Lecture 12: Python & Stata Basics

Economía Laboral

Junghanss, Juan Cruz

Universidad del CEMA

2nd Semester 2022

Contents

Today's lecture content:

- Python Basics
- Python in R?
- Stata Basics
- Stata in R?

Python Basics

Python is a general-purpose and object-oriented programming language. According some indexes, it's the world's most popular programming language before Java, C and Javascript.

Characteristics:

- Easy to Code: high-level programming language.
- Easy to Read: looks like simple English words.
- Open-Source: it's free.
- Robust Standard Library
- Object- and Procedure-Oriented, GUI programming support, integrated, interpreted, etc...

Python Basics

Applications:

1. Scientific and Numeric Applications
2. Artificial Intelligence and Machine Learning
3. Software Development
4. Web & Apps Development
5. Game Development
6. etc...

Python Basics

1. Scientific and Numeric Applications

Most common libraries:

- **Numpy**: handling large dimensional arrays.
- **Pandas**: data manipulation and analysis.
- **Matplotlib**: data visualization.

Python Basics

2. Artificial Intelligence and Machine Learning

Most common libraries:

- **Numpy**: handling large dimensional arrays.
- **Pandas**: data manipulation and analysis.
- **Matplotlib**: data visualization.
- **Statsmodels**: statistical tests, exploration and modelling.
- **Scikit-learn**: data mining and machine learning techniques.
- **Keras**: high-level neural networks API. Similar to **TensorFlow**, **PyTorch**, **Caffe2**, etc.
- etc.

Python Basics

Other useful libraries:

- QuantEcon: game theory, optimization, econometric tools, etc.
- Yfinance: yahoo finance library for market data
- Pyfolio: risk analysis and performance reports of financial portfolios
- Zipline: backtesting and live trading
- SQLAlchemy: database abstraction library
- BeautifulSoup: web scraping

Python Basics - Sintaxis

Asignación de variables e impresión:

Example

```
x = "Esto es un string"
print(x)

# Definición múltiple:
x, y = "Esto es un string", 456
```

Con ";" podemos hacer saltos de linea y escribir todo en un mismo renglon:

Example

```
x, y = "Esto es un string", 456 ; print(x,y) ; print(12*y)
```

Python Basics - Sintaxis

Instalación e importación de librerias:

Example

```
pip install pandas # Para instalar librerias

import pandas # Para importar una libreria

import pandas as pd # Para importar una libreria abreviando
su nombre a la hora de usarla!
```

Python Basics - Help and Documentation

De manera análoga a R con "?", pueden buscar ayuda en Python con el comando help().

Example

```
help(pandas)
```

Python help() function is used to get the documentation of specified module, class, function, variables etc. This method is generally used with python interpreter console to get details about python objects.

Python Basics - Condiciones lógicas

If statements:

Example

```
if a > b and c > a:
    print(‘‘Both conditions are True’’)
```

More generally:

Example

```
if expression1:
    statement(s)
elif expression2:
    statement(s)
else:
    statement(s)
```

Python Basics - Funciones

Functions: you can code multiple arguments functions

Example

```
def tri_recursion(k):
    if(k > 0):
       result = k + tri_recursion(k - 1)
       print(result)
    else:
       result = 0
    return result
```

This particular example is a recursive function, which means the defined function calls itself.

Python Basics - Numpy

NumPy is the fundamental package for scientific computing in Python (NUMerical PYthon). Numpy main functions:

- Linear Algebra: dot-, inner-, outer-products. Eigenvalues, Inverse, Linear solvers, etc.
- Math: Sums, products, differences, exponents, logarithms, complex numbers, etc.
- Stats: Averages, variances, correlations, histograms.

More info in NumPy documentation.

Python Basics - Pandas

Pandas provides high-performance, easy-to-use data structures and data analysis tools for Python. Pandas main functions:

- Creation and operations of DataFrames and Series.
- Data files reading and parsing (csv, xlsx, etc.)
- Groupby, summary stats, etc.

More info in Pandas documentation.

