RjpWiki アーカイブス

【多変量解析関数一覧 (06.06.25) 】

1　クラスタリングと樹状図

1.1　クラス ”hclust” のオブジェクトへの変換 as.hclust()

他のクラスタリング関数の出力オブジェクトを、クラス ”hclust” のオブジェクトに変換する。

```
as.hclust(x, ...)
```

1.2　樹状図のクラスタの周りに枠を描く rect.hclust()

樹状図のクラスタの周りに枠を描く。

```
rect.hclust(tree, k = NULL, which = NULL, x = NULL, h = NULL,
            border = 2, cluster = NULL)
```

1.3　樹状図を並べ換える reorder(x, wts, ...)

樹状図を並べ換える。

```
reorder(x, ...)
## クラス "dendrogram" に対するメソッド
reorder(x, wts, ...)
```

1.4　階層型クラスタリングのための共表型距離 cophenetic()

階層型クラスタリングのための共表型距離を計算する関数である。(生物学用語で ”phenetics” とは「表現学、表型分類」と訳される。)

```
cophenetic(x)
```

1.5　クラスタリング木の分割 cutree()

hclust() の出力であるような、木を、必要なグループ数、もしくは分割の高さを与えて、幾つかのグループに分割する。

```
cutree(tree, k = NULL, h = NULL)
```

1.6 樹状図に対するメソッド

hclust() の出力であるような木を、必要なグループ数、もしくは分割の高さを与えて、幾つかのグループに分割する。

```
as.dendrogram(object, ...)
plot(x, type = c("rectangle", "triangle"),
     center = FALSE, edge.root = !is.null(attr(x,"edgetext")),
     nodePar = NULL, edgePar = list(), xlab = "", ylab = "",
     horiz = FALSE, frame.plot = FALSE, ...)
cut(x, h, ...)
print(x, digits, ...)
rev(x)
str(object, max.level = 0, digits.d = 3,
    give.attr = FALSE, wid = getOption("width"),
    nest.lev = 0, indent.str = "", ...)
```

1.7 階層的クラスタリング hclust(), plclust()

非類似性のセットに基づく階層的クラスタ解析と、それを解析するメソッド。

```
hclust(d, method = "complete", members=NULL)
  クラス 'hclust' に対するプロットメソッド
plot(x, labels = NULL, hang = 0.1,
     axes = TRUE, frame.plot = FALSE, ann = TRUE,
     main = "Cluster Dendrogram",
     sub = NULL, xlab = NULL, ylab = "Height", ...)
plclust(tree, hang = 0.1, unit = FALSE, level = FALSE, hmin = 0,
        square = TRUE, labels = NULL, plot. = TRUE,
        axes = TRUE, frame.plot = FALSE, ann = TRUE,
        main = "", sub = NULL, xlab = NULL, ylab = "Height")
```

1.8 ヒートマップ heatmap()

ヒートマップは疑似的なカラーイメージ (基本的に image(t(x))) で、樹状図が左と上の端に描かれる。典型的には、樹状図が作られた際の制約下で、行と列をある値のセット (行、または列平均) に従い、並べ換える。

```
heatmap(x, Rowv=NULL, Colv=if(symm)"Rowv" else NULL, distfun = dist,
        hclustfun = hclust, add.expr, symm = FALSE,
        revC = identical(Colv, "Rowv"),
        scale=c("row", "column", "none"), na.rm = TRUE,
        margins = c(5, 5), ColSideColors, RowSideColors,
        cexRow = 0.2 + 1/log10(nr), cexCol = 0.2 + 1/log10(nc),
        labRow = NULL, labCol = NULL, main = NULL, xlab = NULL,
        ylab = NULL, verbose = getOption("verbose"), ...)
```

1.9 樹状図のインタラクティブな操作 identify.hclust()

identify.hclust() は、第一マウスボタンが押された時の、グラフィックスポインターの位置を読み取る。それから、木をポインターの垂直位置で切断し、ポインターの水平位置を含むクラスタをハイライトする。オプションとして、この関数はクラスタに含まれるデータ点の添字に適用できる。(identify.hclust() は identify() 関数のクラス "hclust" に対するメソッドである。)

```
identify.hclust(x, FUN = NULL, N = 20, MAXCLUSTER = 20,
                DEV.FUN = NULL, ...)
```

1.10 k-mean 法によるクラスタリング kmeans()

データ行列に対し、k-mean 法によるクラスタリングを実行する。

```
kmeans(x, centers, iter.max = 10)
```

1.11 樹状図中の葉の順序を得る order.dendrogram()

樹状図の葉の順序 (添字) を返す。これは、任意の追加データの適当な成分を得ることに使える。葉の添字の順序が左から右へ与えられる。

```
order.dendrogram(x)
```

2 主成分分析

2.1 主成分分析 prcomp()

