Oracle R Enterpriseの実力とは？

アナリスト目線のファーストインプレッション

株式会社 ef-prime

鈴木 了太 （代表取締役 / データアナリスト）

ef-prime, inc.

会社概要

- 株式会社ef-prime
  - 2006年3月設立
  - データ分析コンサルティング
    - 統計解析、機械学習、最適化といった手法を用いて
      お客様のビジネス課題を解決
    - 課題に応じてデータ整備や分析手法のご提案、
      結果データの納品からソフトウェアによる自動化まで
  - 使用ツール
    - **R**（弊社開発のGUIと併用）
      - > その他SQLやJava、lp_solveなど各種OSS
      - > 要件に応じて商用の解析ツールやクラウドサービス、
        他言語（C#、Python、Ruby、ActionScript...）での開発も

事例：ダイレクトメールの発送最適化

背景

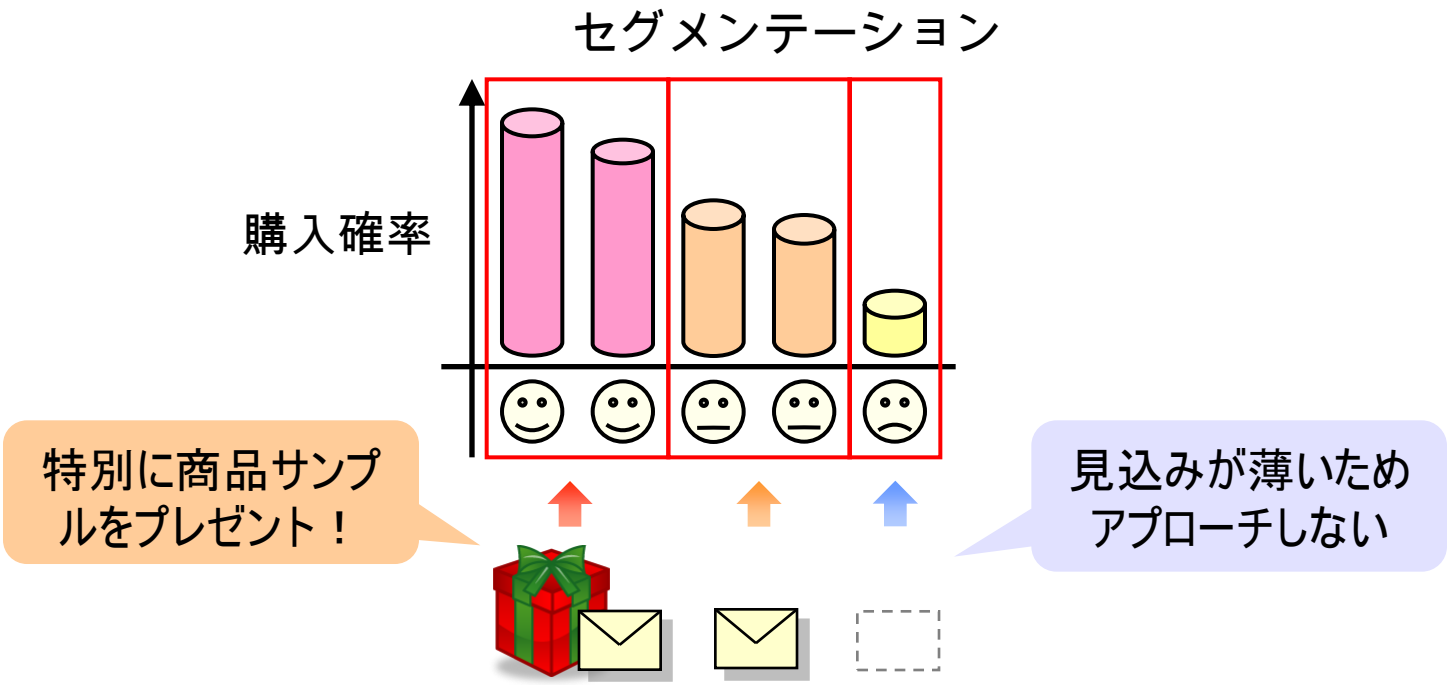
- 見込み顧客にダイレクトメールを送付
  - 膨大な人数の見込み顧客リストを保有
  - ハガキから商品サンプルまでさまざま
- 見込み顧客の各種データが蓄積されている
  - デモグラフィックス（例：性別、年齢、住所）
  - コンタクト履歴（例：過去の問い合わせ有無）
  - アンケートへの回答（例：購入意向）

