応用プログラミング3

第9回　Rによる統計モデリング、機械学習の基礎

専修大学ネットワーク情報学部
田中健太

1．課題の振り返り

1

2. データ分析と統計
モデリング、機械学習

2

今回は、Rを使用する主な目的のひとつである、統計モデリング、機械学習を行う方法の概要を紹介します。とはいえ、この講義の目的は「R言語の基本とトレンドを知る」ことなので、機械学習アルゴリズムを多数取り上げることや、数理的な背景を解説することはせず（※）、まず今回は、組み込み関数や定番パッケージの使い方を中心に解説します。

※そもそも筆者にはできませんし。

2.1　目的変数と説明変数

- 研究、ビジネスのあらゆる場面で、**因果関係**を明らかにしたいことが多い
- **データどうしの関係性を、理解可能な式で表現するのが統計モデリング**
- 表構造のデータにおいて、結果と考える列を**目的変数**、原因と考える（複数の）列を**説明変数**と呼ぶ
- **目的変数と説明変数の間を何らかの関数でつないだものがモデル**

$$y = f(x)$$

目的変数　　説明変数

3

はじめに、最も基本的な「モデル」についての概念を整理しておきましょう。私たちがデータ分析をする時、たいていはデータどうしの（因果）関係を理解したいと考えます。データ間に原因と結果の関係を仮定するなら因果関係、そうでなければ相関関係などを考察します。

モデルは、そのようなデータ間の関係性を一般化し、（主に※）数式として表現したものです。データをモデルとして整理することで、関係性の理解が深まり、一方のデータからもう一方を予測することもできます。モデルを作成することをモデリングと呼びます。

モデリングを行う際には、データを「予測したい値（結果）」と「予測に使う値（原因）」に分けて扱います。この、「予測したい値（結果）」を目的変数（Target variable）や従属変数（Dependent variable）、「予測に使う値（原因）」を説明変数（Explanatory variables）や独立変数（Independent variables）などと呼びます。呼び方は、書籍やWeb記事の執筆者がどの学術領域に属しているかで変わり、経済・社会科学系では目的変数 / 説明変数、医学・心理学系では従属変数 / 独立変数と使い分けている印象があります（筆者の主観です）。また、情報系、近年の機械学習・AI系の文脈では、説明変数を特徴量（Features）と呼ぶことも多いです。

※最近流行のディープラーニングなどは、モデルが複雑すぎて、もはや数式としては理解不能だと思います。

2.2　統計学的アプローチと機械学習的アプローチ

- 統計学は、モデルを作成し、因果関係を理解（解釈）することを重視する
- そのため、統計学的アプローチでは、解釈性の高いモデルを作成することが求められる
- 一方で、モデルの理解はあまり重視せず、モデルを基にした予測に関心があるのが機械学習
- 機械学習的アプローチでは、解釈困難でも精度が高ければ良いモデルとみなす
- ただし、現実的には精度が高いからと解釈性が低いモデルを盲信することもないので、両者の違いはあまり厳密なものではない

4

ここで、「統計学」と「機械学習」の違いについて整理しておきましょう。といっても、明確な定義があるわけではなく、モデリングの目的・用途によって、「この分析は統計学寄りのアプローチだ」「機械学習寄りのアプローチだ」と人間が解釈するものに過ぎません。

一般的に、統計学的なアプローチでは、データに含まれる関係性、つまりモデルを理解することを目的とします。経済活動がどのように営まれるか、人間の心理がどのように表現できるかなどを考察します。一方で、機械学習的なアプローチでは、モデルの理解よりも、モデルを用いて精度の高い予測を行うことを目的とします。モデルの中身は解釈不可能なほど複雑でも、それによって正確に予測ができればそれで良い、という考え方です。

現在、特にビジネスにおいてはモデルの理解よりも予測精度が求められることが多いので、機械学習的なアプローチが主流になっています。とはいえ、いくら精度が高くても、内部がまったく理解できないモデルを信用して予測結果に従うこともないので、両者を厳密に分ける必要はなく、どちらに比重を置くかというスタンスの違いになります。

また、モデルを作成する際に、人間があらかじめ関数式を仮定して当てはめるのが統計学、式を仮定せず、コンピューター（機械）に探索（学習）させるのが機械学習、という捉え方もできます。

3. Rによる統計モデリング

5

　モデリングの概念やアプローチの説明はここまでにして、Rを使って統計モデリングを行う方法を紹介していきます。なお、ここでは統計モデリングという言葉を使いますが、機械学習と読み替えても問題ありません。

3.1　線形モデル

- 目的変数と説明変数の関係を、何らかの「線」で表現するのが（広義の）線形モデル
- よく知られた線形回帰は、目的変数（の誤差）が正規分布すると仮定し、目的変数と説明変数の関係を直線で表現したもの

$$y = ax + b$$

係数　　重み

- Rでは lm() 関数でモデリングできる

6

はじめに、目的変数 y と説明変数 x の関係を何らかの「線」で表現する(広義の) 線形モデル (Linear model) を紹介します。(狭義の) 線形モデルは、線形回帰とも呼ばれ、$y = ax + b$ といったような、直線で表現するモデルです。説明変数 x に係数 (重み) a をかけ、切片 b を加えると、目的変数 y が求められるというものです。なお、y は連続的な数値です。実際には、説明変数は複数用いることが一般的なので、$y = a_1x_1 + a_2x_2 + \cdots a_n + b$ といった式になるでしょう。説明変数を複数用いる場合、重回帰分析と呼びます。

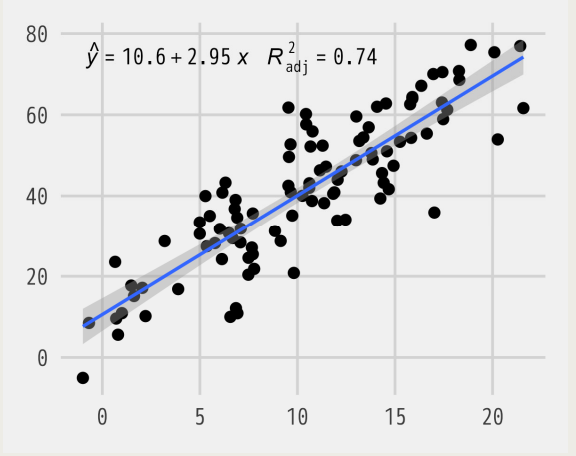
Rでは、組み込みの lm() 関数で線形回帰モデルを作成できます。lm(y ~ x, data = データフレーム名) や lm(y ~ x1 + x2 + ..., , data = データフレーム名)、lm(y ~ ., , data = データフレーム名) というように記述します。ここで、. (ドット) は、「データフレームの残りすべての列」という意味です。

線形回帰モデルの評価指標は、lm() 関数の結果をオブジェクトに代入し、summary() 関数で確認できます。まず確認すべき指標は、"Adjusted R-squared" (決定係数、補正済みR2乗など) と "p-value" (直線の傾きが0であるという仮説の検定結果) です。"Adjusted R-squared" が1に近いほど、モデルの当てはまりが良いことを意味します。"p-value" が0.05 (有意水準) 以上になることはまずないですが、もしそうなれば、y と x の間に関係はない、ということになりモデルは成立しません。

