

**2021.12.18.SAT**

# R AnalyticFlowでローコード分析

株式会社ef-prime

鈴木 了太

# 自己紹介

## ■ 鈴木 了太

- 株式会社ef-prime代表
- Rユーザー歴約20年
  - ・ 開発CRANパッケージ: pvclust
  - ・ 書籍: Useful R 10(共立出版、第IV部)

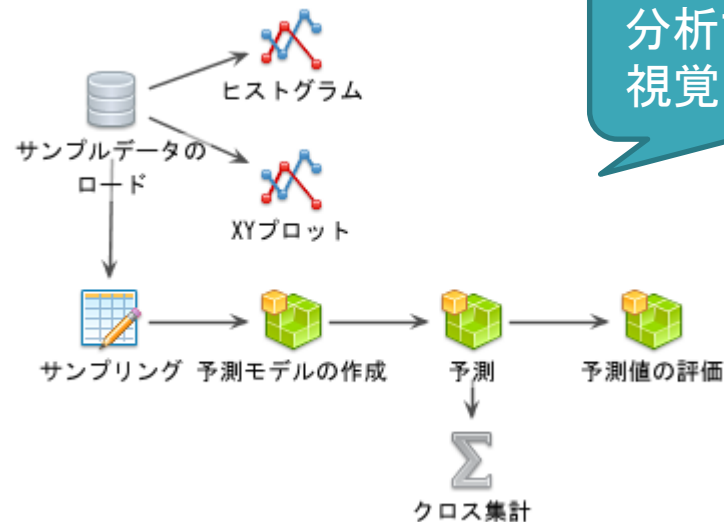
## ■ 株式会社ef-prime(エフプライム)

- データ分析コンサルティング(主に企業向け)
- 2006年設立、多くのプロジェクトでRを活用
- データ分析ツールを開発・公開

# R AnalyticFlowとは

## ■ データ解析のためのR GUI

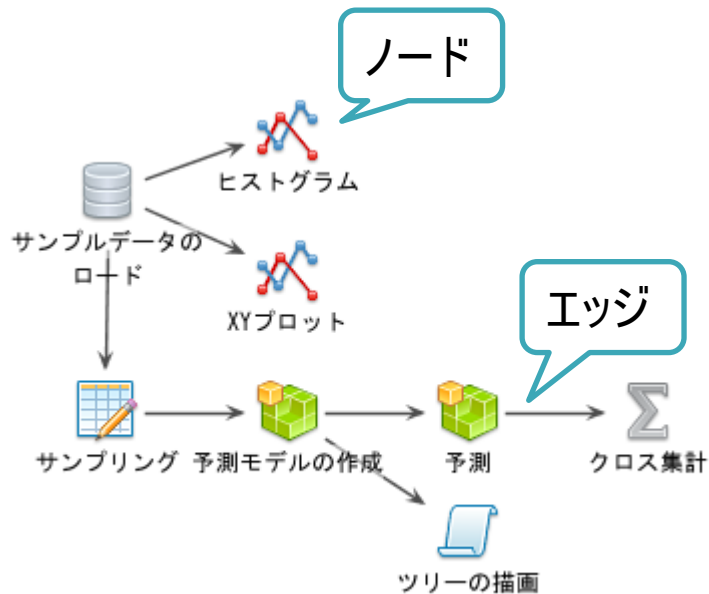
- 分析プロセスをワークフローで表現
- オープンソース、無償公開
- Javaで開発、マルチOS対応
  - ・ Windows / Mac / Linux



分析プロセスを  
視覚的に整理

# R AnalyticFlowとは

ノードをエッジで繋いだ分析フローを作成、Rスクリプトを生成して実行



# 1. データの読み込み

```
data(iris)
```

# 2. 探索的分析

```
plot(iris[, 1:4], col =
as.integer(iris$Species) + 1)
boxplot(Petal.Length ~ Species, data =
iris, col = 3, main = "Petal.Length")
```

# 3. モデリング

```
library(rpart)
rp <- rpart(Species ~ ., iris)
```

# 4. モデルの確認

```
plot(rp, margin = 0.1, branch = 0.3)
text(rp, fancy = T, all = T, use.n = T)
```

# 5. 予測および評価

```
pred <- predict(rp, type = "class")
xtabs(~pred + iris$Species)
```

# R AnalyticFlowとは

ファイルやRオブジェクトの一覧をインタラクティブに利用することができる。  
データファイルを選択して読み込み処理を作成したり、  
実行結果として得られたオブジェクトを確認することができる