課題

限られた予算でなるべく多くの顧客を獲得するには、
「誰に」「どの種類の」ダイレクトメールを送るのがよいだろうか？

事例：ダイレクトメールの発送最適化

- 解決策
  - 見込み顧客それぞれに将来の「購入確率」を予測
  - 購入確率に基づき、セグメント化して施策を実施



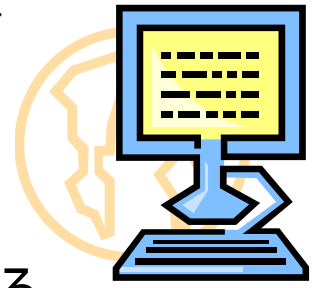
事例：ダイレクトメールの発送最適化

- 成果
  - 購入確率の低い見込み顧客へのダイレクトメールをカット、コストを削減しROIを向上
  - 購入確率の高い見込み顧客に集中的に資本を投下し、購買意向をさらに高めて購買へと繋げる

効率化と見込み顧客の育成を同時に達成！

事例：バナー広告の配信最適化

- 背景
  - 複数の媒体（ウェブサイト）でバナー広告を表示
    - 商品・キャンペーンによって様々な種類のバナーがあり、それぞれ成約時の利益額も異なる
    - どの媒体にどの広告を表示するかは任意に設定可能
    - 媒体によって課金される金額や基準が異なる
  - 成果データは毎日更新され、最新のものが利用できる
    - インプレッション数（表示回数）、広告クリック回数、成果ページのクリック回数などが取得可能