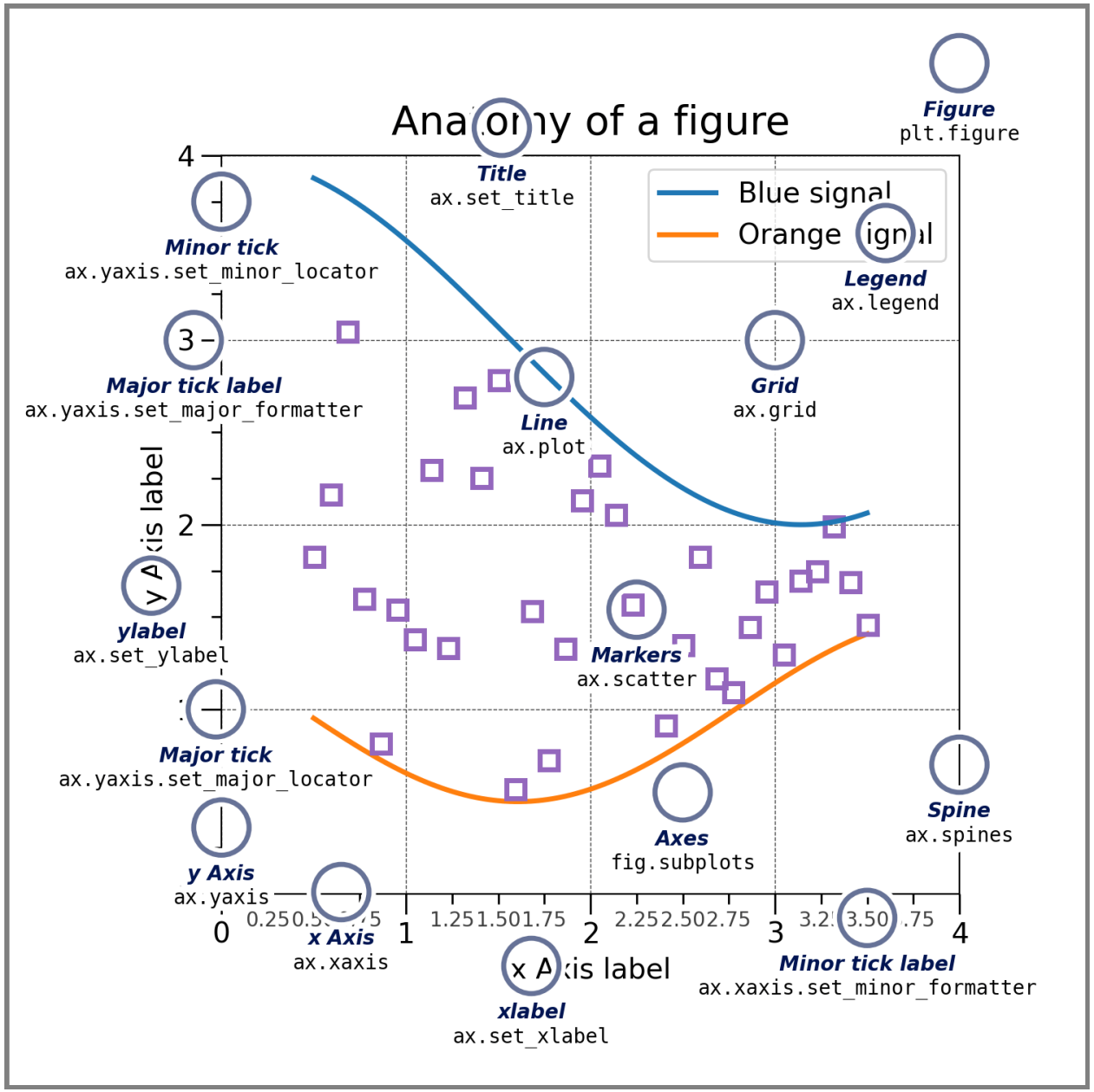
Python Basics - Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib main functions:

- Creation of plots and images.
- Stylize every single part of the plot.
- Creation of animated plots.

More info in Matplotlib documentation.

Python Basics - Matplotlib



Python Basics - Jupyter Notebooks

Before jumping into real coding, let's see what kind of python files you can handle:

- .py is the normal python file (like .R)
- .ipynb is the "iPython notebook" file (like .Rmd)

As it's mostly used in every university, course, etc. we'll see specifically **Jupyter Notebooks** for ipynb files. The Jupyter Notebook is a web application for creating and sharing computational documents combining live code, equations, narrative text, visualizations, etc.

You can easily have Jupyter Notebooks with the Anaconda package for Python.

Python Basics

Now that you get the basics, we can check some interesting Python code.

