Introduction to Programming

For Archaeologists

Part 3: Loading and manipulating data

2021-2022

Universiteit Leiden
The Netherlands

Discover the world at Leiden University

Topics of this lecture series

1. Introduction: Python, variables, comments
2. Lists & Loops
3. **Loading and manipulating data**
4. Graphs & Plots
5. SQL & Databases
6. Advanced methods: Machine Learning, QGIS integration

Assignment

Assignment deadlines

- Assignment 1: 22 April
- Assignment 2: **6 May**
- Assignment 3: 20 May

Assignment 2 can now be found on github!

Topics of this lecture

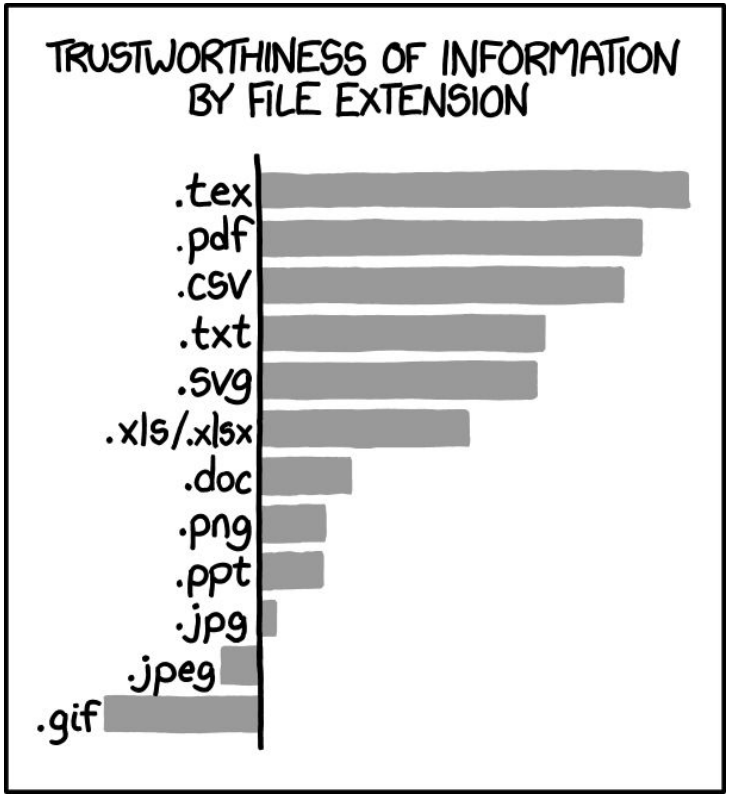
- CSV format for spreadsheets
- Importing and using libraries
- Paths (or how to get to files on your laptop)
- Loading data into Python
- Pandas DataFrames

After this lecture:

- You know what a CSV file is, how to create one, and what they look like
- You can import and use some libraries in Python
- You can load spreadsheet data into Python
- You know what a DataFrame is
- You can do some basic editing and analyses on DataFrames

Comma Separated Values (CSV)

- Very basic spreadsheet format
- Really just a .txt file!
- Basic = easy to read / archive
- Export from Excel



CSV format - test.csv example

```
width,height,weight
2,3,5
5,9,8
```

Importing libraries

- Python can’t read CSV files by default
- We need to import a **library** that can do this
- A library is basically a collection of functions other people wrote
- When you import a library, you get access to the functions
- Looks like this:

```
import os
print(os.getcwd())
```

Output: `'C:\\Users\Alex\Scripts\'`

Installing libraries

- Your Anaconda installation has a bunch of libraries already installed by default
- If you want to use one that's not installed, you can install it using the `pip` command
- Normally you would have to install this in the command line, but Jupyter Notebook lets you run commands using '!'

```
!pip install sklearn
import sklearn
```

Opening files in Python

- Use the `open()` function, which opens the file
- Then use `readlines()` to create a list of strings

```
with open('test.txt') as file:
    lines = file.readlines()

print(lines)
```

Output: `['line 1', 'line 2', 'line 3']`

Opening CSV in Python

- We can also do this with CSV files

```
with open('test.csv') as file:
    lines = file.readlines()

print(lines)
```

Output: `['width,height,weight', '2,3,5', '5,9,8']`

Not very handy, need to manually convert all this to lists/dicts

Loading CSV

- First import the csv module, then use the csv.reader()
- Loop through rows

```
import csv
with open('test.csv') as file:
    rows = csv.reader(file)
    for row in rows:
        print(row)
```

Loading CSV

Output:

```
['width', 'height', 'weight']
[2, 3, 5]
[5, 9, 8]
```

- A lot handier: `csv` module has taken care of splitting the text into a list
- Useful if you want to do something for each row (lookup url, combine fields, etc)

Pandas DataFrames

- Reading with csv can be useful, but most of the time you want to work with the data like a spreadsheet
- We can use DataFrames, which are Python's way of doing spreadsheet things

```
import pandas as pd
test = pd.read_csv('test.csv')
print(test)
```

DataFrame output

```
import pandas as pd
df = pd.read_csv('test.csv')
print(df)
```

Output:

```
   width  height  weight
0      2       3       5
1      5       9       8
```

DataFrame exploration

- `df` , shows whole dataframe, no print needed!
- `df.head(n)`, shows the first *n* rows
- `df.tail(n)`, shows the last *n* rows
- `df.sample(n)`, shows a random sample of *n* rows
- `df.info()`, shows information about each column
- `df.describe()`, shows descriptive statistics (mean, max, etc)

DataFrame filters / subsets

- Useful if we want to select a smaller subset of data for analysis or plotting

```
# select just 1 column
weights = df['weight']


# select rows with specific value in 1 of the cols
heavy_artefacts = df[ df['weight'] > 10 ]
```

Saving DataFrames

- Once we're done editing a dataframe, we can save it back to csv

```
df.to_csv("test.csv")
```

After this lecture:

- You know what a CSV file is, how to create one, and what they look like
- You can import and use some libraries in Python
- You can load spreadsheet data into Python
- You know what a DataFrame is
- You can do some basic editing and analyses on DataFrames

Questions?

- **Any questions about any of the subjects?**

- Contact me at
  - a.brandsen@arch.leidenuniv.nl

Slides are available on Brightspace

Exercises

github.com/alexbrandsen/Introduction-to-Programming-for-Archaeologists

- Go to github
- Click on ‘modules’
- Right click on the third module
- Select ‘save link as’ or ‘download as’
- Save the file in the ‘modules’ folder within your own Scripts folder
- Start Anaconda
- Start Jupyter Notebook
- Navigate to the notebook file and run it