The screenshot displays the R AnalyticFlow interface. On the left, a file explorer shows a project named '分析プロジェクト1 (作業)' containing files 'Boston.txt', 'diamonds.txt', and 'iris.txt'. The 'iris.txt' file is selected, and its contents are displayed in a table below. The table has columns for 'Sepal.Length', 'Sepal.Width', 'Petal.Length', 'Petal.Width', and 'Species', with rows numbered 1 to 10. The 'Species' column consistently shows 'setosa' for the first 10 rows.

On the right, a console window shows a list of R objects: 'iris : data.frame [150,5]', 'test : data.frame [50,5]', 'train : data.frame [100,5]', and 'model : rpart [14]'. Below this, a decision tree plot is shown with the following structure:

```
graph TD
    A["Petal.Length < 2.45"] --> B["setosa"]
    A --> C["Petal.Width < 1.75"]
    C --> D["versicolor"]
    C --> E["virginica"]
```

# 多言語・マルチOS

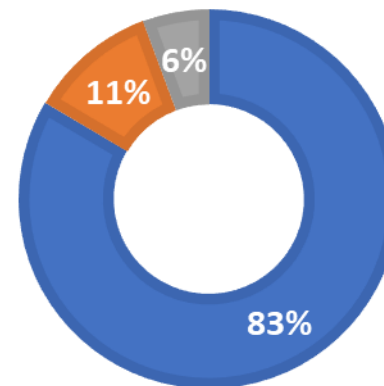
表示言語として日本語と英語を選択可能

世界141ヶ国でダウンロード(2021年12月16日時点、直近1年間)

Windows / Mac / Linux をサポート

1.  Japan
2.  United States
3.  China
4.  India
5.  Germany
6.  Taiwan
7.  Brazil
8.  United Kingdom
9.  Indonesia
10.  Canada

■ Windows ■ Mac ■ Linux



## これまでの歩み

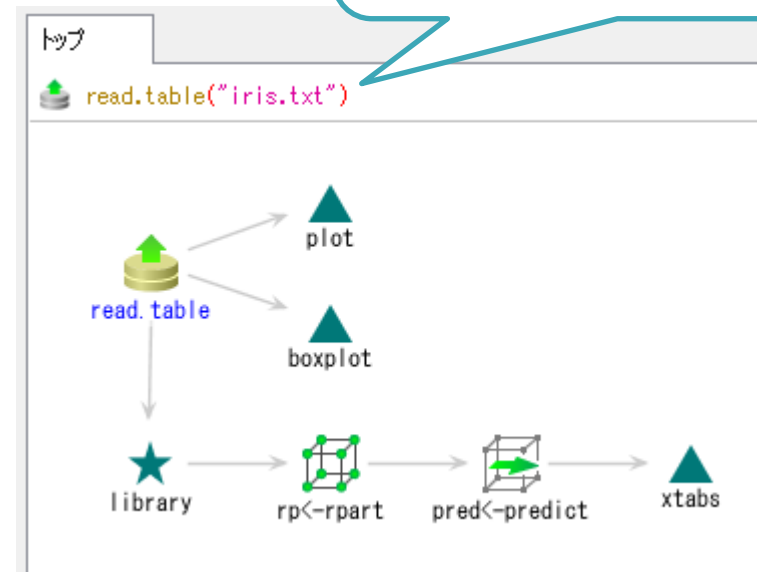
時期	バージョン	詳細
2007年12月	0.1.0	プレビュー版公開 (Windowsのみ)
2009年12月	1.0.0	現在に至る基本機能の多くを実装 Windows / Mac / Linuxで動作
2012年12月	2.0.0	タブインターフェースへの統合、デバッグ補助、 自動バックアップ、64bit対応
2015年12月	3.0.0	GUIによる分析機能、新インターフェース、 プロジェクト管理、カスタムUI作成機能
2017年6月	3.1.0	分析機能の強化 (予測分析、多変量解析、 仮説検定など)、利便性の向上、 安定性の向上 (Rの別プロセス化)
2021年8月	3.1.9	問題の修正、最新のRに対応

