WS_Talk

May 28, 2019

1 Electronic Notebooks

1.1 Jupyter Notebook

1.1.1 Dr. Ugur Ozdemir / PIR

(Some slides are taken from Vernon Gayle's JN talk)

1.2 Overview

- The Jupyter Notebook is an open-source web application that allows you to **create and share documents** that contain live code, equations, visualizations and narrative text.
- Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.
- Data analysis heavy scholarly writing, blog posts, presentations, lecture notes etc.
- The Jupyter Notebook currently supports **interactive data science** and **scientific computing** across over 40 programming languages.
- The computer languages **Julia Python and R almost spell *JuPyteR***

1.3 Structure of a Jupyter Notebook

A Jupyter Notebook is made up of **cells.**
A cell can contain either
i. live research code (e.g. R syntax) that can be executed
ii. text comments that form the documentation of the research workflow
iii. cells that contain the results of data analyses
In addition to running your code the Notebook frontend stores code and output, together with markdown notes, in an editable document called a notebook. When you save it, this is sent from your browser to the notebook server, which saves it on disk as a JSON file with a .ipynb extension.

1.4 Attractive Features

Jupyter Notebooks have a number of attractive features

1. Easy documentation alongside research code
2. 'Language agnostic' 40+ languages. (Easily switch kernels)

3. Rich visual outputs
4. Big data tools e.g. python
5. Teaching and training
6. Collaborative work
7. Portability (publication) easy to share

1.5 Markdown

Markdown is an easy way to write documents.

It is written in what computer geeks like to call 'plaintext'. It is the sort of text that we are used to writing and seeing.

Plaintext is just the regular alphabet plus a few other familiar symbols (for example the asterisk *).

Unlike cumbersome word processing applications, text written in Markdown can be easily shared between computers.

It's quickly becoming the writing standard in some academic areas and in science.

1.6 R Analysis

You must have R installed on your machine.

You must have installed the R kernel (See https://anaconda.org/r/r-irkernel).

Reminder *Switch Kernel to R < Menu kernel - change kernel>*

In [1]:
```
library(foreign)
library(survey)
```

```
Loading required package: grid

Attaching package: 'survey'

The following object is masked from 'package:graphics':

    dotchart
```

In [2]:
```
mydata <- read.dta("http://www.vernongayle.com/uploads/2/2/3/0/22304498/wemp.dta")
summary(mydata)
```

Out[2]:
```
      case             femp              mune               time
 Min.   :   1.0   Min.   :0.0000   Min.   :0.00000   Min.   : 0.0
 1st Qu.: 274.0   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.: 4.0
 Median : 538.0   Median :1.0000   Median :0.00000   Median : 8.0
 Mean   : 517.7   Mean   :0.6456   Mean   :0.07405   Mean   : 7.2
 3rd Qu.: 753.0   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:11.0
 Max.   :1003.0   Max.   :1.0000   Max.   :1.00000   Max.   :13.0
```

```
      und1              und5              age
 Min.   :0.00000   Min.   :0.0000   Min.   :18.00
 1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:29.00
 Median :0.00000   Median :0.0000   Median :35.00
 Mean   :0.07468   Mean   :0.2975   Mean   :36.01
 3rd Qu.:0.00000   3rd Qu.:1.0000   3rd Qu.:43.00
 Max.   :1.00000   Max.   :1.0000   Max.   :60.00
```

Estimating the logit model and sending it to the object "mylogit".

In [3]:
```
mylogit <- glm(femp ~ mune + und5, data = mydata, family = "binomial")

summary(mylogit)
```

Out[3]:
```
Call:
glm(formula = femp ~ mune + und5, family = "binomial", data = mydata)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7586  -1.0024   0.6922   0.6922   2.1177

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.30683    0.07442  17.561  < 2e-16 ***
mune        -1.70331    0.23585  -7.222 5.12e-13 ***
und5        -1.73352    0.12219 -14.187  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2054.5  on 1579  degrees of freedom
Residual deviance: 1757.4  on 1577  degrees of freedom
AIC: 1763.4

Number of Fisher Scoring iterations: 4
```

1.7 Python Analysis

WARNING *Switch Kernel to Python < Menu kernel - change kernel>*
Switch Kernel to Python 3 < Menu kernel - change kernel>

In [10]:
```
import pandas as pd
```

Construct the data frame "df" reading in the data from an Excel (xlsx) file. It could also be read in from a csv file.

```
In [11]: df = pd.read_excel("http://www.vernongayle.com/uploads/2/2/3/0/22304498/wemp.xlsx")
         df.head()
```

Out[11]:

	case	femp	mune	time	und1	und5	age
0	1	1	0	10	1	1	23
1	1	0	0	11	0	1	24
2	1	0	0	12	0	1	25
3	1	0	0	13	0	1	26
4	6	1	0	0	0	0	42