Consideremos un código en Python vinculado con economía sobre representaciones conceptuales del mercado laboral realizado por Nicolas Cachanosky, PhD:

Labor Market de Nicolas Cachanosky

Python & R relation

In data analysis and data science there's a huge debate around Python *versus* R. However, we'll see you can work with both.

First, let's dig into the main differences of both languages.

Python & R relation

R	**Python**
More maintenance.	Robust and easier to maintain.
Statistical language and also used for graphical techniques.	General-purpose language for development and deployment.
Better for data visualization.	Better for machine learning.
R is easy to start with.	Python libraries can be a bit complex.
R is a command line interpreted language.	Simple syntax.
Powerful statistical packages.	Python's statistical packages are less powerful.
R is slower than python.	Python is faster.
Easy to use for complicated mathematical calculations and statistical tests.	Good for building something new from scratch. Used for application development.
Medium popularity	High popularity

Python & R relation

Python is a multi-purpose language, much like C++ and Java, with a readable syntax that's easy to learn. Programmers use Python to delve into data analysis or use machine learning in scalable production environments. For example, you might use Python to build face recognition into your mobile API or for developing a machine learning application.

Python & R relation

Python is a multi-purpose language, much like C++ and Java, with a readable syntax that's easy to learn. Programmers use Python to delve into data analysis or use machine learning in scalable production environments. For example, you might use Python to build face recognition into your mobile API or for developing a machine learning application.

R, on the other hand, is built by statisticians and leans heavily into statistical models and specialized analytics. Data scientists use R for deep statistical analysis, supported by just a few lines of code and beautiful data visualizations. For example, you might use R for customer behavior analysis or genomics research.

Python in R?

The **reticulate** package in R provides a comprehensive set of tools for *interoperability* between Python and R. The package includes facilities for:

- Calling Python from R in a variety of ways including R Markdown, sourcing Python scripts, importing Python modules, and using Python interactively within an R session.
- Translation between R and Python objects (for example, between R and Pandas data frames, or between R matrices and NumPy arrays).
- Flexible binding to different versions of Python including virtual environments and Conda environments.

Python in R?

Sourcing Python scripts: You can source any Python script just as you would source an R script using the source_python() function.

In our example we'll explore how to source two files in R and Python and work with their objects in our current project.

Python in R?

Python REPL:
If you want to work with Python interactively you can call the repl_python() function, which provides a Python REPL embedded within your R session. Objects created within the Python REPL can be accessed from R using the py object exported from reticulate.

Stata Basics

Stata (from "Statistics" and "Data") is a general-purpose statistical software for data manipulation, visualization, statistics, and automated reporting.

Characteristics:

- Licensed: developed by StataCorp, you have to pay in order to download/use it.
- Used mostly by researchers/academics.