## 初期のバージョン(0.1~2.x)

ユーザーがRスクリプトを直接記述することで分析フローを作成。  
作成したフローは誰でも簡単に実行できる

```
トップ  
● dat <- read.table()  
file = header = sep = ...  
Ctrl+スペース または Tab で一覧を表示
```

ノードにRスクリプトを直接記述。  
関数を検出してタイトルにし、  
対応するアイコンを表示する

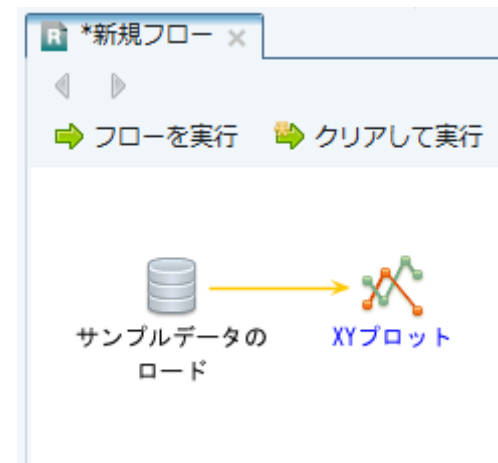
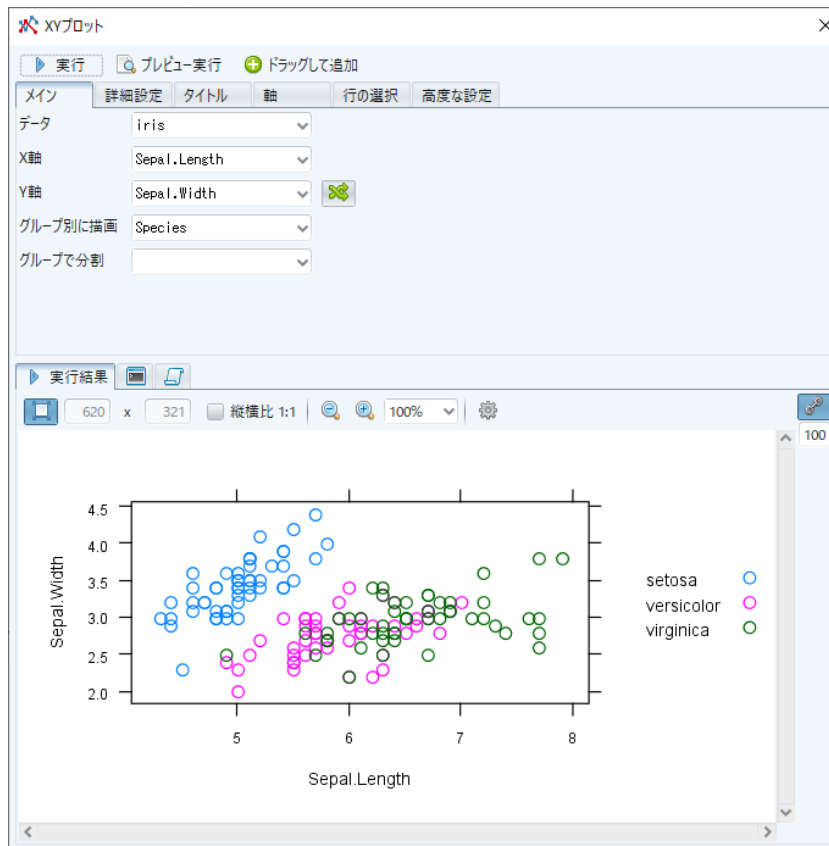