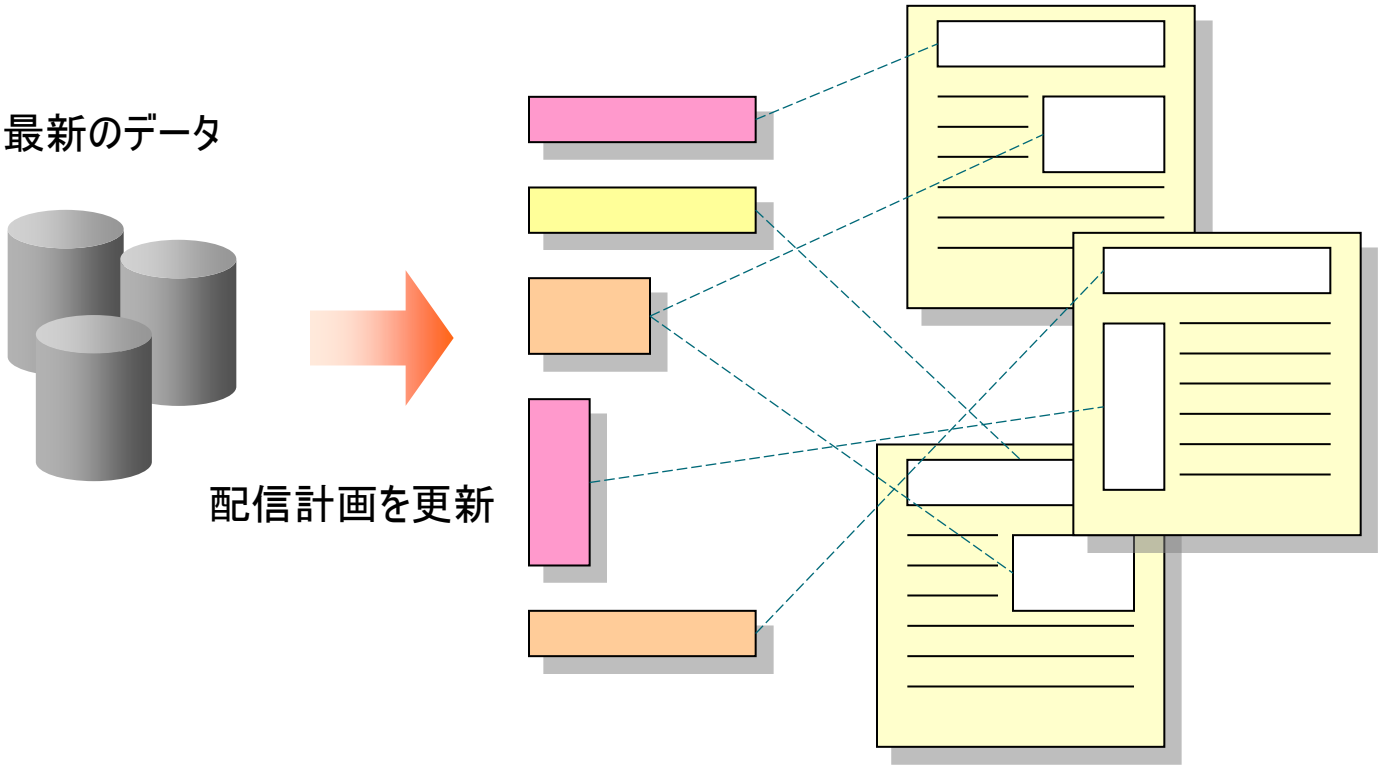


- 課題

利益を最大化するには、「いつ」「どの媒体に」「どのバナーを」「どれだけ表示」すればよいだろうか？

事例：バナー広告の配信最適化

- 解決策
  - 広告を表示したときの「利益額」を予測し、
    予測利益額をもとに最適な配信計画を策定
  - 最新のデータを反映させ、配信計画を自動的に更新



事例：バナー広告の配信最適化

- 成果
  - 最適な配信計画を策定し、利益を最大化
  - 自動化によって人的コストを低下させると同時に、傾向の変化にも即時に対応可能に

見込み顧客の傾向を媒体レベルで捉え、
最適なキャンペーンに誘導！

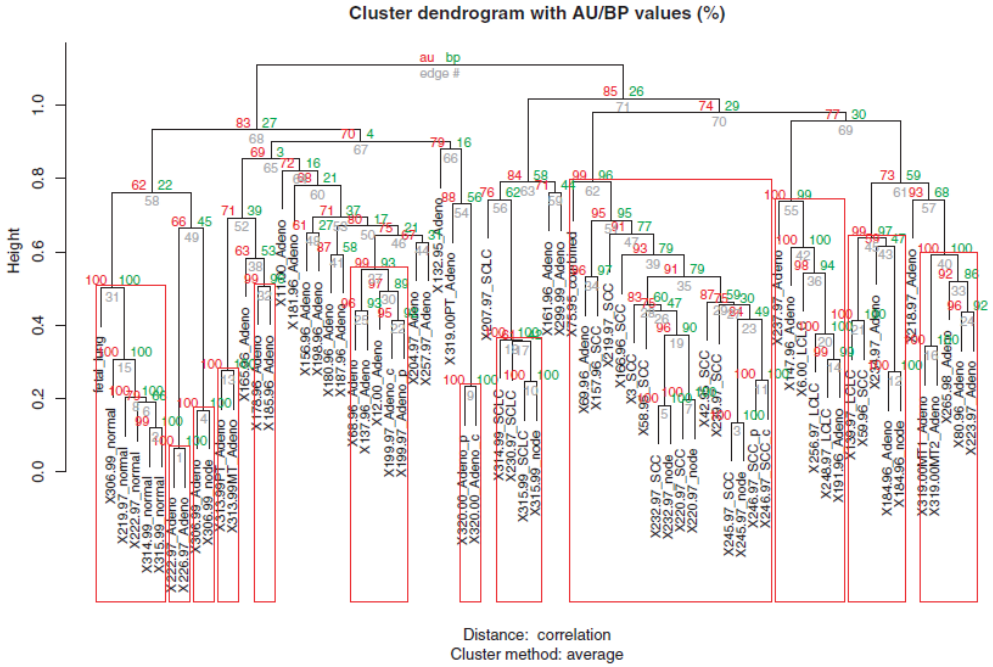
わたしとR

- 出会い
  - 2001年ごろ、S-PLUSと出会う
    - 大学のゼミ（データサイエンス）にて
    - 当時、Rはバージョン1.3前後
  - 2003年ごろ、Rに乗り換え
    - バージョン1.6前後。この頃Rの日本語化がスタート
      - > 「なかま」さん達による日本語化がRの国際化に貢献
      - > http://www.okadajp.org/RWiki/ ？日本語化掲示板

ef-prime, inc.