(次のページに続きます)

3.1　線形モデル

```
# シミュレーションデータの生成
b <- 10
a <- 3
x <- rnorm(100, mean = 10, sd = 5)
e <- rnorm(100, sd = 10)
y <- (a * x) + b + e
df <- data.frame(x, y)
# 線形回帰モデルの作成
lm_res <- lm(y ~ x, data = df)
summary(lm_res)
```




```
## Residuals: (一部省略)
##     Min      1Q  Median      3Q     Max
## -24.862  -5.269   0.251   6.359  23.070
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   10.580      2.033     5.2  1.1e-06 ***
## x              2.949      0.175    16.9  < 2e-16 ***
## Multiple R-squared:  0.744,  Adjusted R-squared:  0.742
## F-statistic:  285 on 1 and 98 DF,  p-value: <2e-16
```

アスタリスク＝有意
＝モデルに必要な変数

7

次に、各説明変数について、y との関係はない、という仮説を検定した結果の表を確認します。ここで、0.05よりも大きい p 値になった説明変数は、モデルに不要であるということになるので、取り除いて改めてモデルを作成します。現実的には、それを自動的に行うステップワイズ法でモデルを洗練させていきます。

- 参考1: Rでステップワイズ回帰 - 井出草平の研究ノート
  https://ides.hatenablog.com/entry/2019/08/31/171340

実際のところ、線形回帰にもいろいろな理論的な仮定があり、本来はそれらをしっかり検討したうえで適用する必要があります。理工系、情報系の授業では、そのあたりをしっかりと説明しますが、本講義では「Rでどうやるか」にフォーカスしている（と言って逃げる😜）ため、数学的な側面については、以下のサイトなどを参考にしてください。

- 参考2: 国友ら（訳）データ分析のための統計学入門
  https://www.openintro.org/go?id=os4_jp&referrer=/book/os/index.php (PDF) P. 313-
- 参考3: 総務省統計局 「高校生のための統計学習教材」 第2講　多変量データの扱い
  https://www.stat.go.jp/teacher/dl/pdf/c3learn/materials/third/dai2.pdf
- 参考4: 文部科学省 高等学校情報科「情報II」教員研修用教材 第3章 情報とデータサイエンス 前半 https://www.mext.go.jp/content/20200702-mxt_jogai01-000007843_004.pdf P. 126-135
- 参考5: 総務省統計局 データサイエンス・スクール レベル別教材 上級向けテキスト
  https://www.stat.go.jp/dss/getting/pdf/upper.pdf P. 53-63

3.2　モデルに基づく予測

- 作成したモデルをもとに、説明変数だけを与えて予測ができる
- Rでは多くの場合 predict() 関数が利用できる
- モデル作成に使用したデータから予測した値は fitted() 関数で取得できる

```
fitted(lm_res)
##      1      2      3      4      5      6      7      8      9     10     11
## 39.525 60.605 36.670 69.830 42.270 12.676 61.982 26.366  8.565 29.933 45.474 ...
```

```
x2 <- rnorm(100, mean = 11, sd = 6)
df2 <- data.frame(x = x2)
predict(lm_res, df2)
##      1      2      3      4      5      6      7      8      9     10     11
## 63.112 55.890 -12.254 35.052 49.105 35.288 69.977 47.867 34.115 49.467 26.874 ...
```

モデルと同じ列名にする必要がある

8

モデルを作成したら、目的変数のわからない説明変数だけを与えて、予測することができます。lm() 関数をはじめ、Rで利用される多くの統計モデリング、機械学習のための関数、パッケージでは、predict() 関数にモデルオブジェクトと新しいデータを与えると、予測値を得ることができます。

基本的には、predict(モデルオブジェクト, 予測対象データ) という書式です。なお、予測対象のデータは、モデル作成時のデータと同じ構造（列数、列名、型など）である必要があります。

実際には、それぞれのパッケージなどで、引数として与えられるオブジェクトの型に応じた処理が実装されて呼び出されていますが、ユーザーとしてはそれを意識することなく、「予測するなら predict()」というように使うことができます。

また、モデル作成時に与えたデータを用いて予測した値は、fitted() 関数で取得できます。これを使い、実測値（正解）と予測値の誤差などを評価できます。

3.3　非線形モデル

- 世の中には直線よりも曲線のほうが当てはまりがよい関係もある
- 非線形回帰は、目的変数と説明変数の関係を直線以外で表現したモデルの総称（非線形だが線形）
- $y = a + bx + cx^2 + dx^3$ など
- Rでは poly() 関数や nls() 関数でモデリングできる

```
x <- 1:100
y <- (x + x^2 + x^3) + rnorm(length(x), mean = 10, sd = mean(x^3) / 4)
y <- (y + abs(min(y))) / 1000

formula <- y ~ poly(x, 3, raw = TRUE)
poly_res <- lm(formula)
```

9

続いて、非線形モデル (Nonlinear model) を紹介します。非線形モデルは直線以外の、何らかの曲線を使って当てはめを行います。例えば、説明変数を2乗、3乗した値を用いる多項式モデルなどがあります。他にも、$y = ax^b$ の累乗モデルや、$y = ab^x$ の指数モデル、ゴンペルツ曲線など、名前の付いた曲線を当てはめるモデルがあります。極論、自分で式を定義して自由にモデリングできますが、一般的には、先行研究などを参照し、妥当なモデルを選択します。

Rでは、多項式モデルは poly() 関数で作成します。lm(poly(目的変数 ~ 説明変数, 次数, raw = TRUE / FALSE)) といったように記述します。プログラム例では、先に式を文字列としてオブジェクトに格納しています。raw オプションは、TRUE / FALSE でそれぞれ利点と注意点があるので、詳しくは以下のURLを参照してください。

- 参考1: Rを使ったPolynomial regressionの実施方法（polyの説明） - 獣医 x プログラミング https://meknowledge.jpn.org/2021/01/15/rpolynomial-regression-poly/

他のモデルについては、nls() 関数で作成します。nls(式, start = c(初期値)) と記述します。start オプションに、使用する変数の初期値を指定します。以下の参考URLでは、「初期値は、データ解析者の勘と経験に頼るのが殆どである」と書かれています。ここでは、指数モデルに従うデータを生成し、nls() 関数でモデリングした例を示しています。

- 参考2: Rと非線形回帰分析 - データ解析・マイニングとR言語 https://www1.doshisha.ac.jp/~mjin/R/Chap_17/17.html