# 最近のバージョン(3.x)

簡単なマウス操作でRスクリプトを記述せずに分析が可能。  
Rの知識が必須ではなくなり、スクリプトの記述量も削減

ローコード、  
ノーコード



ドラッグ & ドロップ

# 機能例：予測分析

複数の予測手法やパラメータ設定のもとで予測モデルを作成し、予測精度を評価して出力

メイン モデル 詳細設定 モデルの評価

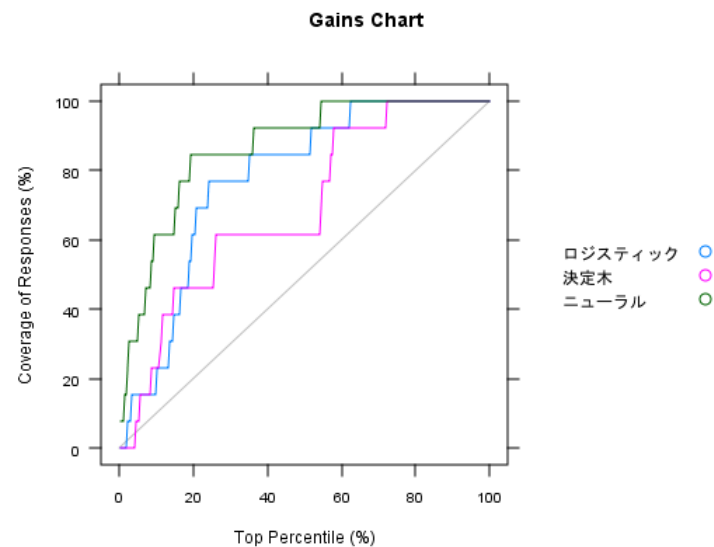
ニューラルネットワーク 高度な設定

出力名 ニューラル

隠れユニットの数 5

減衰パラメータ 0.1

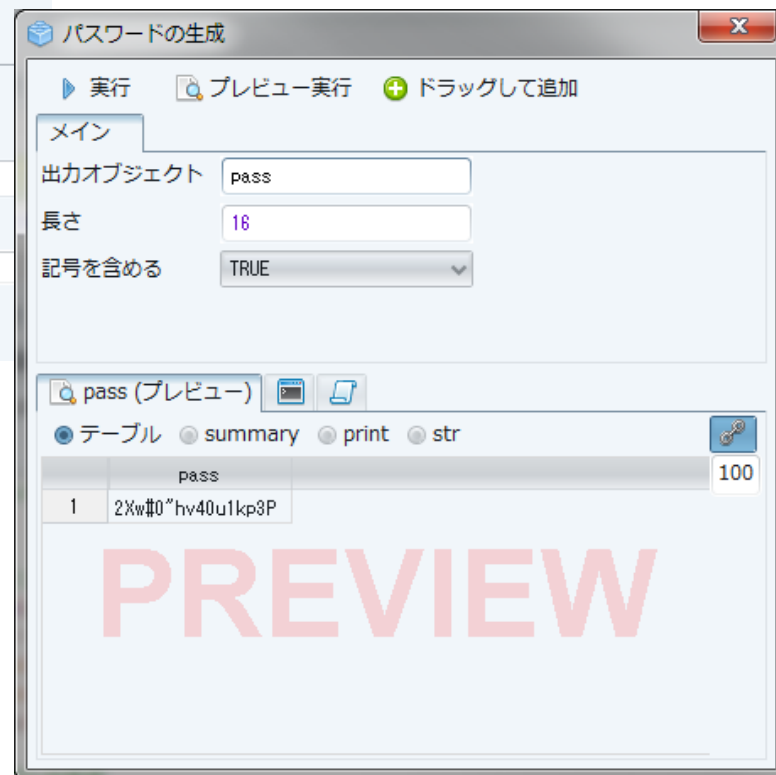
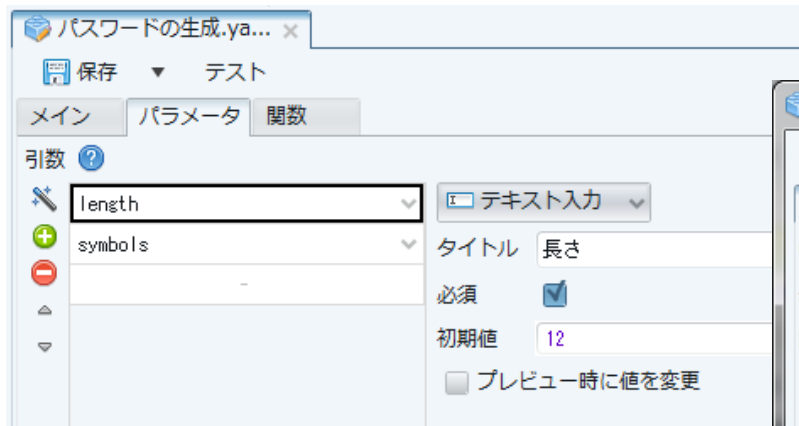
反復計算の回数 (最大値) 100