わたしとR

- pvclust パッケージのリリース
  - 階層型クラスタリングの信頼性評価
  - PCクラスタによる並列計算にも対応
  - バイオインフォマティクス系でよく使われるツールのひとつに



Rの基本

- 基本データ構造はベクトル

```
x <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
print(x)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

- こう書いても同じ

```
x <- 1:10
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

Rの基本

- データフレーム
  - 分析用データの基本データ構造

```
df <- data.frame(x = 1:10, y = 11:20)
print(df)
```

```
##     x  y
## 1   1 11
## 2   2 12
## 3   3 13
## 4   4 14
## 5   5 15
## 6   6 16
## 7   7 17
## 8   8 18
## 9   9 19
## 10 10 20
```

Rの基本

- オブジェクトのクラス

```
class(x)
## [1] "integer"
class(df)
## [1] "data.frame"
```

Rの基本

- クラスに応じた挙動：ジェネリック関数

```
class(x)
```

```
## [1] "integer"
```

```
head(x)
```

```
## [1] 1 2 3 4 5 6
```

```
class(df)
```

```
## [1] "data.frame"
```

```
head(df)
```

```
##   x  y
## 1 1 11
## 2 2 12
## 3 3 13
## 4 4 14
## 5 5 15
## 6 6 16
```

Rの基本

- クラスに応じた挙動：ジェネリック関数

```
summary(x)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    3.25    5.50    5.50    7.75   10.00
```

```
summary(df)
```

```
##        x                y
##  Min.   : 1.00   Min.   :11.00
##  1st Qu.: 3.25   1st Qu.:13.25
##  Median : 5.50   Median :15.50
##  Mean   : 5.50   Mean   :15.50
##  3rd Qu.: 7.75   3rd Qu.:17.75
##  Max.   :10.00   Max.   :20.00
```

Rの基本

- サンプルデータ
  - irisデータ
  - data.frameオブジェクト

```
data(iris)
class(iris)
```

```
## [1] "data.frame"
```

ef-prime, inc.

Oracle R Enterprise

- OREフレーム
  - data.frame風のオブジェクト、実体はデータベースのテーブル

```
IRIS <- ore.push (iris)
class(IRIS)
## [1] "ore.frame"
## attr(,"package")
## [1] "OREbase"
```

ef-prime, inc.

Oracle R Enterprise

- Oracleテーブルの取得
  - ore.frameをdata.frameとして取得
  - Oracleテーブルを簡単にRオブジェクトに変換できる
    - メモリにさえ乗れば、通常Rで行う処理がそのまま可能
    - 実はこれだけでも十分便利！

```
iris.df <- ore.pull (IRIS)
class(iris.df)
```

```
## [1] "data.frame"
```

Oracle R Enterprise

- 透過型実行
  - ore.frameをdata.frameであるかのように操作

```
head(iris, 3)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
```

```
head(IRIS, 3)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
```

その実力は？

- 疑問がいろいろ
  - 日本語は使えるのか？
  - ドキュメントは？
  - データの入出力は？
  - 分析前のデータ加工はDBで？Rでやる？
  - 統計・機械学習モデルは何が使える？
  - モデルを使った予測はどうやって行う？自動化は？

短期間での評価ですが、ひと通り試してみての
ファーストインプレッションをお話させていただきます。

一般

- 日本語
  - データ中の値、カラム名ともに利用可能
    - 文字化けする場合、環境変数LANG、NLS_LANGなどの設定で解消

- ドキュメント
  - 充実している
    - ただし学術用語の訳語には不自然な点もあり
      - > 例：Cross-Validating Modelsの訳語が「交差検定モデル」となっているが、正しくは「モデルの交差検証」または「モデルの交差確認」など。「検定」は統計用語としてはTestに対応する
      - > 場合によっては英語版のヘルプと併用するのが望ましい

データの入出力

- 非常に簡単
  - RからOracleへはore.push、逆はore.pullで変換可能
  - テーブルの作成、削除はore.create、ore.dropで
  - データ型の変換も自動でやってくれる
    - Rでテキストファイルからデータを読み込み、これをore.createすれば簡易テキストファイル入力ツールとしても使える？

データ加工

- いつものR関数がそのまま使える
  - ore.frameでもdata.frameと同じように記述できる
  - 例
    - 部分データの選択：subset
    - カラムの追加：transform
    - 集計：aggregate

