Python Workshop: Introduction

Ties de Kok

Tilburg University

About me

TIES DE KOK

Home Publications Projects Teaching C.V.

Ties de Kok

PhD in Accounting

Tilburg University

Biography

Ties de Kok is a PhD researcher at Tilburg University that specializes in combining computer science with empirical Accounting research. His research interest is in financial accounting, capital markets, empirical management accounting, computer science, and natural language processing.

My university page is located here:
https://www.tilburguniversity.edu/webwijs/show/t.c.j.dekok.htm

Interests

- Financial Accounting
- Management Accounting
- Computational Linguistics
- Big Data

Education

Research Master in Accounting
Tilburg University (2013-2015)

Bachelor Business Administration
Tilburg University (2010-2013)

What will we be doing?

Four main blocks:

1) Introduction to Python (+ Python worfklow)
   - Today

2) Handling data with `Pandas`
   - Today

3) Gathering data from the web
   - Thursday

4) Natural Language Processing
   - Friday

Basic Principles of this course:

1) I cannot inject you with Python skills

2) **It is up to you to make yourself proficient with Python**

Basic Principles of this course:

1) I cannot inject you with Python skills

2) **It is up to you to make yourself proficient with Python**

My goal:

- Make it more efficient for you to **teach Python to yourself**

Basic Principles of this course:

1) I cannot inject you with Python skills

2) **It is up to you to make yourself proficient with Python**

My goal:

- Make it more efficient for you to **teach Python to yourself**

How?

1. By providing starting points
2. By pointing out common pitfalls

Structure:

Each block consists of three elements:

1) Conceptual introduction

- Introduce basic constructs and terminology

2) Setup + Get started

- Make sure everything is setup and working

3) Mini-task

- Get hands-on experience

Slides:

All of the slides are made available here: GitHub page

Python materials:

All materials are available here:

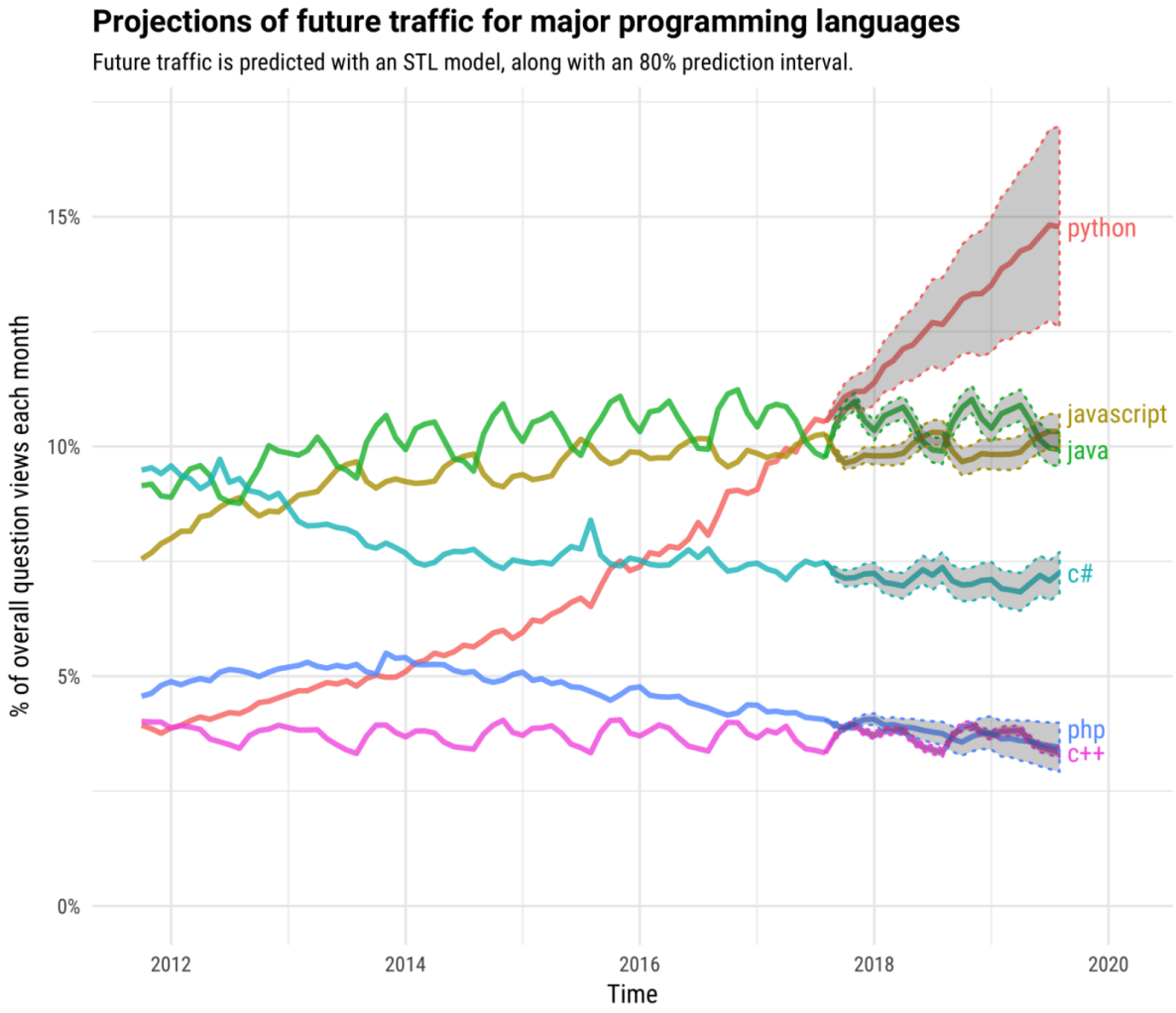
1) Learn Python for Research (GitHub)