	Accuracy	Predict x Actual	
ニューラル	0.9409449	0	1
ロジスティック	0.9251969	0	1
決定木	0.9133858	0	1
		235	9
		6	4

## 機能例: カスタムUI

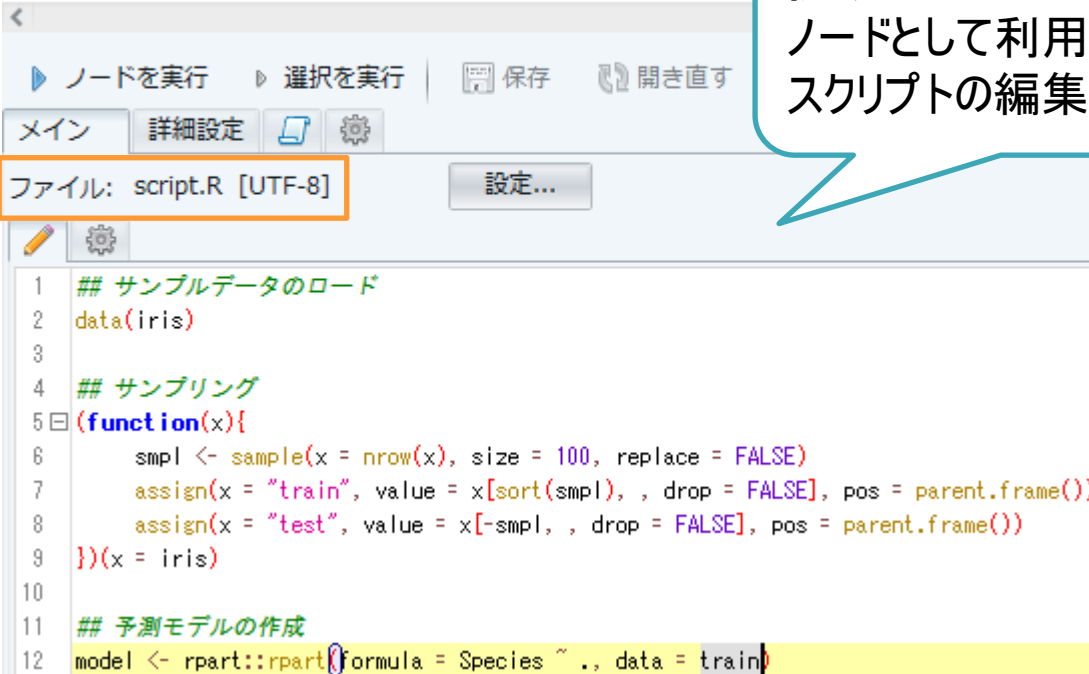
組み込みのGUIに存在しない機能を追加できる。  
Javaプログラミングを行わず、ビルダーを使って簡単に作成可能



ビルダー上で関数(自作も可能)を指定し、  
引数を設定用UIなどを指定して保存

## 機能例：外部ツール連携

IDEが使用するフォルダをプロジェクトフォルダとして設定し、  
Rスクリプトファイルをノードとして利用可能



script.R → ヒストグラム

ノードを実行 | 選択を実行 | 保存 | 開き直す

メイン | 詳細設定 | 設定...

ファイル: script.R [UTF-8] | 設定...

```
1 ## サンプルデータのロード
2 data(iris)
3
4 ## サンプリング
5 (function(x){
6     smpl <- sample(x = nrow(x), size = 100, replace = FALSE)
7     assign(x = "train", value = x[sort(smpl), , drop = FALSE], pos = parent.frame())
8     assign(x = "test", value = x[-smpl, , drop = FALSE], pos = parent.frame())
9 })(x = iris)
10
11 ## 予測モデルの作成
12 model <- rpart::rpart(formula = Species ~ ., data = train)
```

拡張子.Rのファイルを  
ノードとして利用できる。  
スクリプトの編集も可能。

# 設計方針と問題点

## ■ 方針：生成されるコードが「どこでも確実に動く」

- 外部パッケージに依存せず、標準パッケージ群 (base, recommended) のみで動くスクリプトを生成
- パッケージを使用するスクリプトをユーザーが書くことはできるが、GUIで生成するコードには含めない