データ加工

- ときにはデメリットにも
  - 対応するR関数の仕様がそもそも使いにくい場合、その弱点も踏襲する
  - SQLを記述して処理させたほうが良い場面もありそう
    - これが可能というだけでもメリット
    - RではsqldfパッケージをRを用いて実施していた部分
      - > data.frameをSQLiteに一時的に書き出し、SQL文を渡して処理させ、結果をdata.frameで受け取る
  - データのサンプリングはORE側でできると嬉しい部分
    - R上で乱数を発生させて行う方式が一般的、OREドキュメントにも記載
    - 慣れないと使いにくく、可読性も低くなりがち

モデリング

- OREオリジナル実装の関数が便利
  - R標準のlm、glmなどに対し、ore.lm、ore.glmなど
  - 大量データを想定した設計になっており、
    予測の際にも実用的にすぐれた実装

- 残念な部分も
  - ステップワイズ回帰（ore.stepwise）が名義変数
    （ore.factor）に対応していない
    - 多くのカラムから必要なものだけを抽出する機能。
      名義変数とは「都道府県」のような離散値をとる変数
    - R標準のstep関数より高速で動作する（と思われる）
      実装になっており、これを使えないのはもったいない

モデルの評価

- 一通りのことは可能だが、コーディングが必要
  - データを「学習用」「検証用」に分けるとき、乱数を発生させてデータを分割する処理を記述する必要がある
  - クロスバリデーション法が標準では使えない
    - ブログ記事で公開されている実装があるが、実運用で利用できるほどこなれていない印象
      - 例：評価の際にグラフを必ず出力してしまう、評価結果がRオブジェクトやOracleテーブルではなくファイルに出力される

予測

- OREオリジナル実装が使いやすい
  - ore.lmなどによる予測値は内部でSQL文として予測モデルの式を保持している（ように見える）
  - 非常に合理的な実装で、うまく使えばデータの更新に対応しやすくなるのではと期待

- いろいろ期待が持てそうな仕様（未評価）
  - モデルをデータベースに保存しておけば自動化もしやすい（らしい）
  - 行分割実行をうまく使えば、一般のRモデルについて大規模データに対する予測処理がしやすいのでは？

総括：誰にオススメ？

★★★★★

- 既にOracleにデータがあり、ある程度Rが使える方
  - OracleとRの間でシームレスにデータのやり取りができる
  - 専用の関数は運用まで含めてケアされており、
    通常のR関数にかなり近い使用感で利用できる
  - 大規模データに対して特にメリットがあるはず（未評価）

★★★★

- 既存のRユーザーで、運用や大規模データの
  処理などを改善したい方
  - すべての専用関数がいつもと同じように動くわけではないが、
    多くの処理は予想通りに動いてくれる

総括：誰にオススメ？

★★★

- Oracleにデータがあり、統計や機械学習を導入したいが
  Rの利用経験がない
  - Rそのものの学習コストが発生してしまう
    > いちど学習すればOracleに限らず応用できるので
    学習するだけの価値はある
  - ドキュメントが充実しており、日本語でもひととおりの学習が
    できるため「データ分析ことはじめ」としてやってみるのはオススメ
    > 初学者のつまづきがちなポイントであるデータ入出力が
    簡単に利用できる実装になっている
    > 最初からOracleありきで学習しておけば、既存のデータに
    適用してみることは比較的容易

まずは**Oracle Cloud**で気軽に試してみましょう！

ef-prime, inc.

