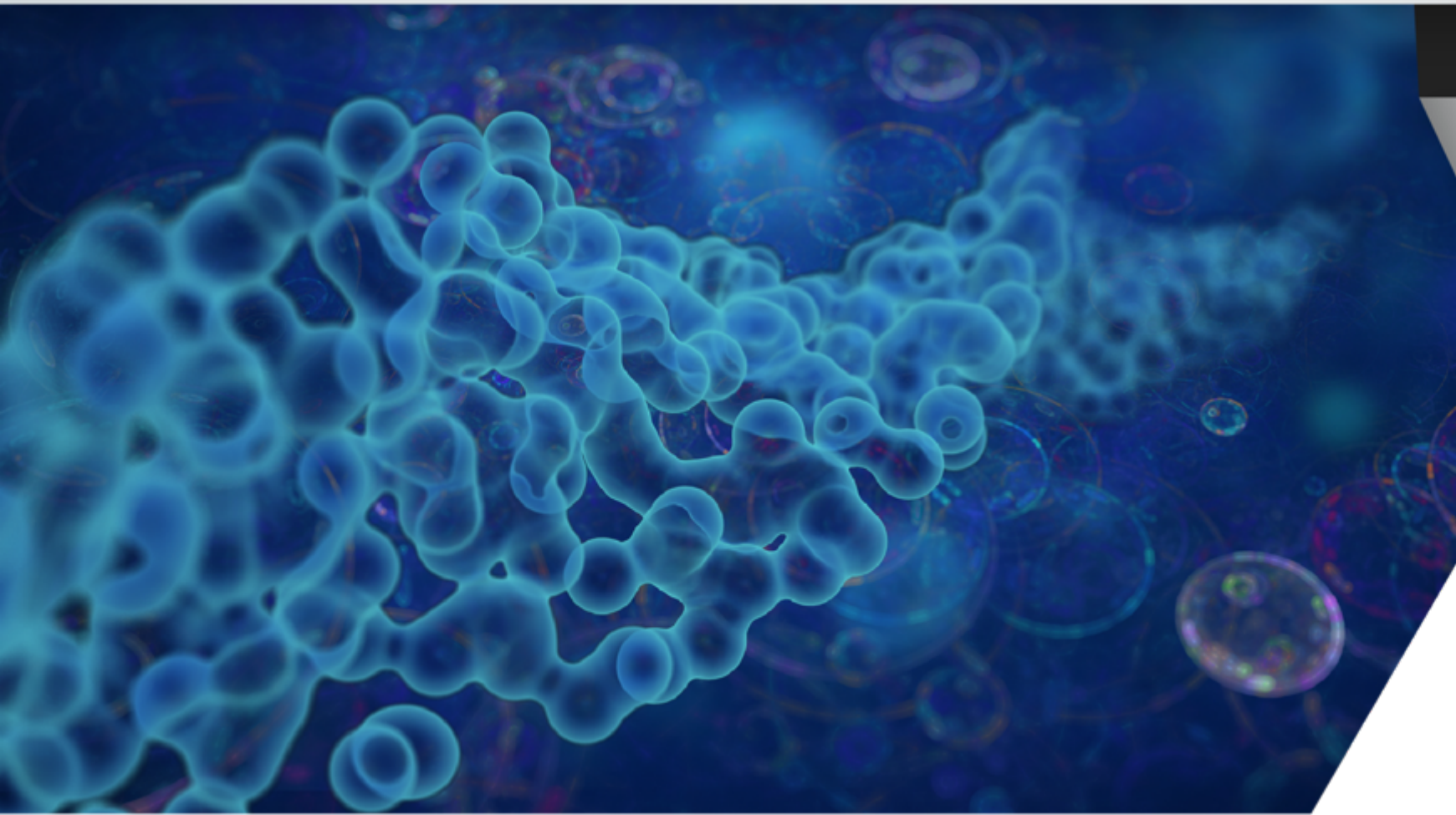
Making Python Objects R-like with pyR



Robert Kirk DeLisle
R/Pharma 2021

somalogic

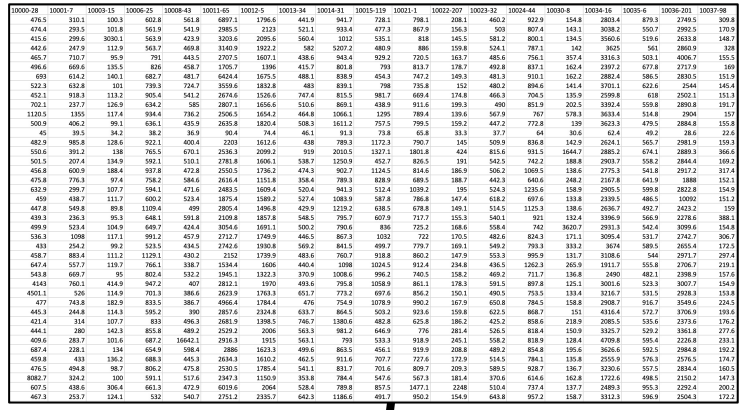
Who We Are and What We Do

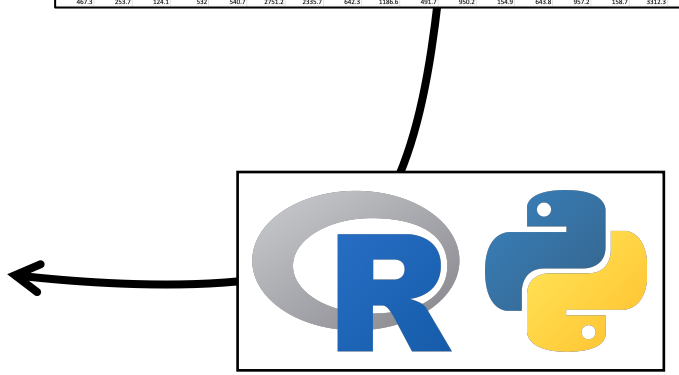
- The SomaScan Platform detects >7,000 proteins in a small biological sample (e.g., blood).
- Sample collections from different groups can be used to identify biomarkers or develop predictive models of biological endpoints (e.g., disease vs control, treated vs untreated, risk of developing disease, etc.)
- Mostly an R shop expanding usage of Python.
- We value automation, reproducibility, and standardized reporting using a shared code base.

+

SOMASCAN ASSAY

=





nature medicine

LETTERS

https://doi.org/10.1038/s41591-019-0665-2

Plasma protein patterns as comprehensive indicators of health

Stephen A. Williams[1,12*], Mika Kivimaki[2], Claudia Langenberg[3], Aroon D. Hingorani[4,5,6], J. P. Casas[7], Claude Bouchard[8], Christian Jonasson[9], Mark A. Sarzynski[10], Martin J. Shipley[2], Leigh Alexander[1], Jessica Ash[1], Tim Bauer[1], Jessica Chadwick[1], Gargi Datta[1], Robert Kirk DeLisle[1], Yolanda Hagar[1], Michael Hinterberg[1], Rachel Ostroff[1], Sophie Weiss[1], Peter Ganz[11,12] and Nicholas J. Wareham[3,12]

Integrating Python with R using *reticulate*

```
library(reticulate)

# set my conda environment
reticulate::use_condaenv('scikitlearn', required = TRUE)

# load the 'linear_model' module of SciKit-Learn
sklearn <- import('sklearn.linear_model')

# Create a logistic regression object
py_linreg <- sklearn$LinearRegression()

# fit the model and get some predictions
py_linreg$fit(X = training_X, y = training_Y)

py_preds <- py_linreg$predict(test_X)

#######################################
# compared to R-based linear regression
r_linreg <- lm(Y ~ ., data = training)

r_preds <- predict(r_linreg, newdata = test_X)
```

- Very direct and concise code to make Python modules available to R
- Direct access to Python object methods
- Code integrates seamlessly

- Requires use of *`import()`* and creation of Python object within code block
- Access to Python object methods goes through the $ operator – this makes the code inconsistent with typical R usage

It would be nice if Python portions behaved more like R code for consistency and to enable automation across R and Python objects.

