスタ数 4-10 のシルエット係数を比較しなさい
  - 適当と思われるクラスタ数による分析を行いなさい
  - Euclid 距離を用いて同様な分析を行いなさい

次回の予定

- **第 1 回 : 時系列の基本モデル**
- 第 2 回 : モデルの推定と予測