与えられたデータ行列に主成分分析を実行し、結果をクラス ”prcomp” のオブジェクトとして返す。

```
prcomp(x, retx = TRUE, center = TRUE, scale. = FALSE, tol = NULL)
```

2.2 主成分分析 princomp()

与えられた数値データ行列に対し主成分分析を行い、結果をクラス princomp のオブジェクトとして返す。

```
## クラス formula に対するメソッド
princomp(formula, data = NULL, subset, na.action, ...)
## 既定メソッド
princomp(x, cor = FALSE, scores = TRUE, covmat = NULL,
         subset = rep(TRUE, nrow(as.matrix(x))), ...)
```

2.3 主成分分析用のバイプロット biplot()

princomp もしくは prcomp() 関数の出力から、狭義のバイプロットグラフを描く。

```
biplot(x, choices = 1:2, scale = 1, pc.biplot = FALSE, ...)
```

2.4 主成分の分散のプロット screeplot()

関数は主成分分析の screeplot(”scree” とは ”小石” を意味する) を行う。

```
screeplot(x, npcs = min(10, length(x$sdev)),
          type = c("barplot", "lines"),
          main = deparse(substitute(x)), ...)
```

2.5 主成分分析結果の要約

クラス ”princomp” のオブジェクトの summary メソッドである。

```
## クラス princomp に対するメソッド
summary(object, loadings = FALSE, cutoff = 0.1, ...)
## クラス summary.princomp に対するメソッド
print(x, digits=3, loadings=x$print.loadings, cutoff=x$cutoff, ...)
```

3 因子分析

3.1 因子分析 factanal()

データ行列の共分散行列に対し、最尤法による因子分析を実行する。

```
factanal(x, factors, data = NULL, covmat = NULL, n.obs = NA,
         subset, na.action, start = NULL,
         scores = c("none", "regression", "Bartlett"),
         rotation = "varimax", control = NULL, ...)
```

3.2 因子分析用の回転 varimax(), promax()

因子分析用の回転メソッド。

```
varimax(x, normalize = TRUE, eps = 1e-5)
promax(x, m = 4)
```

3.3 因子分析負荷量の出力 loadings()

因子分析に於ける負荷量 (loading) を出力する。

```
loadings(x)
## クラス loadings に対する print メソッド
print(x, digits = 3, cutoff = 0.1, sort = FALSE, ...)
## クラス factanal に対する print メソッド
print(x, digits = 3, ...)
```

4 正準相関、canonical correlation

4.1 正準相関 cancor()

二つのデータ行列間の正準相関係数を計算する。

```
cancor(x, y, xcenter = TRUE, ycenter = TRUE)
```

5 多次元尺度法、MDS (multidimensional scaling)

5.1 古典的 (距離) 多次元尺度法 cmdscale()

データ行列の多次元尺度法用の関数である。principal coordinates analysis という名前で呼ばれることがある (Gower)。

```
cmdscale(d, k = 2, eig = FALSE, add = FALSE, x.ret = FALSE)
```

5.2 Kruskal の非計量的多次元尺度法 isoMDS

5.3 非計量的多次元尺度法 sammon

6 ユーティリティ関数

6.1 距離行列を計算する dist()

この関数はデータ行列の各行間の距離を、指定された各種距離を用いて計算し、距離行列を返す。

```
dist(x, method = "euclidean", diag = FALSE, upper = FALSE)
as.dist(m, diag = FALSE, upper = FALSE)
## クラス "dist" に対するメソッド
print(x, diag = NULL, upper = NULL, digits = getOption("digits"),
      justify = "none", right = TRUE, ...)
## クラス "dist" に対するメソッド
as.matrix(x)
```

6.2 多変量データのバイプロット biplot()

現在のグラフィックスデバイスに多変量データのバイプロットグラフを描く。

```
biplot(x, y, var.axes = TRUE, col, cex = rep(par("cex"), 2),
       xlabs = NULL, ylabs = NULL, expand = 1,
       xlim  = NULL, ylim  = NULL, arrow.len = 0.1,
       main = NULL, sub = NULL, xlab = NULL, ylab = NULL, ...)
```

7 判別分析 (discriminant analysis) (MASS パッケージ中)

今一つの多変量解析の代表的古典的手法である「判別分析」(discriminant analysis) 用の関数は R の基本パッケージ中には無い。この節では、R の推奨パッケージである MASS パッケージ (Venables and Ripley による本ののコンパニオンパッケージであり、同書中に解説がある。パッケージバンドル VR 中に含まれる 4 つのパッケージの一つで、VR をインストールすると、4 種類のパッケージが同時にインストールされる。利用に当たっては、予め命令 library(MASS) でパッケージを読み込んでおく。中にある「線形判別分析」と「2 次判別分析」関数を紹介しておく。

7.1 線形判別関数 lda()