## ■ 問題点

- モダンなRのデータ分析はパッケージに強く依存
  - ・ tidyverse (dplyr, ggplot2, tidyr, ...)
  - ・ 予測モデル (glmnet, ranger, xgboost, ...)
- かなりメジャーなパッケージでも、標準配布される様子もない

データ加工  
グラフィックス

# 新しい方針

基本方針は  
変えない！

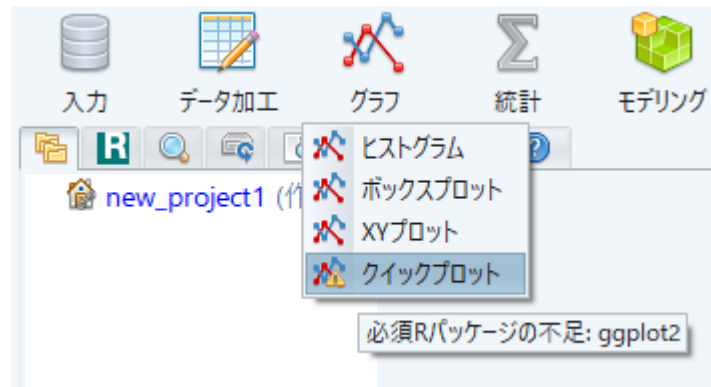
- 方針：生成されるコードが「どこでも確実に動く」
- 実現手段
  - 必要なパッケージを検出して通知
  - パッケージのインストールと管理をサポート
  - 必要であれば事前にパッケージをロード

2021年12月17日リリースの最新版(3.2.0)で実現

# 必要なパッケージを検出して通知

## ■ メニューでの表示

- 分析機能を呼び出す前にパッケージの有無をチェック



# 必要なパッケージを検出して通知

## ■ 分析フロー上での表示

- ノードごとに必要なパッケージを管理し、実行前に通知
- ユーザーの記述したスクリプトからもパッケージを検出



必須Rパッケージの不足: ggplot2

```
base::parse  
utils::getParseData
```

## ■ スクリプトのエクスポート

- 分析フローをRスクリプトに出力する際、パッケージをチェックするコードを含めるオプションを作成。
- 必須パッケージがない場合、実行後すぐにエラーで停止

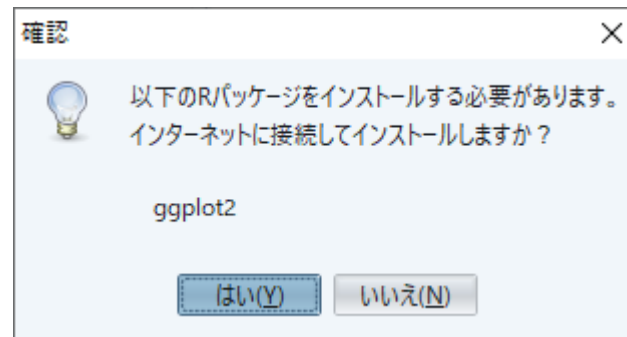
```
Error in loadNamespace(x) :  
  'ggplot2' という名前のパッケージはありません
```



