

A Larger Tent: Integrating Multiple Open-source Environments in SAS

Software Demonstration: SD-02

Sundaresh Sankaran
Samiul Haque



Open Source Today

Understanding the open source ecosystem

Open source is prevalent throughout the technological landscape.
Some examples of analytics areas where we see open source projects:

USER INTERFACE

Where you write your code



LANGUAGE

What syntax you use



FUNCTIONS

Packages or functions used



EXECUTION

Where the code executes



ORCHESTRATION

How is it deployed



Note: This is only the tip of the iceberg of open source projects!

Multiple Analytical Technologies => One ecosystem

- Get the job done as each tool has its own benefits.
- Interact with one another for optimal value.



Open-source users want more.

“I want to quickly install a new Python /R package for testing purposes.”

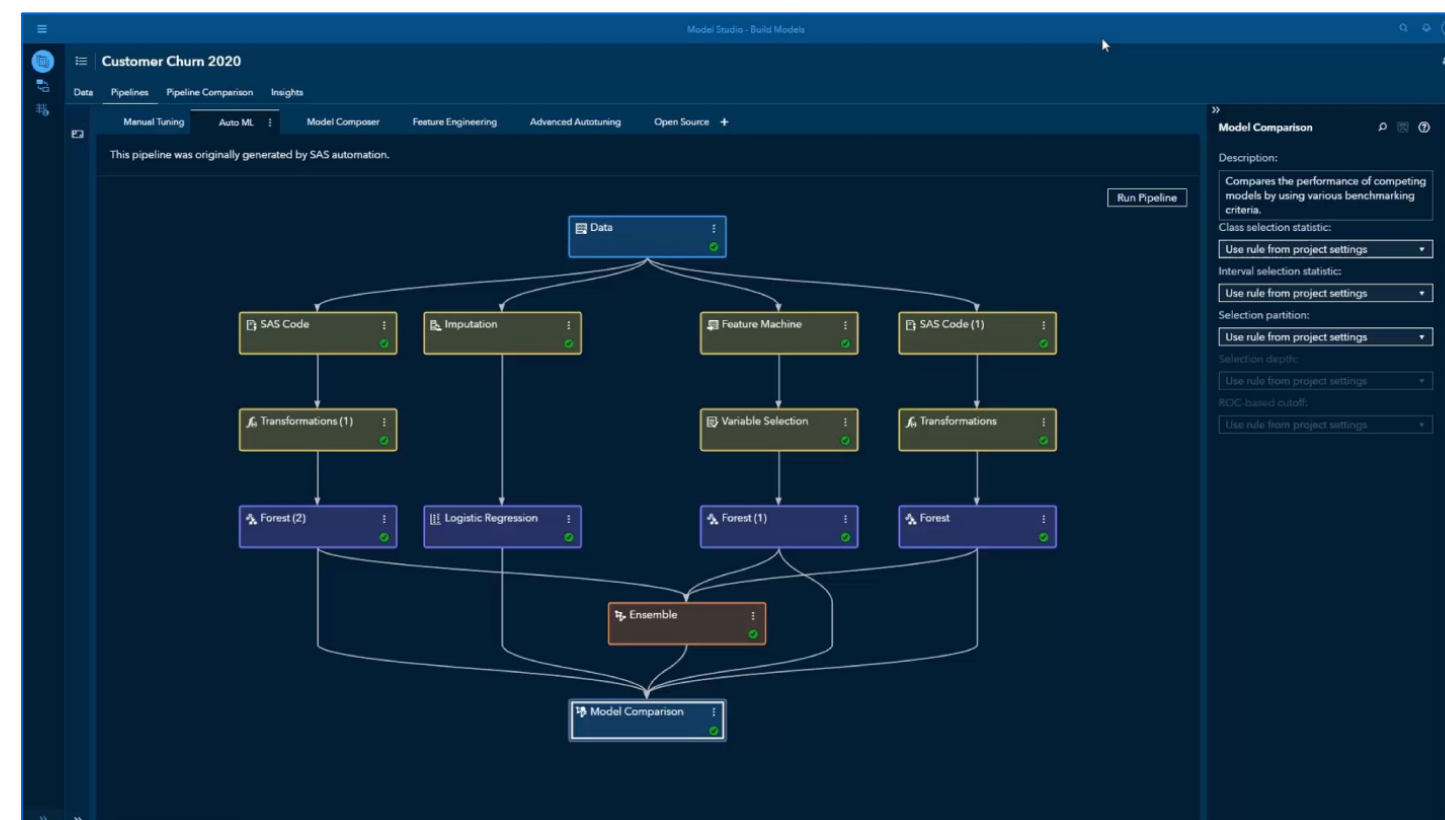
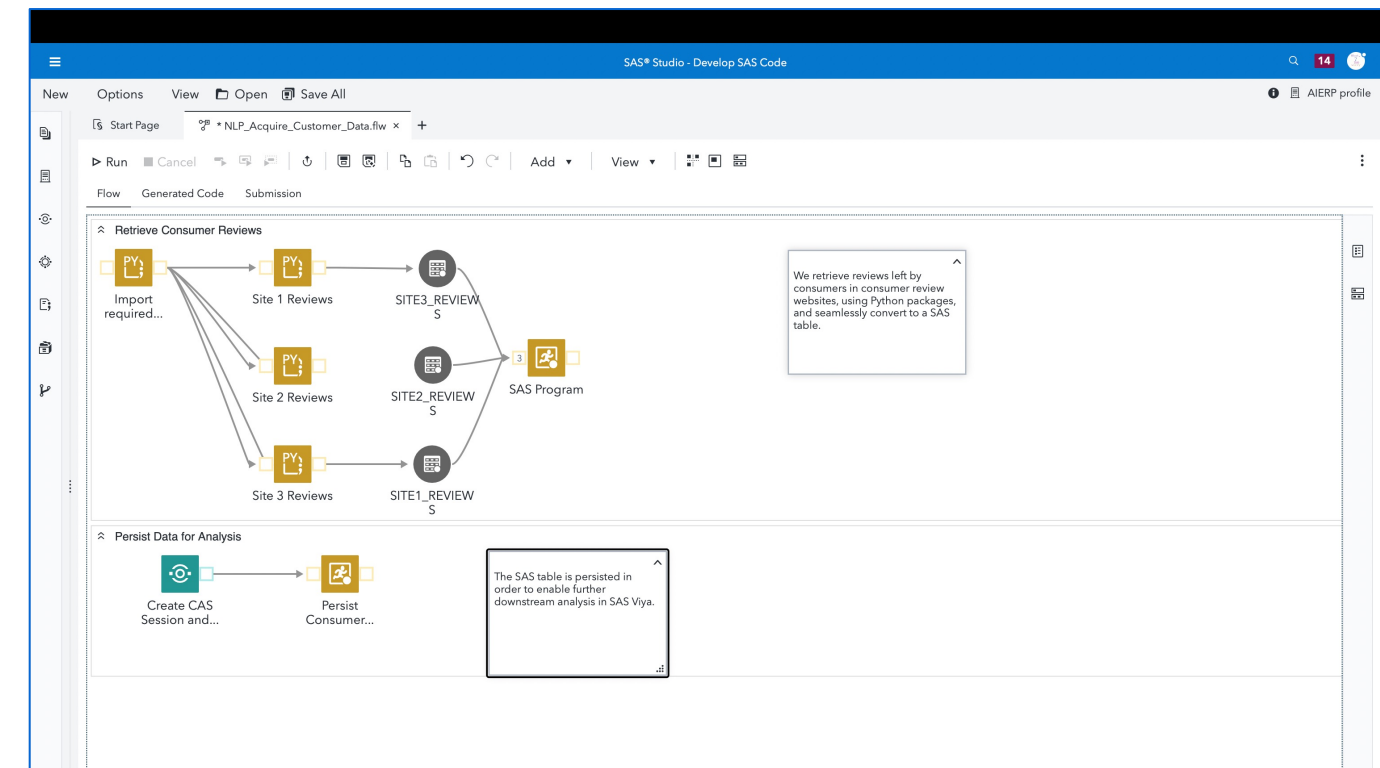
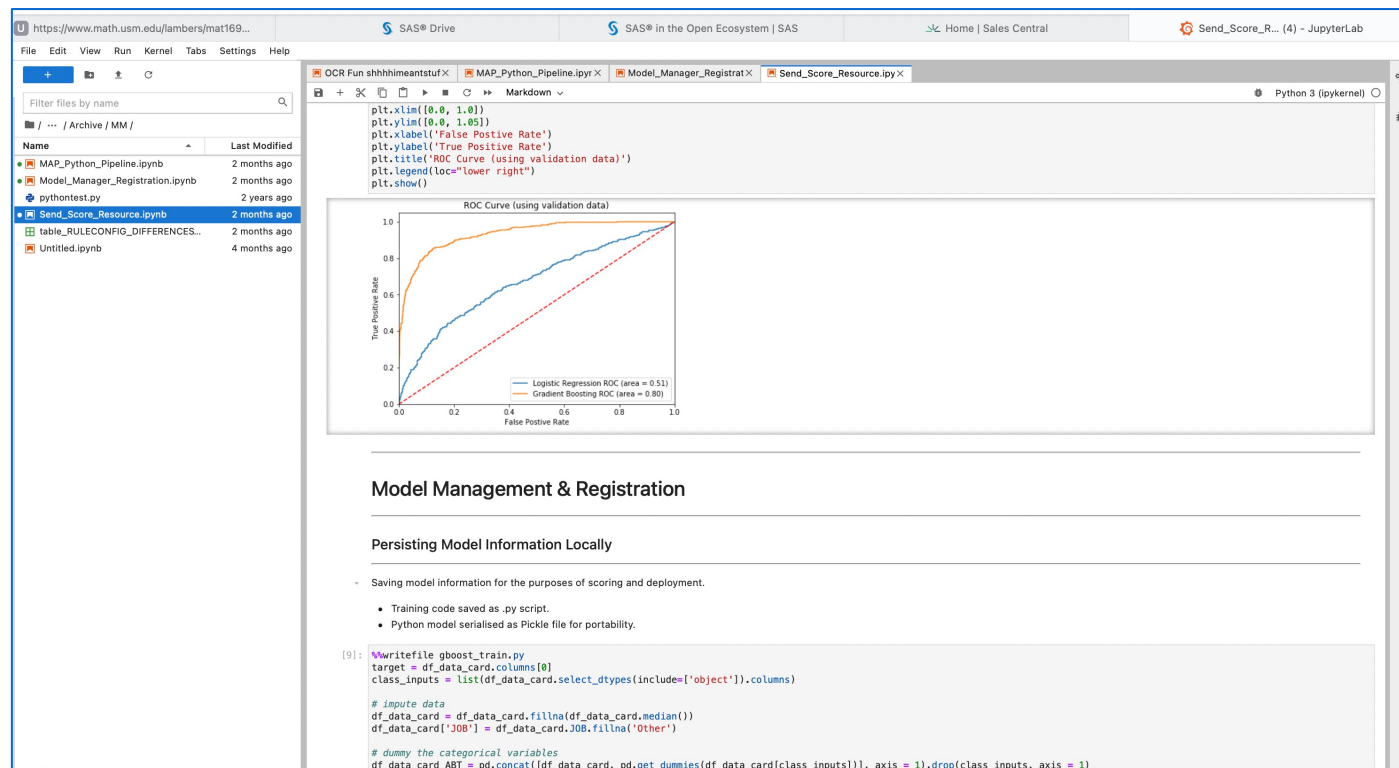
“I need a combination of Package X 0.2.3 with Package Y 0.3.4”