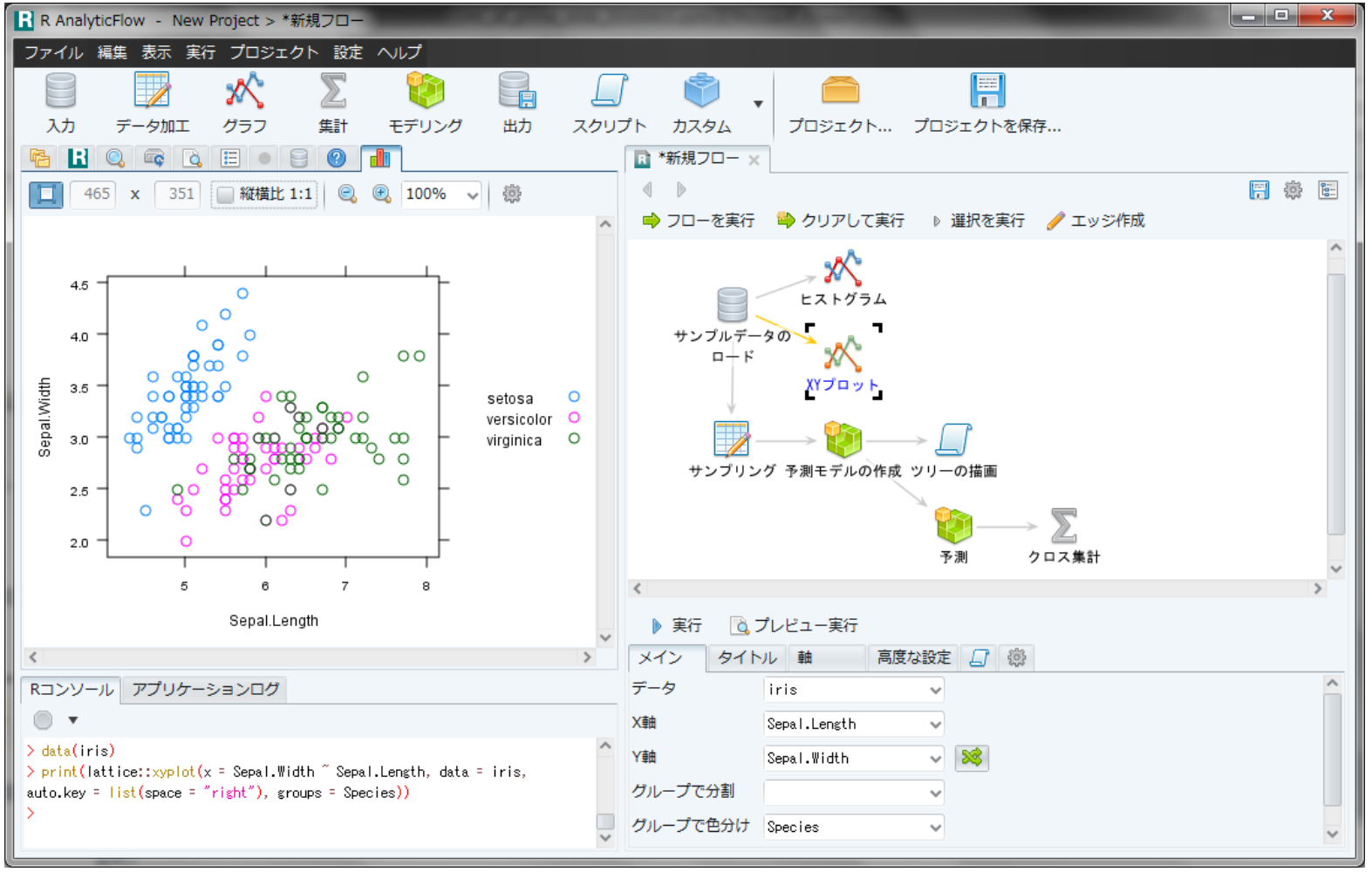
R AnalyticFlowのご紹介

R AnalyticFlowとは

- データ分析のためのR GUI
  - オープンソース
  - Javaで開発
  - Windows / Mac OS X / Linux

ef-prime, inc.