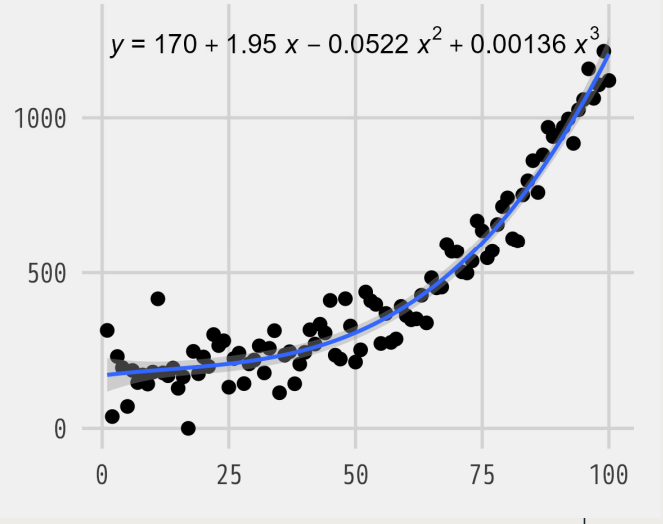
3.3　非線形モデル

```
summary(poly_res)
## Residuals:
##     Min      1Q  Median      3Q     Max
## -194.59  -48.73    1.81   44.27  229.06
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              169.760586  29.147763    5.82  7.6e-08 ***
## poly(x, 3, raw = TRUE)1    1.954515   2.486869    0.79   0.4338
## poly(x, 3, raw = TRUE)2   -0.052232   0.057061   -0.92   0.3623
## poly(x, 3, raw = TRUE)3    0.001362   0.000371    3.67   0.0004 ***
## Residual standard error: 70.2 on 96 degrees of freedom
## Multiple R-squared:  0.946,  Adjusted R-squared:  0.945
## F-statistic:  566 on 3 and 96 DF,  p-value: <2e-16
```

10

3.3　非線形モデル

ggpmiscはグラフィックス
中に回帰式などを書き込む
ためのパッケージ



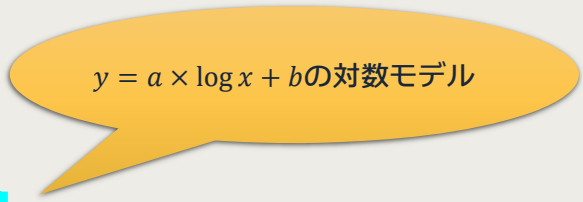

```
library(ggplot2)
library(ggpmisc)

df <- data.frame(x, y)
ggplot(df, aes(x = x, y = y)) +
    geom_point(size = 3) +
    geom_smooth(method = "lm", formula = formula) +
    stat_poly_eq(formula = formula,
                 aes(label = paste(after_stat(eq.label))),
                 parse = TRUE, size = 5) +
                 ylim(0, 1300)
```

11

- 参考: 【R】ggpmisc – DiNOV株式会社 https://www.dinov.jp/?p=535

3.3　非線形モデル

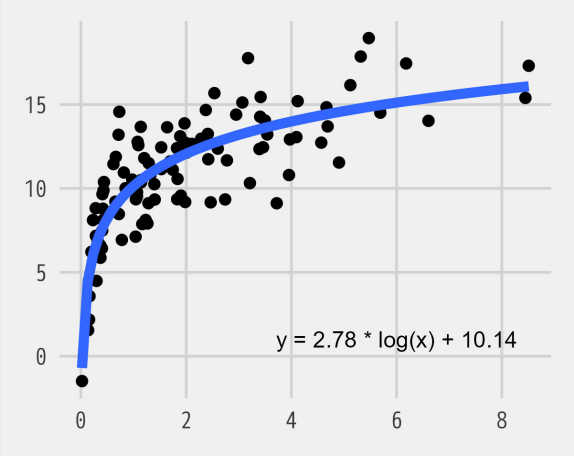
```
b <- 10
a <- 3
x <- rexp(100, rate = 0.5)
e <- rnorm(100, sd = 2)
y <- (a * log(x)) + b + e
df <- data.frame(x, y)

formula <- y ~ a * log(x) + b
nls_res <- nls(formula, start = c(a = 1, b = 1), data = df)

summary(nls_res)
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## a    2.776      0.186    14.9   <2e-16 ***
## b   10.142      0.206    49.1   <2e-16 ***
## Residual standard error: 1.99 on 98 degrees of freedom
## Number of iterations to convergence: 1
## Achieved convergence tolerance: 1.14e-09
```



12

3.3　非線形モデル

```
args <- list(formula = formula, start = list(a = 1, b = 1))
formula_label <- paste0("y = ", round(nls_res$m$getAllPars()[1], 2), " *
log(x) + ", round(nls_res$m$getAllPars()[2], 2))
ggplot(df, aes(x = x, y = y)) +
    geom_point(size = 3) +
    stat_smooth(method="nls", se=FALSE, method.args=args, linewidth=3) +
    annotate(geom = "text", x = 6, y = 1, label = formula_label, size = 5)
```

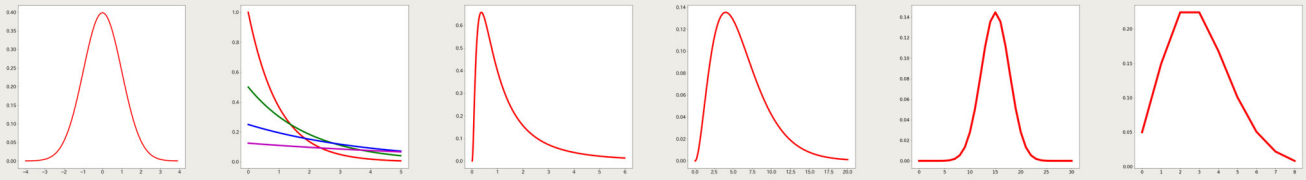


13

- 参考: RPubs - ggplot2で回帰直線を引く https://rpubs.com/uri-sy/ggplot2_stat_smooth

3.4　一般化線形モデル

- 目的変数の（誤差）分布に正規分布以外のさまざまな分布を仮定できる
- リンク関数を用いて、正規分布以外の分布に従うデータを正規分布に変換する
- Rでは `glm()` 関数でモデリングできる
- `family` オプションに誤差分布を指定する
- `link` オプションにリンク関数を指定する



正規分布　指数分布　対数正規分布　ガンマ分布　二項分布　ポアソン分布

14

ここまで紹介した線形モデルも非線形モデルも、「誤差（残差）は正規分布に従う」という仮定を置いています。しかし、世の中のデータは必ずしも、（誤差が）正規分布に従うものばかりではありません。例えば、コールセンターに1時間で何回電話がかかってくるか、というデータは、1回、2回…という離散値しか取りませんので、ポアソン分布を仮定することが妥当でしょう。

一般化線形モデル（GLM; Generalized linear model）は、任意の確率分布に従う説明変数と目的変数の関係を、リンク関数 g を用いて、$g(y) = a_1x_1 + a_2x_2 + \cdots a_nx_n$ と線形モデルに変換して扱うものです。リンク関数 $g(y)$ は、「ある分布の形を正規分布に変換する関数」と言え、一般に対数 $\log$ や逆関数 $\frac{1}{y}$ などが使われます。

Rでは、GLMによるモデリングは組み込みの `glm()` 関数で行えます。`glm(式, family = 分布名(link = "リンク関数"))` といった書式で記述します。分布名 に `poisson` や `Gamma` などを指定し、`link` に `"log"` や `"inverse"` を指定します。`link = "identity"` とすると、変換を行いません。

結局のところ、GLMにおいてどのような誤差分布を仮定し、どのようなリンク関数を指定するかは、実データを観察し、先行研究を調査し、経験と勘で決めていくものになるでしょう。

