データサイエンスにおける Ruby の現在の位置付けと将来性

村田賢太[*]

2016 年 12 月 5 日

1 はじめに

現在 Ruby[*1]は、データサイエンスの業務ではほとんど役に立たないプログラミング言語であると言っても過言ではない。Ruby はデータサイエンスの世界では存在していないも同然の扱いであり、選択肢にすら上がらない。

その理由は、実用的に使える環境が存在しないことである。データサイエンスで使える Ruby 向けのツールは少ないが存在はする。しかし、これらは開発者各自が思い思いのツールを好き勝手に作る方式で供給されており、計画性がないオープンソース活動に支えられている。そのため、ツール間でデータのやり取りができない、データサイエンスのすべての工程を一貫して Ruby で実施できない、ツールの品質が悪くて利用できない、などの理由から業務での利用を諦めざるを得ないのが現実である。

この状況を一変させるには、データサイエンスの仕事を最初から最後まで Ruby で実施できる実用的な環境を作り上げることが必要である。その方法として、本レポートでは、データサイエンスの仕事でよく利用されている Python[*2]と R[*3]のツールを Ruby から利用する方法の開発を提案する。開発の進め方としていくつかのマイルストーンを示し、この計画の現状に対する利点と欠点をそれぞれ述べる。

2 データサイエンス領域におけるプログラミング言語

本節では、データサイエンス領域において重要なカギを握るプログラミング言語として、現在主流の Python と R、そして、将来性が期待されている Julia[*4]について述べる。

2.1 データサイエンスにおける Python と R の位置付け

データサイエンスの領域では、プログラミング言語として Python と R が現在の主流として多くのユーザを集めている。

Python は機械学習に強いプログラミング言語である。その理由は、scikit-learn[*5]がとてもよく作られていて使いやすいからである。scikit-learn のユーザが多いため、xgboost[*6]や keras[*7]などの後発ライブラリが scikit-learn と互換性のあるインターフェイスを提供することで、大きなライブラリ生態系が作られている。

R は、もともと統計解析のために作られたプログラミング言語であり、統計解析のための機能は非常に豊富である。時系列解析のように、R を使わなければ実現できない手法もある。

両言語がよく利用されるもう一つの理由として、データフレームを便利に扱えることが挙げられる。Python には pandas[*8]がデータフレームを提供し、R は言語のコア機能としてサポートしている。

Python と R はスクリプト言語であり、科学技術計算で要求される高速な実行速度は望めない。そのため、両言語とも、データサイエンスで利用さ

[*] muraken@gmail.com
[*1] https://www.ruby-lang.org/
[*2] https://www.python.org/
[*3] https://www.r-project.org/
[*4] http://julialang.org/
[*5] http://scikit-learn.org/stable/
[*6] https://github.com/dmlc/xgboost/
[*7] https://keras.io/
[*8] http://pandas.pydata.org/

れるライブラリ群のコア機能は C, C++, Fortran で実装されている。Python は、SciPy コミュニティ[*9]が開発している Cython[*10]と呼ばれるグルー言語により、C 拡張を実装しやすくなっている。R は Rcpp[*11]ライブラリによって C++ との連携が強化されている。

2.2 データサイエンスにおける Julia の位置付け

Julia は、2012 年に登場したばかりの若いプログラミング言語であり、活発に開発が進んでいる。この言語は、Python のように書きやすく、C++ や Fortran のように高速に実行できる[*12]ことが特徴である。そのため、アルゴリズムの実装とそのアプリケーションの両方を Julia で書ける。これはプログラマにとって非常に良い特徴であり、データサイエンス領域で将来主流となることが期待されている。

現在の Julia は、言語仕様そのものが過渡期特有の不安定性を持っており、バージョンが上がる度に下位互換性の無い変更が入るため、実用的では無い。

一方、Julia は、Python や R の資産と連携するためのライブラリが準備されており、現在の環境から移行するユーザの利便性が考えられている。例えば、Python のユーザは、ScikitLearn.jl[*13]と PyPlot.jl[*14]を用いて、それぞれ scikit-learn と matplotlib[*15]を Julia から利用できる。これらは PyCall.jl[*16]を利用して作られたラッパーであり、Python で実装された他の既存資産も同様に呼び出せる。

3 SciRuby の現状と将来性

SciRuby[*17]は、Ruby のための科学技術計算とデータ可視化のためのライブラリを開発している開発者コミュニティである。SciRuby は 2011 年に John Woods 氏により開始され[*18]、毎年 Google Summer of Code[*19]でプロジェクトを運営するなど、現在も活動が続いている。

しかしながら、データサイエンス領域では、Python, R, Julia とは比べるまでもなく、SciRuby の存在感は無いに等しい。この状況には次の 3 つ要因が相互に関連し、ネガティブフィードバックを作り上げていることが関係していると筆者は考えている。その要因とは、(1) 必要な道具が揃っていない、(2) ユーザが少ない、(3) 開発者が少ない、である。

必要な道具が揃わないのは、SciRuby がボランティアベースの開発者コミュニティーだからである。開発者が各々で好き勝手に欲しい道具を作るため、異なる道具間の連携が整備されず、一つの環境としての完成度は低い。そのためユーザと開発者が増えず、オリジナルの開発者が開発を止めるとそのまま放置されてしまい、完成度がどんどん低下していく。このような負のサイクルを止めなければ、Ruby がデータサイエンスの領域で利用されるプログラミング言語にはなれない。その理由は、比較対象である Python と R の完成度が非常に高く、さらに Julia との比較でも SciRuby の完成度は低すぎると言って良いレベルだからだ。