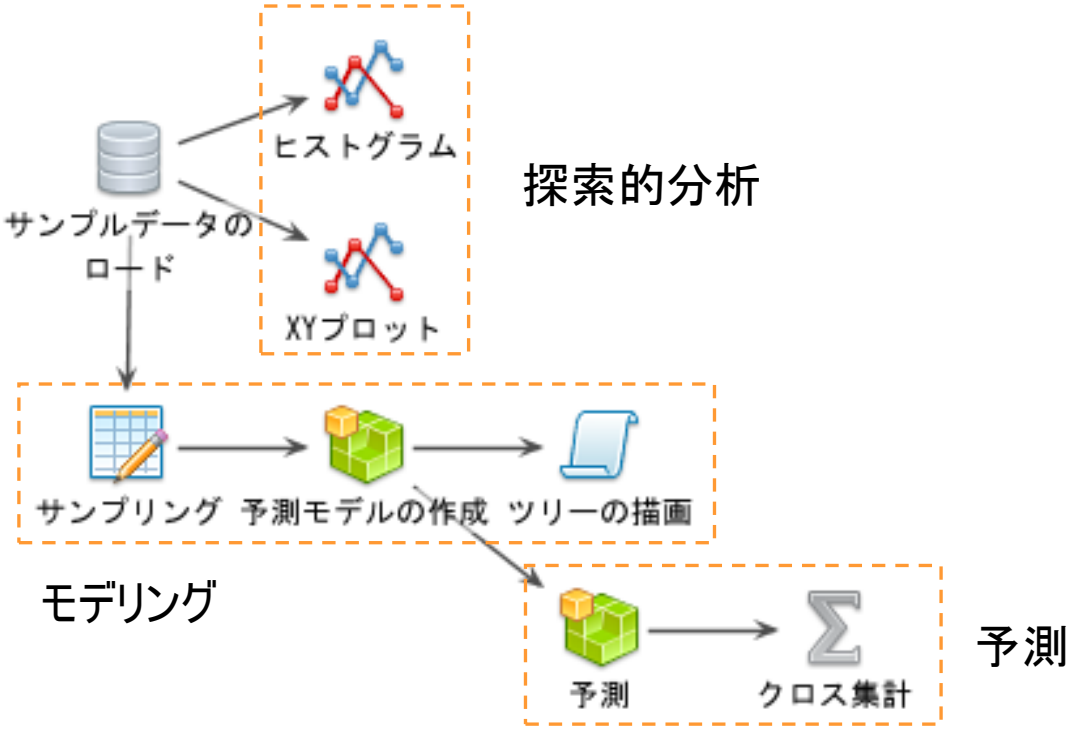
スクリーンショット


R AnalyticFlow - New Project > *新規フロー
ファイル 編集 表示 実行 プロジェクト 設定 ヘルプ
入力
データ加工
グラフ
集計
モデリング
出力
スクリプト
カスタム
プロジェクト...
プロジェクトを保存...
*新規フロー
フローを実行
クリアして実行
選択を実行
エッジ作成
ヒストグラム
サンプルデータのロード
XYプロット
サンプリング
予測モデルの作成
ツリーの描画
予測
クロス集計
Sepal.Width
Sepal.Length
setosa
versicolor
virginica
実行
プレビュー実行
メイン
タイトル
軸
高度な設定
データ
iris
X軸
Sepal.Length
Y軸
Sepal.Width
グループで分割
グループで色分け
Species
Rコンソール
アプリケーションログ
> data(iris)
> print(lattice::xyplot(x = Sepal.Width ~ Sepal.Length, data = iris,
auto.key = list(space = "right"), groups = Species))
>

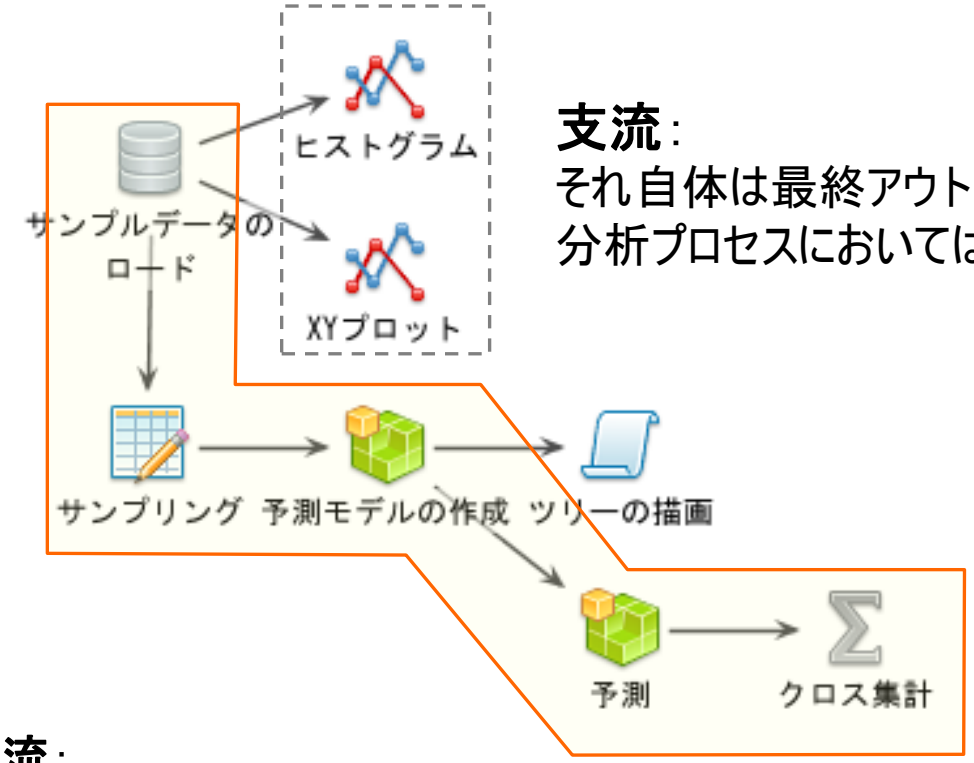
分析スクリプト

```r
# 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)
```