3.4　一般化線形モデル

```
data(airquality)
aq_df <- airquality %>% select(-Month, -Day) %>% drop_na()

glm_res <- glm(Ozone ~ Solar.R + Wind + Temp, data = aq_df,
   family = Gamma(link = "log"))
summary(glm_res)
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7096  -0.4081  -0.0914   0.2413   1.1798
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.451357   0.531785    0.85  0.39791
## Solar.R      0.002104   0.000535    3.93  0.00015 ***
## Wind        -0.065899   0.015095   -4.37  2.9e-05 ***
## Temp         0.043029   0.005848    7.36  4.0e-11 ***
##     Null deviance: 71.950  on 110  degrees of freedom
## Residual deviance: 25.863  on 107  degrees of freedom
## AIC: 925.9
```

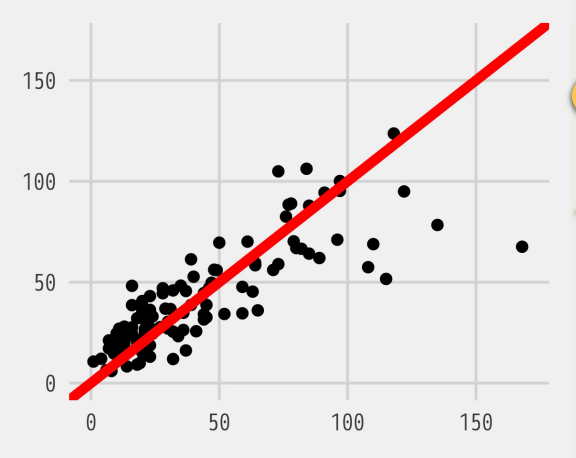
GLMでは R^2 ではなく、
Devianceで評価する

15

- 参考: 30分だけでは決してよくわからない とてもとても難しい 一般化線形モデル with R
  https://cogpsy.educ.kyoto-u.ac.jp/personal/Kusumi/datasem13/shrasuna1.pdf

3.4　一般化線形モデル

```
Ozone_pred <- fitted(glm_res)
df <- data.frame(Ozone = aq_df[["Ozone"]], Ozone_pred)

ggplot(df, aes(x = Ozone, y = Ozone_pred)) +
    geom_point(size = 3) +
    geom_abline(intercept = 0, slope = 1, linewidth = 3, colour = "red") +
    lims(x = c(0, 170), y = c(0, 170))
```



重回帰なので、グラフは
実測値（x）と予測値（y）
のズレをあらわす

16

3.5　ロジスティック回帰モデル

- 目的変数がカテゴリー変数である場合のモデル
- 特に2値（1 / 0）である場合に使用する
- リンク関数にロジット $\log\left(\frac{p}{1-p}\right)$ を指定する
- glm(..., family = binomial(link = "logit"))

```
set.seed(340)
b <- 10
a <- 3
x <- rnorm(100, mean = 10, sd = 5)
e <- rnorm(100, sd = 5)
y <- (a * x) + b + e
y <- if_else(y >= mean(y), 1, 0)
logi_res <- glm(y ~ x, family = binomial(link = "logit"))
```

17

GLMにおいてよく用いられるモデルに、ロジスティック回帰があります。ロジスティック回帰は、目的変数が2値 (0 / 1) であったり、割合 (パーセンテージ) である場合に、誤差分布に二項分布を仮定し、リンク関数にロジットを用いるモデルです。ロジットはスライドに示す式で定義されます。

Rでは、glm() 関数に family = binomial(link = "logit") と指定します。このモデルで得られる予測値は、0から1の範囲になり、確率として扱えます。例えば、2値変数を「買った: 1」「買わなかった: 0」などと対応付けた場合、「購入確率」と捉えることができます。このような、離散的な値 (ラベル) のモデリングを、分類タスク (問題) と言います。

ロジスティック回帰の数理的な背景については、以下のURLなどを参考にしてください。

- 参考1: 国友ら (訳) データ分析のための統計学入門 https://www.openintro.org/go?id=os4_jp&referrer=/book/os/index.php (PDF) P. 381-
- 参考2: 数理・データサイエンス・AI教育強化拠点コンソーシアム 応用基礎レベル教材 1. データサイエンス基礎 1-4. データ分析 http://www.mi.u-tokyo.ac.jp/pdf/1-4_data_analysis.pdf P. 22-33

3.5　ロジスティック回帰モデル

```
summary(logi_res)
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1508  -0.0889   0.0115   0.2038   2.5313
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -11.347      3.037   -3.74  0.00019 ***
## x              1.206      0.309    3.90 0.000097 ***
##     Null deviance: 137.628  on 99  degrees of freedom
## Residual deviance:  39.279  on 98  degrees of freedom
## AIC: 43.28
```

18

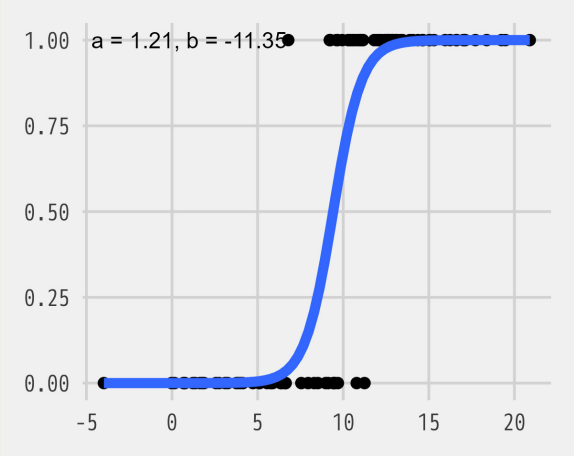
ロジスティック回帰モデルの精度を評価する指標として、Devianceがあります。Devianceは、尤度を用いる指標です。Devianceは、すべての説明変数を使って実測値を表現する、いわば完璧なモデルと、今回作成したモデルの尤度を比較するものです。完璧なモデルは、このデータにしか当てはまらない、実用性はないモデルですが、精度は100%です。これと、未知のデータにも適用できる一般性を持ったモデルのズレを評価します。そのため、Devianceはズレが小さい、ゼロに近いほど当てはまりが良いモデルということになります。

尤度は、データがあるパラメーターを持った確率分布からサンプリングされたものと言えるかをあらわす指標です。尤度の "ゆう" は、「もっともらしい」という意味ですが、つまり、このデータはこの分布からサンプリングされたと考えることが、どの程度もっともらしいか、という指標です。手元のデータの分布と、そのデータの母集団として仮定した分布を比較し、両者が似通っているほど、尤度は大きくなります。これを統計モデルの評価指標として考えると、モデルから得られた予測値の分布と、正解である実測値の分布を比較して、尤度が高くなれば、モデルは実測値の特徴をよく表現できている、ということになります。