線形判別分析 (linear discriminant analysis) を行う。この関数は、クラス内共分散行列が特異かどうか細心の注意を払う。もしグループ間分散が tol^2 未満の変数があれば、処理は停止し、その変数が定数であると報告する。これは、問題のスケーリングが不適切である結果の可能性もあるが、定数値変数の結果であることがより尤もらしい。predict.lda() で上書きされない限り、prior を指定すると判別結果に影響を与える。重みつきのグループ間共分散行列が使用されるため、ほとんどの統計パッケージと異なり、これはまた線形判別子のそれらの空間内における回転にも影響を及ぼす。したがって、最初の幾つかの線形判別子は、グループ間の差異を事前分布で与えられる重みで強調するので、データセット中でのそれらの出現度とは異なる可能性がある。

```
lda(x, grouping, prior = proportions, tol = 1.0e-4,
    method, CV = FALSE, nu, ...)
## クラス "formula" 用
lda(formula, data, ..., subset, na.action)
## クラス "data.frame" 用
lda(x, ...)
## クラス "matrix" 用
lda(x, grouping, ..., subset, na.action = na.fail)
```

7.2 線形判別分析による予測 predict()

多変量観測値を lda() に連動して判別する。そして、データを線形判別子に射影する。この関数はクラス "lda" に対する総称的関数 predict() の一つのメソッドである。これは適当なクラスのオブジェクト x に対して呼び出し predict() を行うか、または直接に x のクラスに無関係に呼び出し predict.lda(x) を行うことにより起動される。newdata 中の欠損値は、線形判別子が計算できなければ NA を返す。もし newdata が省略され、欠損値処理 na.action が欠損値の無視を指示していれば、これらは予測に対して無視される。このバージョンは線形予測子に主眼を置いているので、グループの重心の (prior で重みを付けた) 重みつき原点にある。

```
## クラス "lda" 用
predict(object, newdata, prior = object$prior, dimen,
        method = c("plug-in", "predictive", "debiased"), ...)
```

7.3 二次判別関数 qda()

qda() は二次関数 (Maharanobis 距離による楕円状の判別領域を用いる) による判別分析 (quadratic discriminant analysis) を行う。QR 分解を使い、もしグループ内共分散行列が特定のグループに対し特異なら、エラーメッセージを与える。

```
## 既定手法
qda(x, grouping, prior = proportions,
    method, CV = FALSE, nu, ...)
## クラス "formula" 用
qda(formula, data, ..., subset, na.action)
## クラス "data.frame" 用
qda(x, ...)
## クラス "matrix" 用
qda(x, grouping, ..., subset, na.action = na.fail)
```

7.4 二次判別関数による予測

この関数はクラス "qda" に対する総称的関数 predict() の一つのメソッドである。これは適当なクラスのオブジェクト x に対して呼び出し predict() を行うか、または直接に x のクラスに無関係に呼び出し predict.qda(x) を行うことにより起動される。newdata 中の欠損値は、線形判別子が計算できなければ NA を返す。もし newdata が省略され、欠損値処理 na.action が欠損値の無視を指示していれば、これらは予測に対して無視される。二次判別分析関数 qda() による判別子を用いて多変量観測値の判別を行う。

```
## S3 method for class 'qda':
predict(object, newdata, prior = object$prior,
        method = c("plug-in", "predictive", "debiased", "looCV"), ...)
```

8　多変量解析アドオンパッケージ multiv 中のオブジェクト

以下は library(help=multiv) で得られる multiv パッケージ中の関数の一覧である。使用に当たっては、まずパッケージをダウンロード・インストールし、命令 library(multiv) でライブラリを読み込む必要がある。

なお、パッケージバンドル VR 中の推奨パッケージ MASS, class 中にも多変量解析関係の関数がある。

`ca()`	対応分析 (correspondence analysis)。多変量データに対し、最初の座標が最大の慣性 (inertia) を持ち、二番目の座標は最初の座標に直交するという制約下最大の慣性を持ち、等々の基準で新しい座標系求める。主成分分析に比較すると、各行と列は関連する質量 (mass) を持ち (行と列の総和に関係する)、カイ自乗距離がユークリッド距離にとって代わる。入力データをどうコーディングするかが重要であり、これが主成分分析の入力データ変換にとって代わる
`bea()`	結合エネルギアルゴリズム (bond energy algorithm)。配列の行と列を、大きな値の配列要素の近似度を最大化するように入れ換える
`flou()`	(3 元分類) ファジーコーディング。ベクトルの単純なファジー、または区分的線形、コーディング。ベクトル中の各値は 1 (67 番目のクォンタイル以上なら) か、0 (33 番目のクォンタイル以下なら) か、0 と 1 の間で線形補間 (33, 67 番目のクォンタイル間なら) に置き換えられる
`hierclust()`	階層的クラスタリング。hclust() の代替関数
`logique()`	論理値コーディング。中央値以上、未満の値をそれぞれ 1, 0 で置き換える
`members()`	データの各クラスターへの所属関係を表示