“I need to install and use a package for a limited period.”

“I have some specialized packages.”

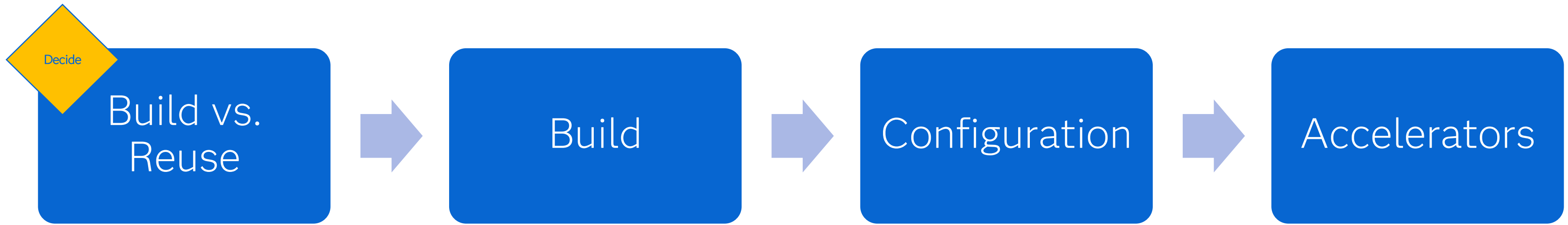
“I want to pass on a complete package to my colleague.”

SAS integrates with open source through many applications



Demonstration

A Path towards Integration



**Fresh Python
& R
environments**

**Existing
Python & R
environments**

**SAS
Configurator
for Open
Source**

- Install Python & R environments on accessible paths
- Locations, versions, profiles & packages

**sas-open-
source-config
transformers**

- Point / direct Viya to desired Python & R location(s).
- Environmental variables

**Custom
Steps,
packages &
methods**

- SAS callback methods, proc python, swat
- R Runner, Virtual Python environments and others

Build vs. Reuse Decision

Pros and Cons

Customers to decide based on organization's needs

Build Python & R

- Automated
- Cloud-native: up & down with the platform
- Easy to configure, multiple envs¹, isolation² & insulation³
- Additional admin overhead
- Egress / firewall permissions; pulls software from external
- Open-source vulnerabilities need to be handled separately

Reuse Existing Envs

- Continuity: reuse existing code & artifacts
- One less step; only configuration required
- Already validated / available
- Needs to be accessible to Viya
- Limited customization
- Change management / updates require coordination

- Pros
- Cons

Notes:

1. Referred to as multiple 'profiles'
2. Environments can exist separate from mainstream Python / R environments
3. Access to environments governed by symlinks which can be redirected to newer versions non-disruptively.

R Runner

Commonly (earlier) known mechanisms for executing R from SAS Studio

PROC IML

- Executes R from SAS Session
- Requires IML License
- Not easy to manage multiple R installations
- Getting out R variables into SAS is complicated
- Tough to orchestrate

X Command

- Executes R through command line
- Requires elevated IT permissions – a security challenge
- Not easy to manage
- Difficult to govern and monitor R sessions
- Tough to orchestrate

Introducing R Runner

Execute R from PROC Python

[github](#), [youtube](#)



1

Leverages rpy2

[rpy2](#) is a python package that can connect to a R session and execute R script



2

Custom Step

We introduce a custom step to execute R code through a self-service GUI & low code component.

3

Bring R output to SAS

Users can operate on data in R and bring back the results into SAS Studio session as SAS dataset

**Required packages are part of SAS Open Source configurator*

R Runner

▶ Run

■ Cancel

↺

↻

ParametersConfigurationAbout

Provide an input dataset:

SASHELP.HEART

🗨

Note: If you have attached input data to the "inputtable" input port of this custom step, ensure you refer to the same as r_input_table within your R script.

Provide your R script location: ⓘ

/path/to/your/R/Script

📁

Run an R snippet:

new_df=aggregate(x=r_input_table\$MSRP,by=list(r_input_table\$Origin),FUN=mean)

Provide name of R data frame you wish to output:

new_df

Note: If you are using this step, ensure that the R dataframe you refer does exist in the R session.

Provide output dataset for R environment variables:

WORK.R_ENV

🗨

Provide desired output table:

WORK.RDF

🗨

SAS table

R Script (type R code or select your R file)

SAS Dataset containing R variables

An R data frame exported as SAS table

R Runner: Demonstration

Benefits of unified environments

Customizable

Thank you

Contact: Sundaresh.Sankaran@sas.com; Samiul.Haque@sas.com