実際には、尤度は式の都合上計算が複雑なので、対数を取って計算しやすくした対数尤度を使うことが一般的です。

3.5　ロジスティック回帰モデル

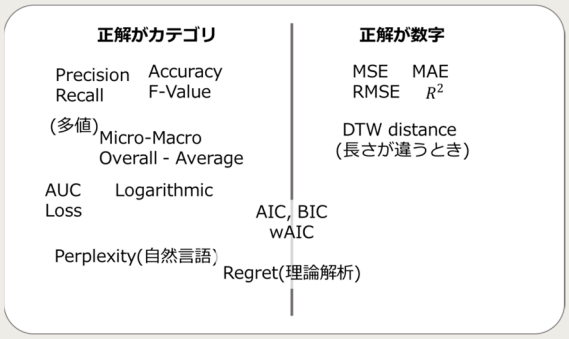
```
df <- data.frame(x, y)
formula_label <- paste0("a = ", round(logi_res$coefficients[2], 2),
   ", b = ", round(logi_res$coefficients[1], 2))
ggplot(df, aes(x = x, y = y)) + geom_point(size = 3) +
   geom_smooth(method = "glm", method.args =
   list(family = binomial(link = "logit")), se = FALSE, linewidth = 3) +
    annotate(geom = "text", x = 1, y = 1, label = formula_label, size = 5)
```



19

3.6　モデルの評価

- lm(), glm() はじめ多くのモデリング関数で作成したモデルは、summary() 関数で評価できる
- 回帰モデルの評価指標としてR^2、（決定係数）やRMSEが用いられる
- 分類モデルの評価指標として正解率、ROC曲線、AUCなどが用いられる
- 回帰、分類いずれにおいても使われる指標としてAICなどがある



機械学習で使う指標総まとめ（教師あり学習編） - プロクラシスト
https://www.procrasist.com/entry/ml-metrics

20

ここまで紹介してきたように、Rにおける線形・非線形モデルの結果は、summary() 関数にオブジェクトを与えることで、整理されて出力されます。そこで出力される評価指標を見ることで、作成したモデルの良し悪しが確認できます。

一般に、回帰モデルの評価指標として、R^2 (決定係数) や、機械学習寄りのアプローチの場合は、RMSEなどが用いられます。

分類モデルにおいては、正解率や適合率、再現率などの数値や、ROC曲線やAUCなどのグラフィックスを伴う評価指標が使われます。

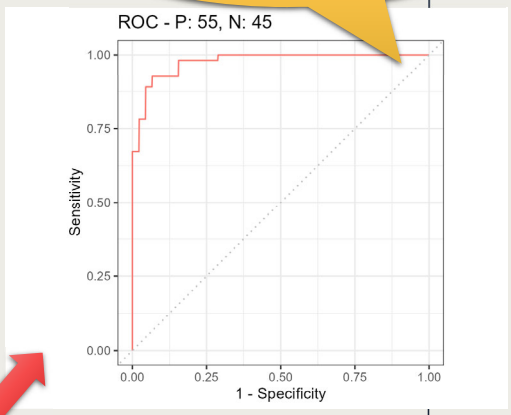
また、両方に共通する指標として、AIC (Akaike's information criterion; 赤池情報量規準) などがあります。

多くの評価指標は組み込み関数には対応するものがないので、次回紹介する機械学習フレームワークに含まれる関数などを使って出力します。

3.6　モデルの評価

```
summary(lm_res) # 線形回帰モデルの評価
## Residuals:
##     Min      1Q  Median      3Q     Max
## -24.862  -5.269   0.251   6.359  23.070
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   10.580      2.033     5.2  1.1e-06 ***
## x              2.949      0.175    16.9  < 2e-16 ***
## Residual standard error: 9.2 on 98 degrees of freedom
## Multiple R-squared:  0.744,  Adjusted R-squared:  0.742
## F-statistic:  285 on 1 and 98 DF,  p-value: <2e-16

# ロジスティック回帰モデルのROC曲線を描画
library(precrec)
logi_mmdata <- mmdata(fitted(logi_res), y)
logi_curve <- evalmod(logi_mmdata)

autoplot(logi_curve, "ROC") + theme(legend.position = "none")
```

点線＝50%＝偶然
左上に寄るほど高精度



21

このページの内容ではないですが、スペースがあるので。前のページで触れた、回帰モデルの評価指標のひとつであるRMSEは、いくつかの関数を組み合わせて簡単に求めることができます。RMSEは「平方根平均二乗誤差」と日本語訳されますが、まさにその通りの計算をすればよいだけです。まず、データ1件ずつについて、実測値と予測値の差を取ります。そして、その差を2乗します。さらに、差を平均し、その結果の平方根を取ったものがRMSEです。

```
y_pred <- predict(lm_res, df) # 本来は別途用意したテスト用データを使う
y_diff <- df[["y"]] - y_pred
y_squared <- y_diff ** 2
y_mse <- mean(y_squared)
y_rmse <- sqrt(y_mse)
y_rmse
[1] 9.942332
```

また、分類タスクにおける正解率も容易に求められます。ロジスティック回帰の予測値については、predict() 関数に type = "response" と指定すると、(2値分類の場合) 0から1の確率が得られるので、それを四捨五入 (ないし任意の閾値で切り上げ / 切り下げ) することで、目的変数と対応する予測値が求められます。それを比較し、同一であれば TRUE= 1、異なっていれば FALSE = 0が得られるので、TRUE の合計をデータ数で割れば正解率になります。

```
y_resp <- predict(logi_res, df, type = "response") # 上記dfとは中身が異なる
y_pred <- dplyr::if_else(y_resp >= 0.5, 1, 0)
y_diff <- df[["y"]] == y_pred
y_rate <- sum(y_diff) / nrow(df)
y_rate
[1] 0.89
```

4. Rによる機械学習

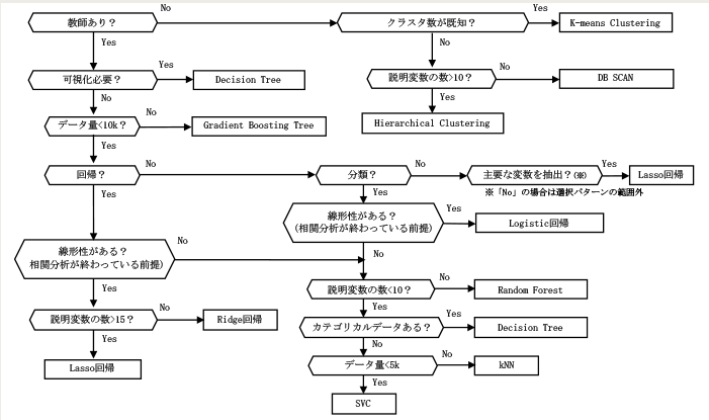
22

　ここからは、機械学習寄りのアプローチで、モデリングするためのアルゴリズムを紹介します。例によって、数理的な背景は説明しませんので、それぞれ参考として紹介するURLなどを参照してください。

　なお、ここまでモデルを作成する、あるいはモデリングと書いてきましたが、機械学習の文脈では、同じことを「学習する」とも言います。そこで、ここからは「〇〇アルゴリズムで学習したモデル」といったように表現します。同様に、説明変数と書いてきましたが、ここからは特徴量という言葉を使います。

4.1　さまざまな機械学習アルゴリズム

- 機械学習はモデルをコンピューターが探索し作成する仕組み
- 解釈性よりも精度を追及する傾向が強い
- 機械学習の文脈では説明変数を特徴量と言うことが多い



出典：晦日ら，2020，Kaggleカーネルを参照した機械学習アルゴリズムの選択／適用パターンの抽出と習得，デジタルプラクティス，Vol.11 No.1，情報処理学会
https://www.ipsj.or.jp/dp/contents/publication/41/S1101-1908.html