Python is more general purpose and not primarily orientated towards social science data analysis. Therefore some things are a little more fiddly. For example we must set a constant for all case (int=1).

```
In [5]: df['Int']=1
```

Examining the data in the data frame "df".

```
In [6]: df.head()
```

Out[6]:

	case	femp	mune	time	und1	und5	age	Int
0	1	1	0	10	1	1	23	1
1	1	0	0	11	0	1	24	1
2	1	0	0	12	0	1	25	1
3	1	0	0	13	0	1	26	1
4	6	1	0	0	0	0	42	1

Import the package "statsmodels".

```
In [7]: import statsmodels.api as sm
```

Estimate a logistic regression model the independent variables are "mune" "und5" and "int" the outcome variable is "femp".

```
In [8]: independentVar = ['mune', 'und5', 'Int']
        logReg = sm.Logit(df['femp'] , df[independentVar])
        answer = logReg.fit()
```

```
Optimization terminated successfully.
         Current function value: 0.556127
         Iterations 5
```

The results are in the object "answer".

```
In [9]: answer.summary()
```

```
Out[9]: <class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:                   femp   No. Observations:                 1580
Model:                          Logit   Df Residuals:                     1577
Method:                           MLE   Df Model:                            2
Date:                Tue, 24 Jan 2017   Pseudo R-squ.:                  0.1446
Time:                        09:56:39   Log-Likelihood:                -878.68
converged:                       True   LL-Null:                       -1027.2
                                        LLR p-value:                 3.056e-65
==============================================================================
                 coef    std err          z      P>|z|      [95.0% Conf. Int.]
------------------------------------------------------------------------------
mune          -1.7033      0.236     -7.222      0.000        -2.166    -1.241
und5          -1.7335      0.122    -14.187      0.000        -1.973    -1.494
Int            1.3068      0.074     17.561      0.000         1.161     1.453
==============================================================================
"""
```

1.7.1 Summary

This example was designed to demonstrate that Jupyter Notebooks are language agnostic.

The language agnostic aspects of Jupyter Notebooks mean that they could be an appropriate unified environment in which to undertake research analyses using alternative software packages and languages.

This feature is especially attractive in some collaborative endeavours. For example Stata is the primary data analysis software package at the ADRC-Scotland. From time to time there may be a requirement to used other data analysis tools.

1.8 Videos

```
In [1]: from IPython.display import YouTubeVideo
        YouTubeVideo('p47tetYy7co')
```

```
Out[1]:
```

1.9 LaTeX

```
In [2]: %%latex
        \begin{align}
        a = \frac{1}{2} && b= \frac{1}{2} && c = \frac{1}{4} \\
        \end{align}
```

$$a = \frac{1}{2} \qquad b = \frac{1}{2} \qquad c = \frac{1}{4} \tag{1}$$

(2)

```
In [3]: %%latex
        $e^{i\pi} + 1 = 0
        $
```

$e^{i\pi} + 1 = 0$

2 'Widgets'

Widgets are clever devices that can be included in notebooks to help users visualize and control changes in the data. Widgets may be useful in teaching and training because users can easily see how changing inputs to something impacts on the results.

2.0.1 An Interesting Wee Widget

```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        from numpy import pi, exp, real, imag, linspace
        from ipywidgets import interact

        def f(t,a=1,b=6,c=-14,d=0):
            return exp(a*1j*t) - exp(b*1j*t)/2 + 1j*exp(c*1j*t)/3 + exp(d*1j*t)/4

        def plot_swirly(a=1,b=6,c=-14,d=0):
            t = linspace(0, 2*pi, 1000)
            ft = f(t,a,b,c,d)
            plt.plot(real(ft), imag(ft))

            # These two lines make the aspect ratio square
            fig = plt.gcf()
            fig.set_size_inches(6, 6, forward='True')

        interact(plot_swirly,a=(-20,20),b=(-20,20),c=(-20,20),d=(-20,20));
```

```
interactive(children=(IntSlider(value=1, description='a', max=20, min=-20), IntSlider(value=6, d
```

2.0.2 The Sine Wave Example

```
In [2]: from ipywidgets import widgets
        import numpy as np
        import matplotlib.pyplot as plt
        from IPython.display import display
        from numpy import arange, sin, pi
        %matplotlib inline
```

```
In [3]: from IPython.html.widgets import *
        t = arange(0.0, 1.0, 0.01)

        def pltsin(f):
            plt.plot(t,sin(2*pi*t*f))
            plt.show()

        interact(pltsin, f=(1,10,0.1))
```

```
/opt/conda/lib/python3.6/site-packages/IPython/html.py:14: ShimWarning: The `IPython.html` packa
  "`IPython.html.widgets` has moved to `ipywidgets`.", ShimWarning)
```

```
interactive(children=(FloatSlider(value=5.0, description='f', max=10.0, min=1.0), Output()), _do
```

```
Out[3]: <function __main__.pltsin>
```