分析フローによる表現


ヒストグラム
サンプルデータの
ロード
XYプロット
探索的分析
サンプリング
予測モデルの作成
ツリーの描画
モデリング
予測
クロス集計
予測

本流と支流

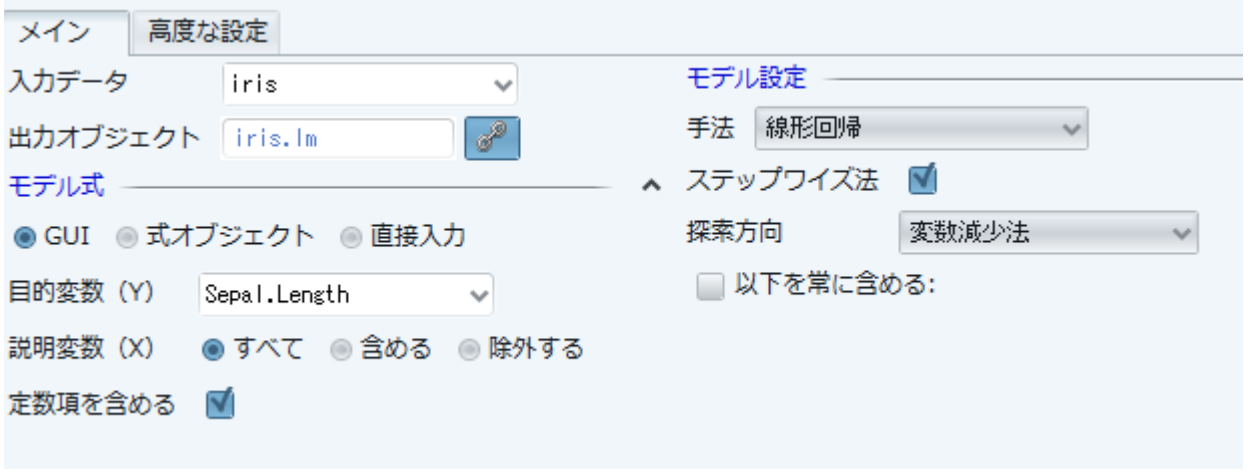


支流:
それ自体は最終アウトプットではないが、
分析プロセスにおいては重要

本流:
分析の最終アウトプットとなる部分

GUIからRスクリプトを生成

- 例：予測モデルの作成
  - マウス操作で設定を入力



Rスクリプトを自動生成

```
iris.lm <- step(lm(formula = Sepal.Length ~ ., data = iris),
        direction = "backward")
```

R AnalyticFlowの特徴

- 特徴
  - GUI操作でRスクリプトを自動生成
  - 作成したフローはクリック操作で簡単に実行
  - デバッグ機能などのコーディングサポート機能も

- 動作環境
  - マルチOS対応
    - Windows / Mac OS X / Linux で動作
    - Javaで開発、JRI (Java R Interface) でRと接続
  - 多言語対応
    - 日本語および英語

ef-prime, inc.

R AnalyticFlowの入手と設定

- ソフトウェアの入手
  - OSSとして無償公開
  - http://www.ef-prime.com
    - または **R AnalyticFlow** で検索

- インストールと起動
  - Windows / Mac OS X
    - Rが利用可能なPCにインストールして起動
  - Linux
    - R、Oracle Java 8 JDK、rJavaパッケージをインストール
    - ダウンロードしたアプリケーションを起動

http://www.ef-prime.com/ or 🔍 **R AnalyticFlow**

サポート

- ご利用サポート
  - ドキュメント、チュートリアルが付属
  - 問題の報告、ご質問などはこちら
    - rflow-support @ ef-prime.com
      - > 繁忙期など難しい場合もありますが、なるべくご回答差し上げます

- その他ご相談
  - 法人向けサポートなど
    - contact @ ef-prime.com
    - 研修、分析コンサルティングのご相談も承ります

http://www.ef-prime.com/ or 🔍 R AnalyticFlow

ef-prime, inc.

http://www.ef-prime.com

@ef-prime_jp

R AnalyticFlow