23

ここでは、機械学習の領域で、古くから広く使われている、「定番」と言えるようなものを紹介します。まずはこのあたりのアルゴリズムをRで利用できるようになってから、より高度なものに挑戦してみてください。

なお、実務においてどのようなデータと問題に対してどのようなアルゴリズムを選ぶべきか、ということについて、以下の論文があります。

- 参考: 晦日ら, 2020, Kaggleカーネルを参照した機械学習アルゴリズムの選択／適用パターンの抽出と習得, デジタルプラクティス, Vol.11 No.1, 情報処理学会, https://www.ipsj.or.jp/dp/contents/publication/41/S1101-1908.html

著者はいずれも企業の研究者です。本来は「このアルゴリズムにはこういう利点があるが、一方でこんな注意点があり、またこちらのアルゴリズムにはゴニョゴニョ…」と言い切らないのが研究者的な態度でしょうが、現実にはIT企業の開発現場でもAI案件が盛んで、すべての案件に専門家が関わることが難しい状況です。その中で、必ずしも専門家ではないSE（システム・エンジニア）が迷わないような「ハウツー」が求められています（SEは、研修で上述のような科学的に真摯な＝煮え切らない説明をすると怒ります…）。
おそらく、この論文もそのようなニーズから書かれたものでしょう。

4.2　決定木

- 目的変数と特徴量の関係を木構造で表現するモデル
- 特徴量空間（散布図）上で、目的変数がより明確に区別できる閾値を不純度をもとに決定し、分割する
- rpartパッケージなどを使用する

```
library(rpart)
library(partykit)
library(ggparty)

res_dt <- rpart(Species ~., data = iris)
```

24

はじめに、決定木 (Decision tree) を取り上げます。決定木は、線形・非線形モデルとは異なり、目的変数と特徴量の関係を、木構造で表現します。木構造は、「枝」ごとに、ある特徴量の値が基準以上か以下かを判断し、枝分かれするもので、最上部が「根」です。
根から枝を辿っていくことで、「このような条件に合致する特徴量であれば、目的変数の値はこうなる」という結果に行き着きます。枝の分岐を決める際には、不純度という概念と、その評価指標であるGini係数が用いられます。理論的な背景については、以下のURLを参照してください。

- 参考: 決定木入門編 「ウォーリーを探せ」から考える不純度の考え方 - Np-Urのデータ分析教室 https://www.randpy.tokyo/entry/decision_tree_theory (ウォーリーは出てこないですが…)

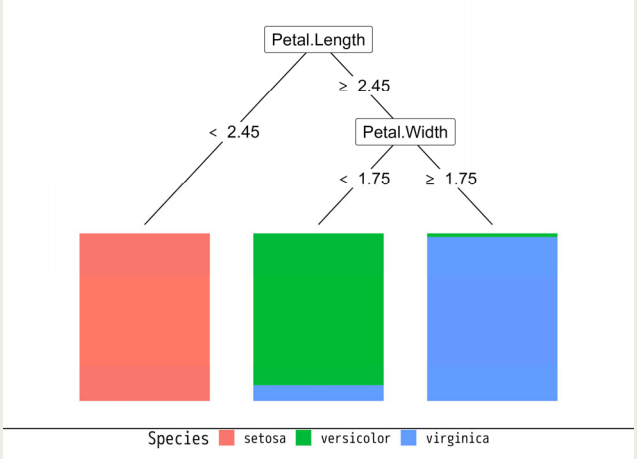
Rで決定木を作成するには、rpartパッケージが使われます。また、作成したオブジェクトを他のパッケージ、関数から扱いやすい形式に変換するためのpartykitパッケージも合わせて使われます。決定木を可視化するためには、rpart.plotパッケージやggpartyパッケージが使われます。

決定木を作成するには、rpartパッケージの `rpart()` 関数を使います。`rpart(式, data = データフレーム名)` とします。作成したオブジェクトは、そのままではやや扱い辛いので、partykitパッケージの `as.party()` 関数で変換します。変換したオブジェクトは、ggpartyパッケージの `ggparty()` 関数などで可視化できます。

プログラム例は、irisデータの `Species` 列を目的変数、他の列を特徴量として決定木を作成、描画するものです。

4.2　決定木

```
ggparty(as.party(res_dt)) +
  geom_edge() +
  geom_edge_label() +
  geom_node_splitvar() +
  geom_node_plot(gglist = list(geom_bar(aes(x = "", fill = Species),
         position = position_fill()), xlab("Species")),
                shared_axis_labels = TRUE, legend_separator = TRUE)
```



25

- 参考1: R ggpartyパッケージを用いた決定木の可視化 | トライフィールズ
  https://www.trifields.jp/visualization-of-decision-tree-using-ggparty-in-r-3499
- 参考2: ggparty: Graphic Partying
  https://cran.r-project.org/web/packages/ggparty/vignettes/ggparty-graphic-partying.html

4.3　ランダムフォレスト

- 決定木には過学習しやすいという注意点がある
- 精度を高め、過学習を回避するために木をたくさん生やすアプローチ
- 1本1本の木にランダムに少数の特徴量を与え、シンプルなモデルを多数作成する
- 構造の異なる木を組み合わせることで、高精度なモデルになる
- その代わりに、グラフィックスとして表現することは困難
- randomForestパッケージ、rangerパッケージなどを使用する

26

次に、ランダムフォレストを取り上げます。ランダムフォレストは、決定木を拡張したものです。決定木は、1本の木で目的変数と特徴量のあらゆる関係を表現するため、データ中に数件しか出現しないようなレアなパターンも学習してしまいます。そのため、学習用のデータにはぴったりフィットするのに、実務に適用すると思ったような精度が出ない、という過学習 (Overfitting) が起こりやすいことが知られています。また、精度を高めるためにモデルが複雑化すると、可視化の利点がなくなる (見てもわからない) という問題もあります。

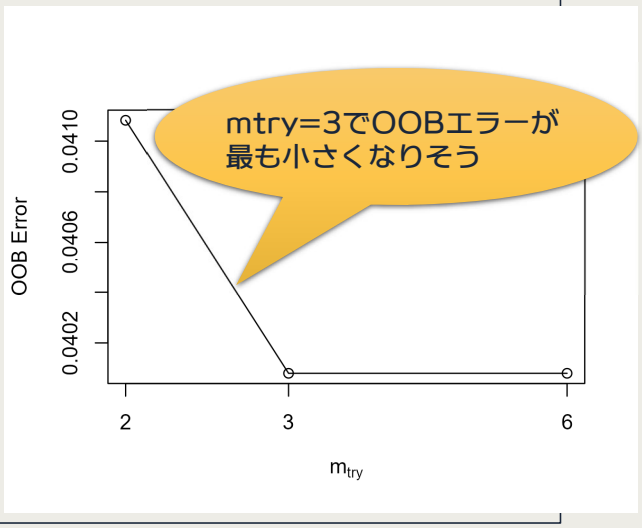
そこで、過学習を回避するために複数 (大量) のシンプルで構造が異なる決定木を組み合わせて学習する、というのがランダムフォレストのコンセプトです。ランダム、というのがポイントで、木ごとに使用する特徴量の数や種類 (列) を変えることで、さまざまな観点から目的変数と特徴量の関係を表現するモデルが作成されます。最終的に、それぞれの木が出力した予測結果を、回帰の場合は平均、分類の場合は多数決で出力します。