2) Natural Language Processing (NLP) Tutorial (GitHub)

Agenda

1. Python eco-system
2. Using Python
3. Jupyter Notebook
4. Python syntax
5. Extra topics
    - Folder structure
    - GitHub crash-course
    - Begin-to-End example
    - General tips

Why Python?



(Source: https://stackoverflow.blog/2017/09/06/incredible-growth-python/)

Python 2 vs. Python 3:

- Simple: always use Python 3 unless you have to use Python 2.

Python 3 receives new updates, Python 2.7 is slowly phased out.

Note! Python 3 syntax is not always backwards compatible!

`print 'Hello, world!'` ▶ Only works in Python 2.7

`print('Hello, world!')` ▶ Works in Python 2.7 and Python 3.X

We will use Python 3.6

Modules and packages

A module/package is Python code that you "import" to add functionality.

Two types of modules/packages:

1. Build-in modules that are included with Python
2. Third-party modules/packages
   - The Python Package Index hosts more than 130,000 packages!

Example:

```
import os
```

- Standard module

```
import pandas as pd
```

- Third-party module

Modules and packages

How to install third-party modules/packages?

Use `pip` to install packages hosted on the Python Package Index

Use `conda` to install packages hosted by `Anaconda` or `Conda-Forge`

Modules and packages

How to install third-party modules/packages?

Use `pip` to install packages hosted on the Python Package Index

Use `conda` to install packages hosted by `Anaconda` or `Conda-Forge`

Recommendation:

- always start with the default Anaconda 3 distribution!

Anaconda: Python bundled with most used data science packages.

For more info: Anconda Distribution website

Why Python?

Python Eco-system

Using Python

How to run Python code?

1) Save code to `.py` file and run from command line: `python file.py`

```
file.py
1  print('Hello World!')
2
3
```

```
(base) C:\Users\kokti>python file.py
Hello World!
```

2) Use an interactive console in the command line: `python` or `ipython`

```
(base) C:\Users\kokti>ipython
Python 3.6.4 |Anaconda custom (64-bit)| (de
In [1]: print('Hello World!')
Hello World!
```

- **3) Use Jupyter Notebooks!**

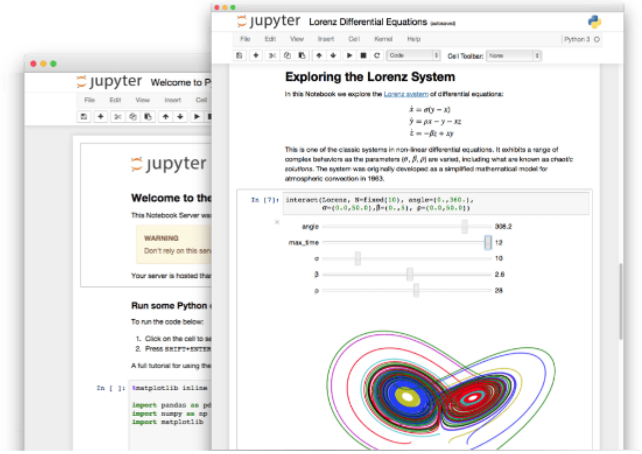
Why Python?