# パッケージのインストールと管理をサポート

## ■ インストールの提案

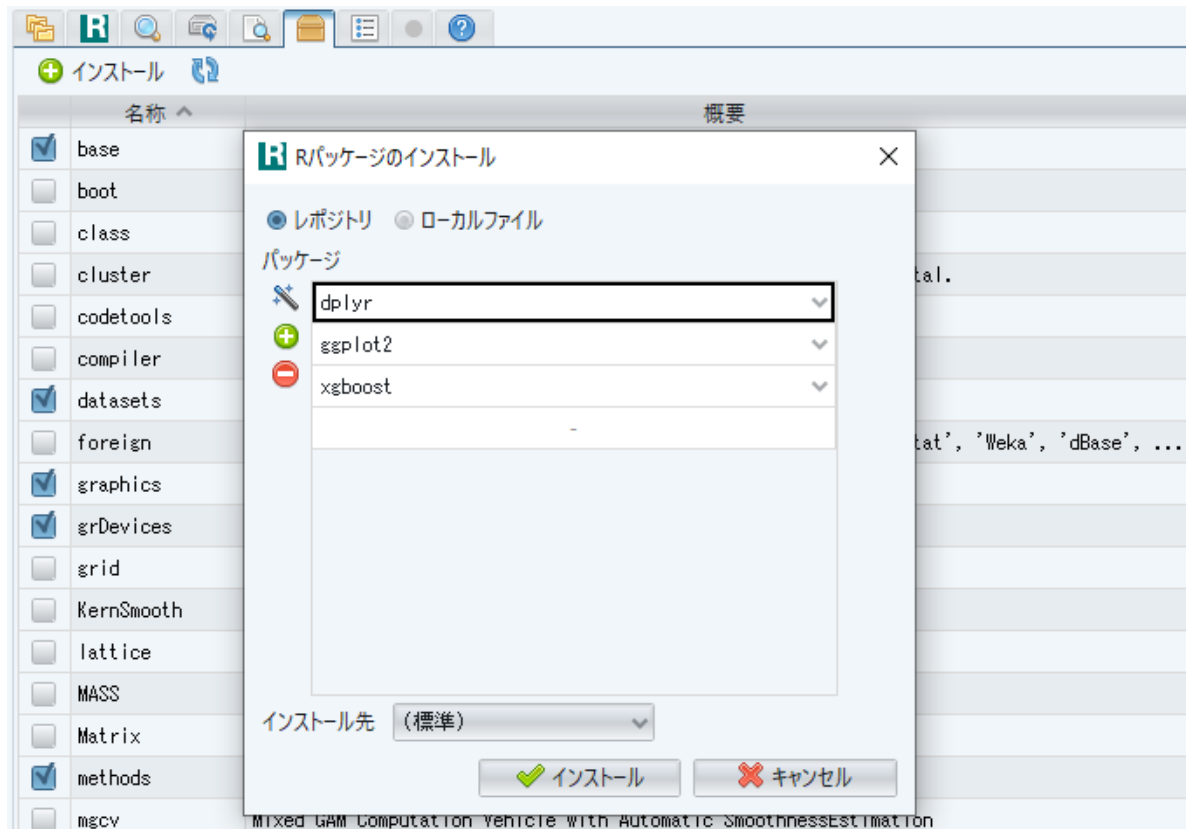
- 不足するパッケージがあればダイアログを表示し、ワンクリックでインストール
  - ・ 標準のCRANミラーを設定しており、初心者にとっては選択に困るミラーサイトの選択も省略



# パッケージのインストールと管理をサポート

## ■ パッケージマネージャ

- Rスクリプトを使わずにパッケージのインストールなどの管理を行うための仕組みを提供



# 必要であれば事前にパッケージをロード

## ■ パッケージの自動ロード

- 必要がある分析機能に限り、実行時に自動的にパッケージを読み込み・確認するコードを生成
  - ・ `library()` または `requireNamespace()`
- 関数をマスクするなどの意図しない副作用を起こす可能性があるため、できる限りアタッチせず最低限の利用とする方向で検討
  - ・ 基本的には `package::function()` の形