Rでは、randomForestパッケージが広く使われています。また、やや新しいパッケージとして、rangerパッケージも使われています。いずれにしても、関数の書式自体はシンプルで、そこに指定するハイパーパラメーター (設定項目) のチューニングがモデルの精度を左右します。ここでは、ハイパーパラメーターのひとつである`mtry`の値を、`tuneRF()` 関数を使って事前に最適化し、その値を指定して `randomForest()` 関数で学習しています。木の数は、100本で固定していますが、多すぎても計算の無駄なので、何度も繰り返して使うモデルであれば、適宜調整します。

4.3　ランダムフォレスト

```
df <- read_csv("./pseudo_sake_data.csv", col_types = cols(容器 =
col_factor(), タイプ = col_factor(levels = c("一般酒", "吟醸酒", "純米酒",
"本醸造酒"))))

library(randomForest)
best_mtry <- tuneRF(x = df[, 1:ncol(df) - 1], y = pull(df[, ncol(df)]),
doBest = TRUE)

## mtry = 3  OOB error = 4.01%

## Searching left ...

## mtry = 2     OOB error = 4.11%

## -0.025 0.05

## Searching right ...

## mtry = 6     OOB error = 4.01%

## 0 0.05
```

pull() はtibbleからベクトルを取り出す関数



27

大規模な機械学習タスクになると、少しでも処理速度が速いアルゴリズムや関数が求められます。randomForestパッケージとrangerパッケージの処理速度を比較した記事が以下にあります。rangerパッケージのほうが、25%ほど処理速度が速いという結果だったようです。

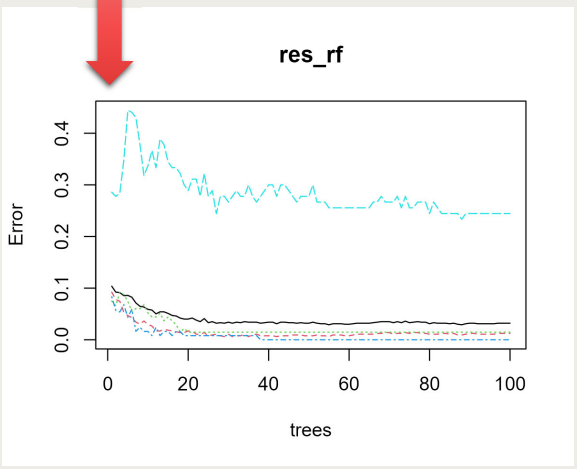
- 参考: The comparison between randomForest and ranger - CSDS
  https://arikuncoro.xyz/blog/data-science/r-python-sql-linux/the-comparison-between-randomforest-and-ranger/

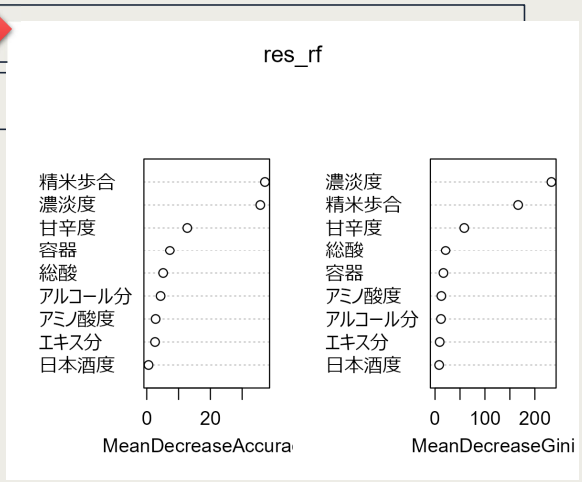
4.3　ランダムフォレスト

```
res_rf <- randomForest(タイプ ~., data = df, mtry = best_mtry$mtry,
   ntree = 100, importance = TRUE)
```

```
varImpPlot(res_rf)
```

```
plot(res_rf)
```





28

ランダムフォレストについて説明する文書ではたいてい、木がたくさん横並びになった絵が示されますが、それはあくまでイメージで、実際にはランダムフォレストの可視化は困難です。

そこで、木構造を可視化する代わりに、どの特徴量が目的変数と強い関係にあるかを示す、特徴量（変数）重要度（Variable importance）が使われます。数値として見たり、大小を棒グラフなどであらわします。randomForestパッケージの `varImpPlot()` 関数に結果のオブジェクトを与えると、グラフィックスが描画されます。

また、`plot()` 関数にオブジェクトを与えると、木の本数と精度の関係をプロットしてくれます。折れ線グラフは、それぞれのラベルの誤り率と、Out-of-bag error（OOB）をあらわします。

ランダムフォレストの概念や数理的な背景については、以下を参照してください。

- 参考1: ランダムフォレスト（Random forest）| スタッツギルド株式会社
  https://www.stats-guild.com/analytics/12869
- 参考2: Random Forestで計算できる特徴量の重要度 - なにメモ
  https://alfredplpl.hatenablog.com/entry/2013/12/24/225420
- 参考3: ランダムフォレスト(Random Forests, RF)～アンサンブル学習で決定木の推定性能を向上！～ | データ化学工学研究室(金子研究室)@明治大学 理工学部 応用化学科
  https://datachemeng.com/randomforest/

4.4　クラスタリング

- データ（行）間の類似度（ユークリッド距離など）を計算し、グループ化するアプローチ
- 目的変数がないので、教師なし学習と呼ばれる
- データどうしの類似度を一対で比較する階層的クラスタリング（Ward法など）と、任意に決めたクラスタ数に収束させる非階層的クラスタリング（k-means 法など）がある
- 組み込みの hclust() 関数や kmeans() 関数が利用できる

29

今回の講義の最後に、クラスタリングを紹介します。クラスタリングは、データ（行）間の類似度（距離）を求め、それをもとに「似た」データをグループ化する手法です。目的変数はなく、教師なし学習と呼ばれるアプローチの一種です。マーケティングなどにおいて、顧客をグループ分けして、それぞれに対してアプローチしたい時などに使われます。

クラスタリングには、大きく2つの手法があります。階層的クラスタリングと非階層的クラスタリングです。階層的クラスタリングは、データを一対比較して、類似度を算出し、全ての組み合わせの結果をもとにグループ化します。グループ化の過程が視認でき、後からいくつのクラスタに分けるか検討できることが利点ですが、計算量が多いという注意点があります。距離の求め方などにより、複数の方法が知られています。