Python Eco-system

Using Python

Jupyter Notebook

Jupyter Notebook



The Jupyter Notebook

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.

Language of choice

The Notebook has support for over 40 programming languages, including Python, R, Julia, and Scala.

Share notebooks

Notebooks can be shared with others using email, Dropbox, GitHub and the Jupyter Notebook Viewer.

Interactive output

Your code can produce rich, interactive output: HTML, images, videos, LaTeX, and custom MIME types.

Big data integration

Leverage big data tools, such as Apache Spark, from Python, R and Scala. Explore that same data with pandas, scikit-learn, ggplot2, TensorFlow.

Try it in your browser

Why Python?

Python Eco-system

Using Python

Jupyter Notebook

Jupyter Notebook

localhost:8888/notebooks/Example_notebook.ipynb

jupyter Example_notebook (unsaved changes)

File Edit View Insert Cell Kernel Help Python 3

Code CellToolbar

In [1]:
```
import ipystata
from IPython.display import Image
```

In [ ]:
```
words = ['But', 'also', 'live', 'code']
for i in words:
    print(i)
```

```
##
```

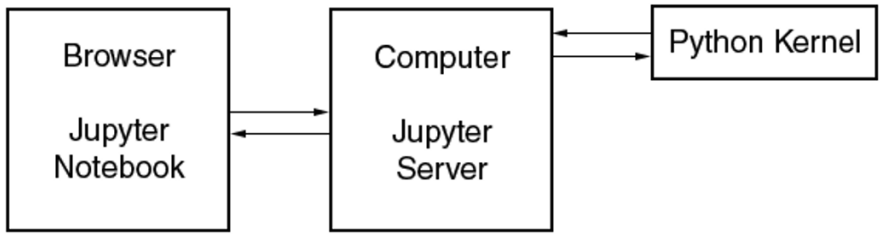
```
$$Scaled\ change\ in\ Earnings_t: \frac{(Earnings_t-Earnings_{t-1})}{MV_{t-2}}$$
```

```
##
```

In [ ]:
```
Image(r'Example/Images/table1a.png', width=700)
```

Jupyter Notebook

How does it work:



How to start:

1. Open up a command line / terminal
2. Change to project directory using `cd`
3. Type: `jupyter notebook`

How to stop:

- Press `ctrl+c` in command line / terminal

Jupyter Notebook

Using a Jupyter Notebook is largely self-explanatory.

Most relevant shortcuts for reference purposes:

`command mode` --> enable by pressing `esc`
`edit mode` --> enable by pressing `enter`

`command mode`	`edit mode`	`both modes`
`Y` : cell to code	`Tab` : code completion or indent	`Shift-Enter` : run cell, select below
`M` : cell to markdown	`Shift-Tab` : tooltip	`Ctrl-Enter` : run cell
`A` : insert cell above	`Ctrl-A` : select all	
`B` : insert cell below	`Ctrl-Z` : undo	
`X` : cut selected cell		

Why Python?

Python Eco-system

Using Python

Jupyter Notebook

Python syntax

Python syntax is easy!

```
numbers = [11, 5, 20, 6]
response = 'The following number is too big:'

for i in numbers:
        if i > 10:
                print(response)
                print(i)
```

Why Python?

Python Eco-system

Using Python

Jupyter Notebook

Python syntax

Python syntax is easy!

```
numbers = [11, 5, 20, 6]
response = 'The following number is too big:'

for i in numbers:
        if i > 10:
                print(response)
                print(i)
```

Where to start?

- I recommend to use my Python Basics Notebook

A couple of caveats

1) It is best practice to include all imports at the start

2) The spacing (i.e. tabs) are not just for looks!

```
for i in numbers:
        if i > 10:
                print(response)
                print(i)
```