Stata Basics

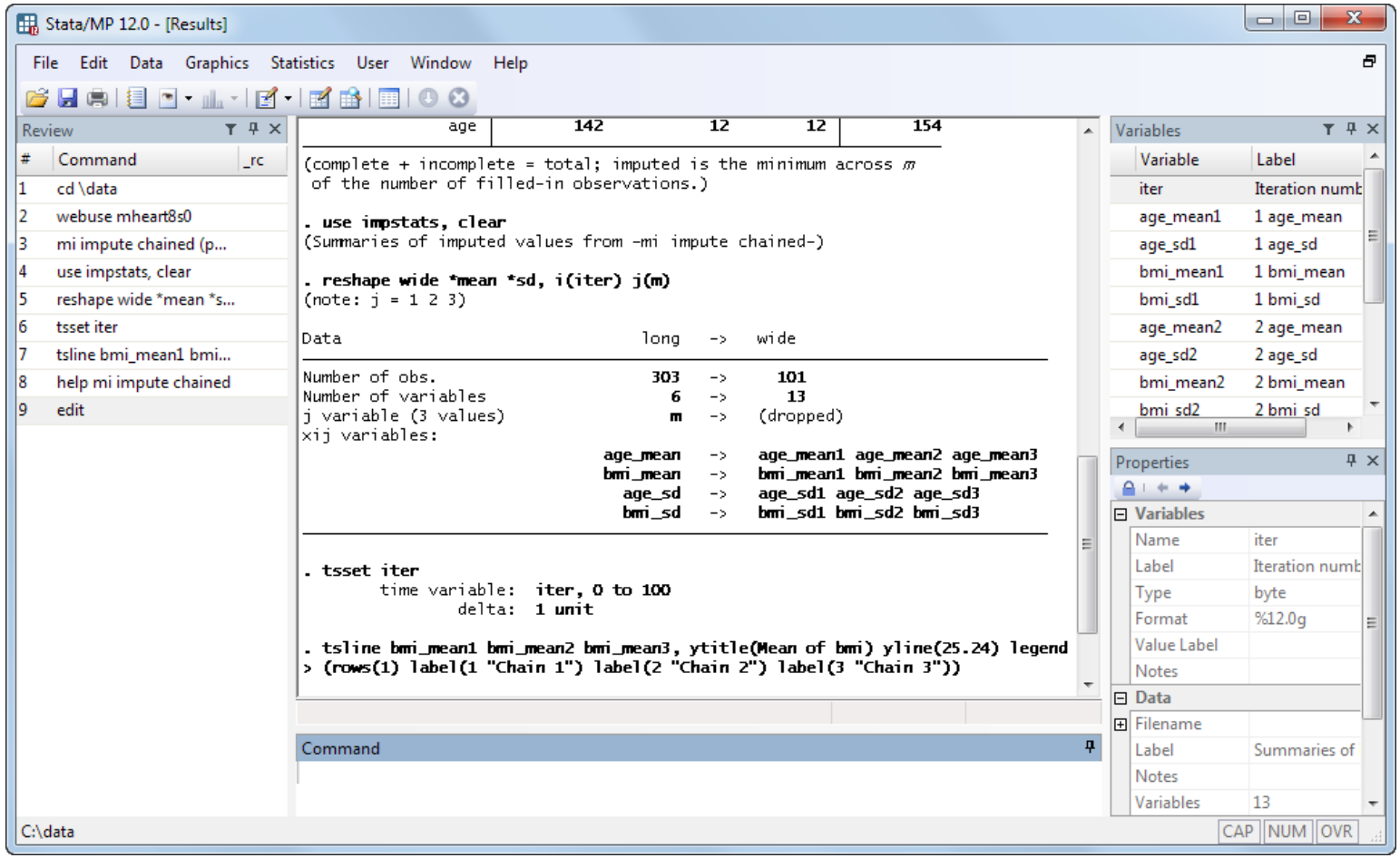
El tipo de archivos que utiliza Stata son:

- dta: archivos de datos
- do: archivos de comandos
- ado: programas
- hlp: archivos de ayuda
- gph: gráficos
- dct: archivos diccionarios
- smcl: archivos log

El archivo .do sería como el file .R que generamos con nuestro código. Por otro lado, la base de datos puede ser un file .dta en vez de .csv, .xlsx u otros a los que estamos acostumbrados. En Python, por ejemplo, pueden usar la libreria Pandas para abrir archivos .dta.

Stata Basics

A diferencia de R y Rstudio, no hace falta descargar ningun IDE para codear en Stata, ya que al ser un programa íntegro viene con su propia interfaz de usuario:



Stata Basics

Otro tipo de programas estadísticos con licencia como Stata que pueden llegar a escuchar alguna vez son:

- Minitab
- Eviews
- SAS
- SPSS

Stata Basics

Recomendación:

Si algun día deben comenzar a utilizar Stata regularmente, una buena fuente de tutoriales para aprender múltiples técnicas la tienen en el canal de YouTube oficial de Stata.

Stata in R?

To work with Stata code in R you have some options:

- **Rstata** package (*recommended*)
- RStudio uses an R package called **knitr** (this could also be called directly from R), which includes the ability to evaluate Stata

Stata in R with Rstata

Exactly like reticulate for Python, you'll need first to specify the Stata path and version.

Example

```
options("RStata.StataPath" = "/Applications/Stata/StataMP.app/
options("RStata.StataVersion" = 16)
```

After that, you can begin writing Stata code with stata() function:

- Running single line commands
- Running many lines commands
- Running an entire .do file

So... R, Python or Stata?

Consejos:

- Intenten especializarse *bien* en un **único** lenguaje, ya que migrar a otro eventualmente en un trabajo o proyecto va a ser muy fácil.
  - Elijan el que se ajuste mejor a sus **objetivos**.
- Sin embargo, traten de "*tocar de oído*" **librerias y características** de otros lenguajes de programación.
- Guardense cheatsheets, material e info sobre los lenguajes que no dominen porque les será muy valioso si deben aprender en el futuro.