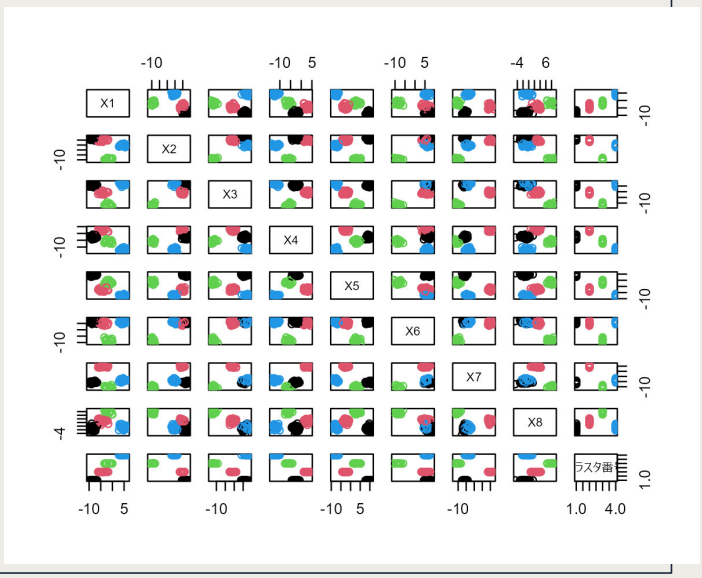
非階層的クラスタリング（事実上k-means法一択）は、クラスタの数を先に決めて、ランダムに配置したクラスタの中心点と各データとの距離から所属するクラスタを決め、それによって中心点の位置が移動し、それによって所属するクラスタが変わり…ということを繰り返し、最終的な中心点の座標と所属を決定する仕組みです。計算量が少なく、高速に処理できるため、現在の主流です。

Rでは、組み込みの hclust()（階層的クラスタリング）関数、kmeans()（非階層的クラスタリング）関数を使い、容易にクラスタリングができます。hclust() 関数には、先に dist() 関数でデータ間の距離を計算し、その結果を与えます。kmeans() 関数には、数値データからなるデータフレームを与えて実行します。結果は、それぞれ数値でも確認できますが、可視化すると理解しやすいでしょう。非階層的クラスタリングについては、factoextraパッケージの fviz_cluster() 関数を使うと、ggplot2を用いた見栄えの良いグラフィックスが得られます。

4.4　クラスタリング

```
## k-means法によるクラスタリング
res <- kmeans(df[, 1:8], centers = 4)

df[["クラスタ番号"]] <- res[["cluster"]]
pairs(df, col = df[["クラスタ番号"]])
```

各行が何番のクラスタに
属するかが返ってくる



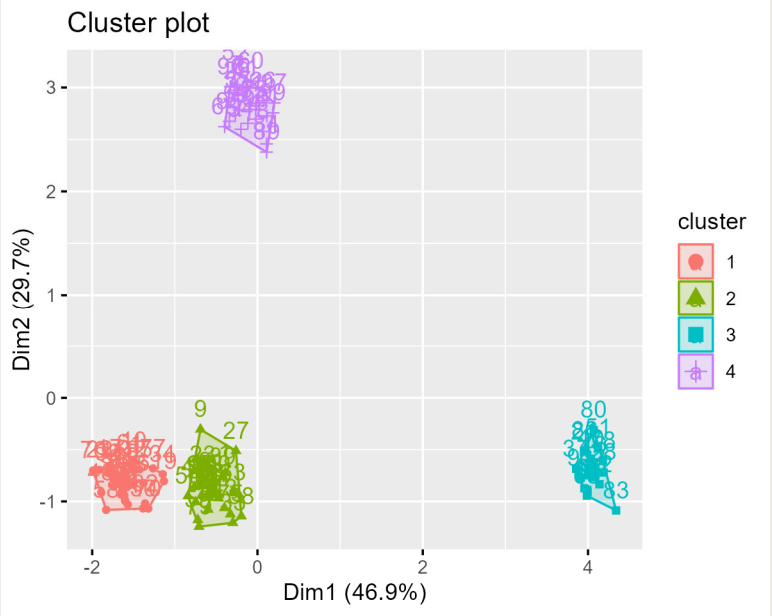
30

クラスタリングの数理的な背景については、以下のURLを参照してください。

- 参考1: 文部科学省 高等学校情報科「情報II」教員研修用教材 第3章 情報とデータサイエンス 後半 https://www.mext.go.jp/content/20200609-mxt_jogai01-000007843_007.pdf P. 152-156
- 参考2: 数理・データサイエンス・AI教育強化拠点コンソーシアム 応用基礎レベル教材 1. データサイエンス基礎 1-4. データ分析 http://www.mi.u-tokyo.ac.jp/pdf/1-4_data_analysis.pdf P. 11-12
- 参考3: 数理・データサイエンス・AI教育強化拠点コンソーシアム リテラシーレベル教材 4. オプション 4-9. データ活用実践 (教師なし学習) http://www.mi.u-tokyo.ac.jp/consortium2/pdf/4-9_literacy_level_note.pdf P. 47-55

4.4　クラスタリング

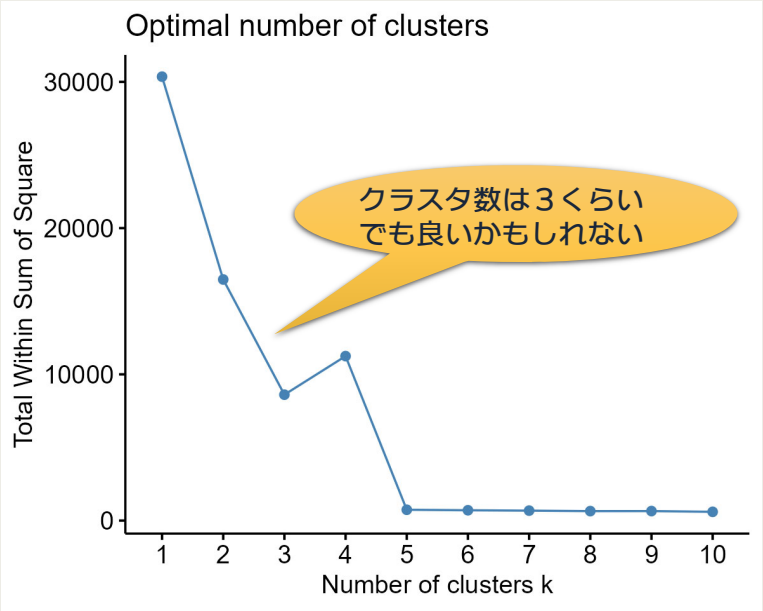
```
library(factoextra)
## クラスタリング結果の描画
fviz_cluster(res, data = df[, 1:8])
```



31

4.4　クラスタリング

```
## 最適なクラスタ数の判断
fviz_nbclust(df[, 1:8], kmeans, method = "wss")
```



32

5. まとめ

33

5.1　今日の内容

- 統計モデリングの概念
- 統計学的アプローチと機械学習的アプローチ
- 線形モデルと非線形モデル
    - 一般化線形モデル
- さまざまな機械学習アルゴリズム
    - 決定木
    - ランダムフォレスト
    - クラスタリング

34

5.2　次回までの課題

- RStudio Cloudプロジェクト内の `09_stats_ml_exercise.R` について、指示に従ってプログラムを作成してください
- 編集したファイルは、ファイル一覧でチェックを入れ、[more] メニューから [Export] を選択し、[Download] ボタンを押してダウンロードしてください
- ダウンロードしたファイルを、Classroomの課題ページから提出してください
- **提出期限: 2023-11-27 23:59**

35