# 今後について

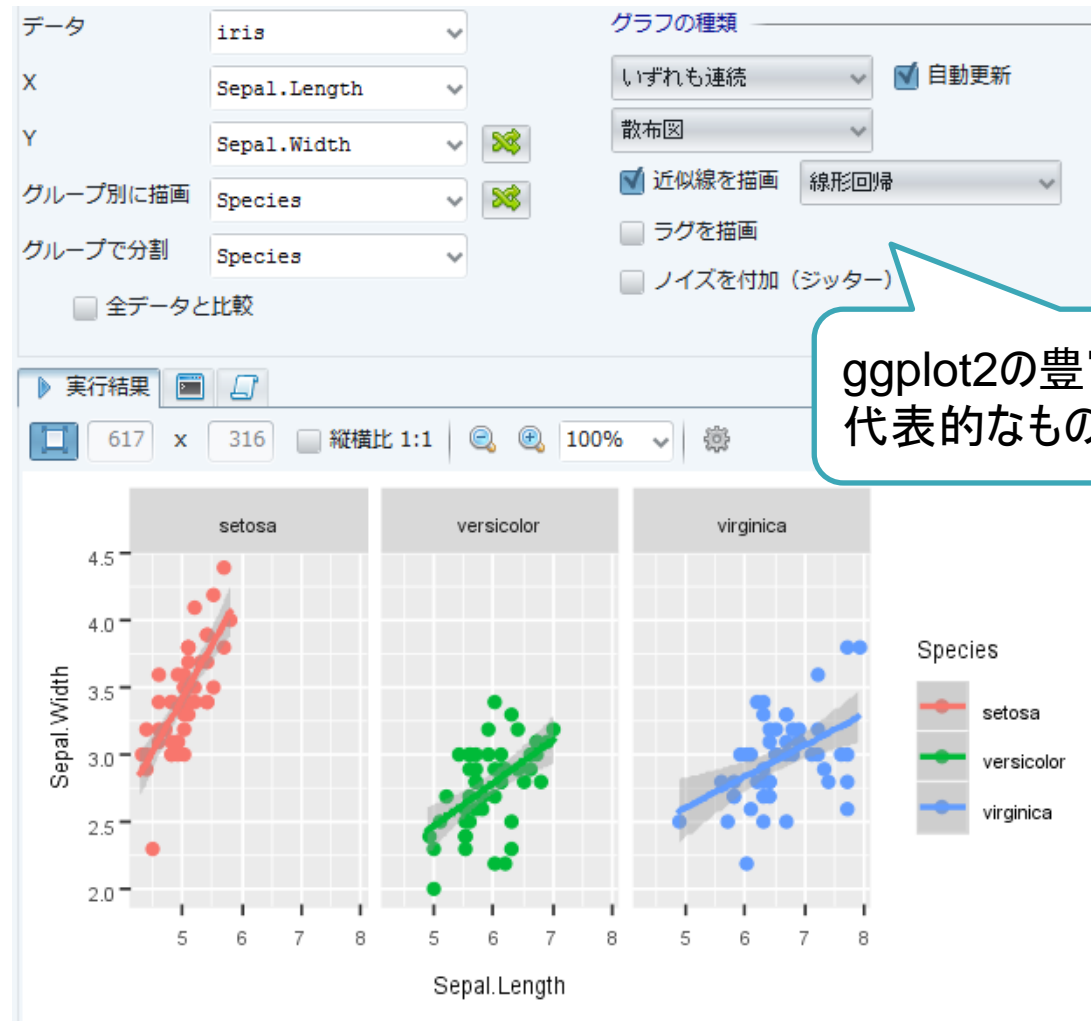
## ■ 進捗状況

- パッケージ管理および分析機能を実装し、公開済み
  - ・ プロット(ggplot2)
  - ・ モデリング(glmnet, ranger, xgboost)
  
- 今後は機能拡張を検討し、メンテナンスを継続予定



Java + R な開発者の方、ご興味ありましたらご一報を！

# デモ: ggplot2によるグラフィックス



ggplot2の豊富な機能から  
代表的なものをUIで提供

# デモ: 予測モデル (glmnetなど)

さまざまな手法を選択可

モデル  単一  リスト

入力データ

出力オブジェクト

モデル式

GUI  式オブジェクト  直接入力

目的変数 (Y)

説明変数 (X)  すべて  含める  除外する

モデル設定

手法

定数項を除く

出力タイプ

混合パラメータ

$\alpha$ の候補

クロスバリデーションの分割数

iris.glmnet (プレビ...

```

1 iris.glmnet <- (function(formula, data, subset = NULL, na.action = na.fail, alpha,
  family = NULL, .CV_ARGS = list(), ...) {
2   mf <- match.call(expand.dots = FALSE)
3   m <- match(c("formula", "data", "subset", "na.action", "family", ".CV_ARGS"), mf)
4   mf <- mf[c(1L, m)]
5   mf[[1L]] <- quote(stats::model.frame)
6   mf$drop.unused.levels <- TRUE
7   mf <- eval(mf, parent.frame())
8
9   mt <- attr(mf, "terms")
10  xlev <- stats::getXlevels(mt, mf)
11  mt <- update(mt, . ~ . - 1)

```

パッケージごとに必要となる  
前処理も含めてスクリプトを  
生成

ご清聴ありがとうございました



 **AnalyticFlow**

 <https://r.analyticflow.com>

 @ef-prime\_jp  R AnalyticFlow