Model Class Hierarchy

Model Abstract Base Class

Defines the functions (interface) that *every* model must support, regardless of type

Does not provide function implementations

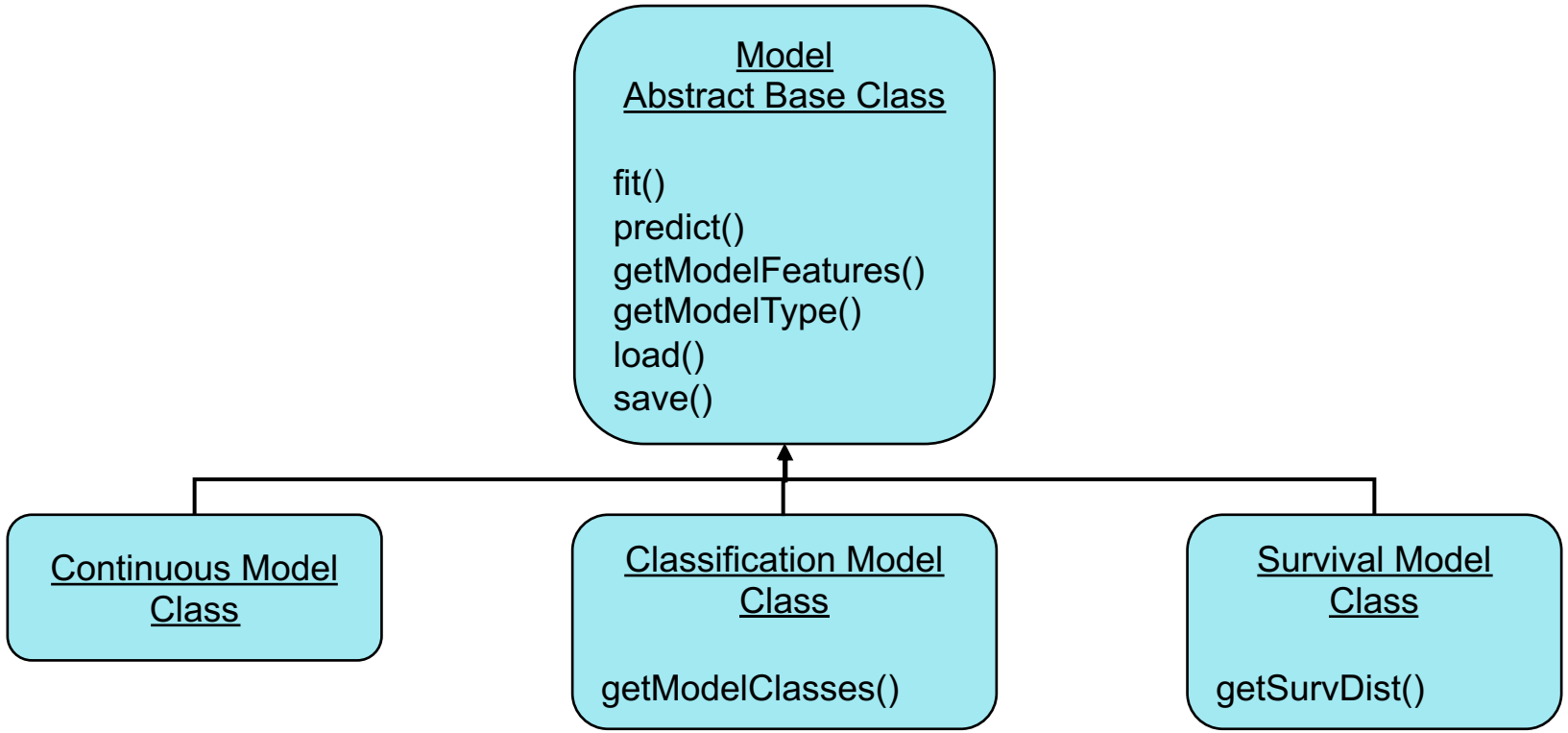
Derived Classes

encapsulate specific models (from R or Python)

