Interfacing between R and Python

Hoofar Pourzand
Research Computing and Cyberinfrustructure Group
Pennsylvania State University

June 28, 2013

Abstract

R has a rich developing community and consequently a growing list of libraries, methods for data mining & statistical analysis of data. Also, Python is a popular interpreted language used in many back-end and front end web-development technologies, as well as data mining tools & libraries that easily incorporates Object oriented design concepts into a code. Calling R from Python, for a very simple example is shown here to give Python the extra computational edge that R already provides.
RPy is the interface between R and Python which is used for this task.

1 Preparation

Installing RPy If you are running Ubuntu, simply type:

Listing 1: Installing RPy

```
$sudo apt-get install python-rpy
```

To load RPy from Python, whether in interactive mode or in Batch (PBS) mode, to your python script just add:

Listing 2: Loading PPy

```
>>>from rpy import*
```

This will load a Python class instance r.

Running RPy From Python prompt type in:

Listing 3: Running RPy

```
>>>r.hsit(r.rnorm(10), main= '', xlab= '')
>>>a = [5,12, 13]
>>>b= [10,28,30]
>>>lmout = r.lm('v2~v1' , data = r.data_frame(v1=a, v2= b))
```

Note R function names are prefixed by *r.* . Note Python does not include a tilde character, in these cases we need to specify the model formula via a string. For calling R functions in Python that have a period in their R function name, an underscore is substituded, e.g. $data\_frame()$ The output object is a Python Dictionary- closest to an R list type.
To access the results accordingly, just type in the attributes title:

```
lmout['coefficient']
>>>lmout[coefficients']['v1']
```

To avoid syntax clashes between Python and R, you can submit R commands to work on R namespaces by using the function r().

Listing 4: Calling Python in R Example

```
>>>r.library('lattice')
>>>r.assign('a',a)          ***copy a variable from Python's
   namespace to R.
>>>r.assign('b', b)
>>>r('g <- expand.grid(a,b)')   *** assigning the result to g
   in R's namespace.
>>>r('g$Var3 <- g$Var1^2 +g$Var1 * g$Var2')
>>>r('wireframe(Var3~Var1 + Var2, g)')
>>>r('plot(wireframe(Var3 ~Var1 +Var2, g)))      *** wireframe
   () didn't diplay.
```

References

[1] http://rpy.sourceforge.net/rpy/doc/rpy.pdf

[2] http://www.daimi.au.dk/~besen/TBiB2007/lecture-notes/rpy.html