将来に向けて負のサイクルを止めるには、実用的な環境を短い期間で整備し、Ruby を使ってデータサイエンスに取り組むユーザを増やす必要がある。

4 将来に向けた開発計画

前節で述べたように、Ruby が将来、データサイエンスで使えるプログラミング言語になるためには、Ruby 向けの実用的な環境を短期間で実現する必要がある。そのための現実的な手段として、既存の良い環境、即ち Python と R のために開発された環境を Ruby から使えるようにする方法を提案する。

Ruby のために独自環境をゼロから開発する方法は現実的ではない。これは、短期間で実用的なものにならないだけでなく、多くのユーザを確保する面でも困難である。なぜなら、成功するかどうか分からない独自環境を採用することは、エンジニアにも企業にもメリットがないからである。すでにメインストーリムとして確立されている Python や R に適合させる方が、エンジニアとしては仕事を得る機

[*9] https://www.scipy.org/
[*10] http://cython.org/
[*11] http://www.rcpp.org/
[*12] http://julialang.org/benchmarks/
[*13] https://github.com/cstjean/ScikitLearn.jl/
[*14] https://github.com/JuliaPy/PyPlot.jl/
[*15] http://matplotlib.org/
[*16] https://github.com/JuliaPy/PyCall.jl/
[*17] http://sciruby.com/
[*18] http://sciruby.com/blog/2011/08/15/first-post/
[*19] https://developers.google.com/open-source/gsoc/

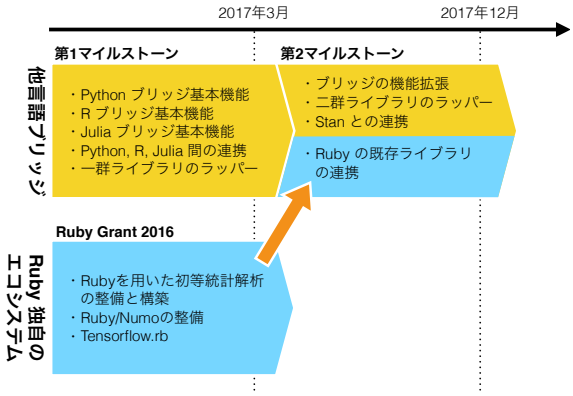

図1　他言語ブリッジの開発計画 (上段) と Ruby Grant 2016 (下段) との関係

会が多くなり、企業としては世界中に多数存在する優秀なエンジニアを獲得できる可能性が高くなる。

4.1　開発計画のマイルストーン

提案する開発計画は、大きく 2 つのマイルストーンから構成される。

第 1 マイルストーンは、データサイエンスのために最低限必要な機能のラッパー、そのラッパーを実現するために必要となる言語間ブリッジの基盤整備、他言語間連携への対応で構成される。第 2 マイルストーンは、ブリッジ機能の強化、追加機能のラッパー、ベイズ推定のための Stan との連携、そして Ruby の既存ライブラリとの連携の開発で構成される。2 つのマイルストーンを図1の上段に図示した。

データサイエンスで利用される他言語のライブラリを一群と二群に分け、一群の基本機能を第 1 マイルストーンで、一群の機能強化および二群の基本機能への対応を第 2 マイルストーンで扱うこととする。一群ライブラリと二群ライブラリの構成は以下の通りである。

5　Ruby Grant 2016 と本レポートの開発計画との関係

Ruby Grant 2016[*20]では SciRuby に関連する開発プロジェクトが 4 件採択されている。4 件とも本レポートで提案する開発計画とは異なり、Ruby のための独自環境の整備に該当するプロジェクトである。

表1　一群ライブラリと二群ライブラリの一覧

【一群ライブラリ】

ライブラリ名	プログラミング言語	分類
numpy	Python	計算基盤
scipy	Python	計算基盤
pandas	Python	データフレーム
scikit-learn	Python	機械学習
matplotlib	Python	可視化
base	R	計算基盤
stats	R	統計解析
graphics	R	可視化

【二群ライブラリ】

ライブラリ名	プログラミング言語	分類
seaborn	Python	可視化
gensim	Python	機械学習
pillow	Python	画像
keras	Python	深層学習
chainer	Python	深層学習
dplyr	R	データフレーム
tidyr	R	データフレーム
vars	R	統計解析
forecast	R	統計解析
tseries	R	統計解析
ggplot2	R	可視化

前節で提案した開発計画は、第 2 マイルストーンで取り組む「Ruby の既存ライブラリとの連携の開発」において、Ruby Grant 2016 のプロジェクトとの連携機能が導入されることになる (図1における下段から上段への矢印)。

6　まとめ

本レポートでは、Ruby をデータサイエンスで使えるプログラミング言語にする計画について述べた。それは、Python と R の資産を Ruby から利用するためのブリッジとラッパーの開発である。

さらに、将来の技術動向の変化に対応していくために、プログラミング言語 Julia とのブリッジの必要性を説明した。

提案した方式には、データサイエンスの世界でメインストリームとなっているツールを Ruby でも利用でき、業界標準のスキルセットを Ruby でも流用できる、などの利点がある。そして、短い期間で実用的な環境を作り上げることが可能である。

[*20] http://www.ruby.or.jp/en/news/20161121.html