Inherit interface from Model Abstract Base Class

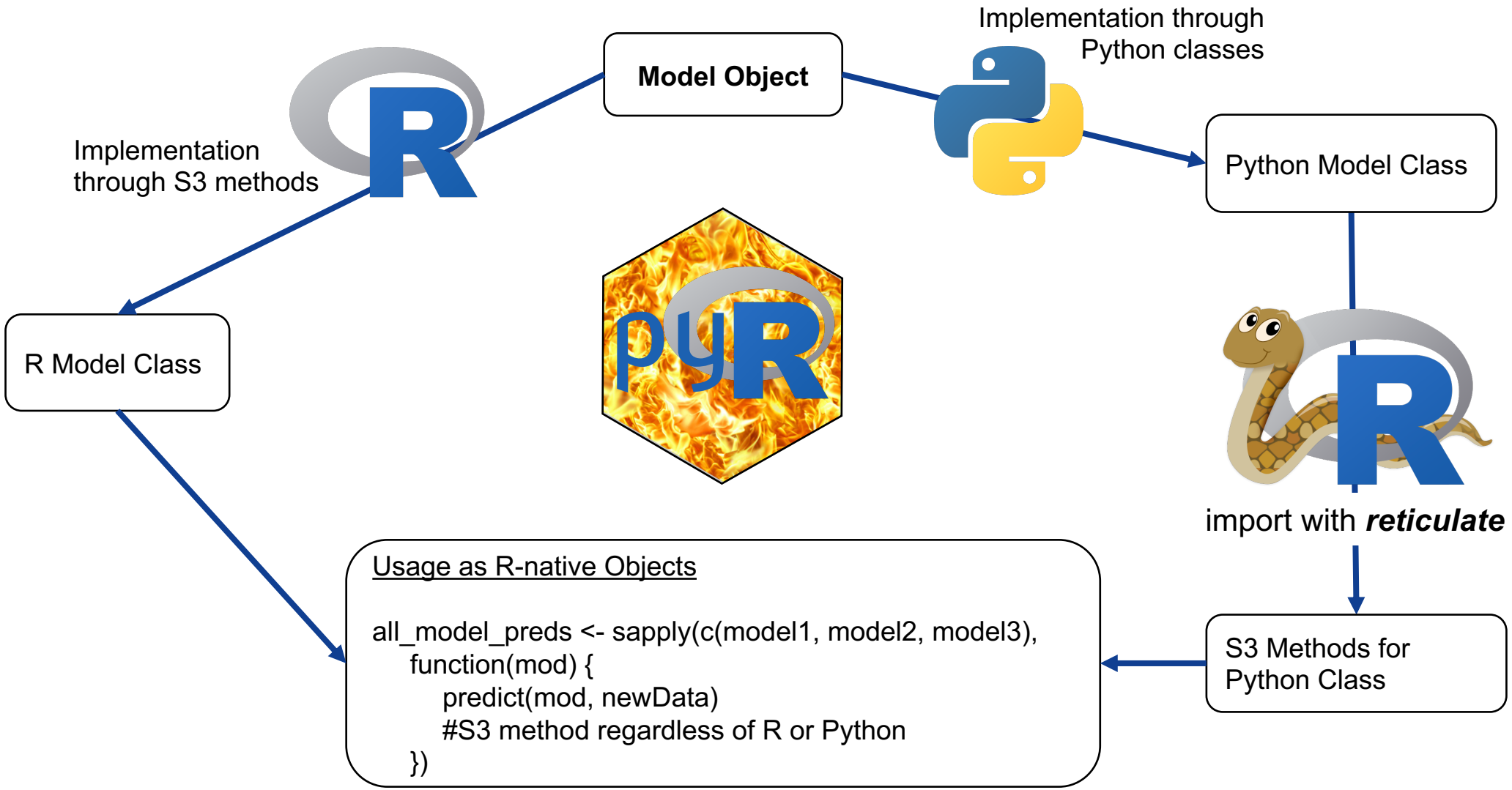
Defines the interface for that specific model type by providing additional specialization functions, if needed

Provide implementations of all functions for that model type (e.g., classification, regression, survival)



somalogic

pyR Architecture Concept



Any code can expect the Model Object Interface to be present – this enables all models to be used consistently, and *non-R objects to be used as if they were native R objects.*

somalogic

pyR Object Factory

Saved pyR objects

pyR_GPFlow_Class.pyr

pyR_TPOT_Class.pyr

New pyR objects

"pyR_logreg"

pyR ObjectFactory knows which class to build from saved objects, or how to create new objects from scratch.