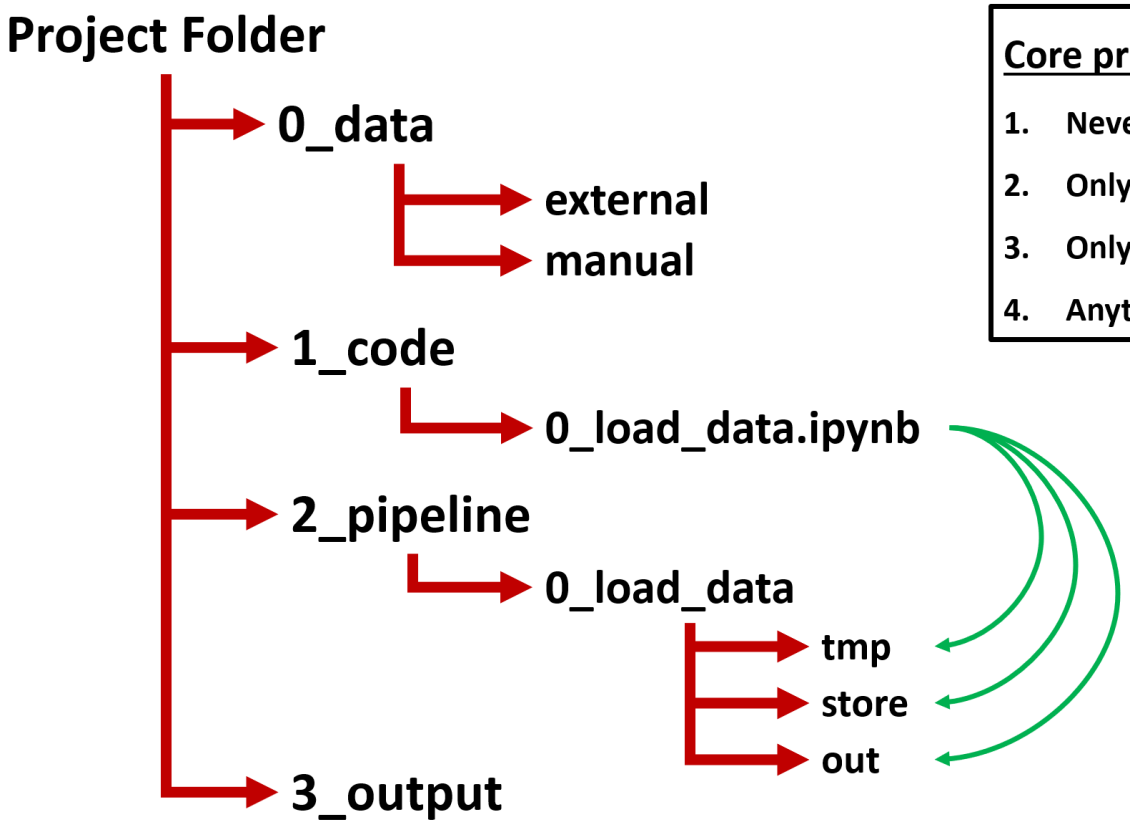
3) Avoid "blind" `try` and `except` blocks.

```
try:
    num_list.remove(4)
except ValueError as e:
    print('Number not in the list')
```

Good to be specific!

Folder Structure

Folder structure



Core principles:

1. **Never modify** 0_data
2. **Only save to** pipeline **folder**
3. **Only load from** 0_data **or** out
4. **Anything in** tmp **can be deleted**

GitHub Repository

You should really look into using version control with Git + GitHub.

GitHub provides **free** private repositories for academics:

- Apply for GitHub Education

Steps to get your project on GitHub:

1. Create a new empty repository on GitHub.com
2. Clone empty repository using GitHub Desktop app to your computer
3. Copy your project files into this folder
4. Commit + Push using GitHub Desktop app

- The earlier you do this the better!

Project Begin-to-End

My usual workflow for a research project:

1. Create empty repository on GitHub + setup folder structure
   - can save a lot of headache later on!
2. Start with Python to gather and clean data
   - usually 70% of the work
3. Once the data is ready, I switch over to R or Stata
   - but still in the Jupyter Notebook!
4. Write the paper and create the tables in LaTeX
   - I highly recommend ShareLaTeX for LaTeX!

Questions?

Closing remarks

Setup + Get Started

Setup:

1. Make sure you have Anaconda installed
2. Make sure you can start / open a Jupyter Notebook

Get Started:

Goal: Solve some of the "Basic Python Tasks" in a Jupyter Notebook.

1. Open a Jupyter Notebook in the `Materials` folder
2. Solve some of the "Basic Python Tasks"
   - Find them in `Materials > Day_1 > mini_task`

For help:

- Python tutorial
- Python Basics Notebook