pyR S3 Objects

pyR GPFlow_Class Object

pyR TPOT_Class Object

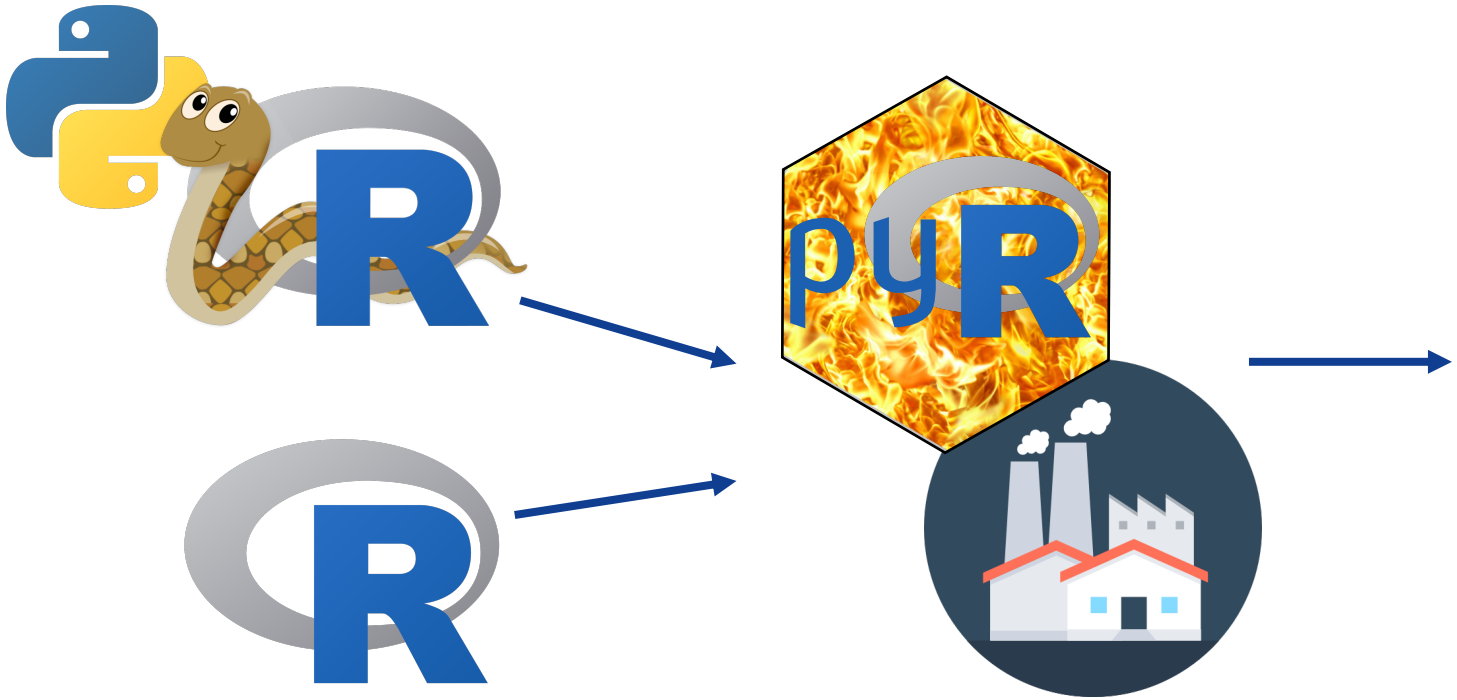
pyR Logistic Regression Object

Common usage as native R objects

```
# get saved classifier models
model_gp <-
pyR_ObjectFactory.loadObject(
     'pyR_GPFlow_Class.pyr')

model_tpot <-
pyR_ObjectFactory.loadObject(
     'pyR_TPOT_Class.pyr')

# create a new classifier model
model_logreg <-
pyR_ObjectFactory.newObject(
     'pyR_logreg')
fit(model_logreg, training_data)

# generate predictions for all
preds <- sapply(model_gp,
   model_tpot, model_logreg),
     function(fn) {
          predict(mod, newData)
     })
```

somalogic

Unification of R and Python with pyR



R-native S3 Objects
- Consistency
- Shareability
- Reproducibility
- Standardization

```
all_model_preds <-
  sapply(c(model1, model2, model3),
    function(mod) {
      predict(mod, newData)
      #S3 method regardless of R or Python
    })
```

Thanks to my colleagues!
Laura Sampson
Leigh Alexander
Stu Field

Robert Kirk DeLisle
kdelisle@somalogic.com
KirkD_CO

somalogic