Python Programming and Machine Learning for Economists (Jan/Feb 2022)

Michael E. Rose, PhD

Introduction

Who am I?

- Senior Research Fellow, Max Planck Institute for Innovation and Competition, PhD in Econ (University of Cape Town)
- Writing code since 8th grade
- Author of 3 open-source projects: `pybliometrics`, `sosia`, `scholarmetrics`
- Teaching experience:
  - *This course* @ Kiel Institute for the World Economy (ASP), University of Zurich, ifo Institute Munich, LMU Munich, Scheller College of Business at Georgia Tech, TU Munich
  - Risk Management Computing Skills [Matlab, SQL, Excel, VBA] @ University of Cape Town
- `Michael.Ernst.Rose@gmail.com`

Who are you?

- Name, Status
- Which languages, how long?
- Which operating system?
- Who is more in control, your computer or you?

Course content

1. Empirical research using Python
2. Project management
3. Supervised Machine Learning
4. Unsupervised Machine Learning
5. Natural Language Processing

Course Design

- Lecture in the morning, exercises in the afternoon
- Each exercise session starts with a Monty Python sketch
- 10 Minutes breaks after 50 Minutes of Teaching

Exercises (= mini projects)

- Difficulty increases as the course progresses

Data sets
in tutorials

Data sets in
the wild

- Your grades depend on the final exercises

Learning outcomes

- Programming part
  1. List some of the right basic tools for empirical researc
  2. Use python independently
  3. Apply `pandas`, `seaborn`, `sklearn`
  4. Understand coding principles
  5. Use PyCharm
  6. Understand version control and use git
- Machine Learning
  1. Apply simple Neural Networks, clustering algorithms and Principal Component Analysis
  2. Interpret and evaluate any machine learning application
  3. Teach yourself how to apply machine learning algorithms we don't speak about

Required Readings

- Shapiro, J. and M. Gentzkow: “Code and Data for the Social Sciences: A Practitioners Guide” - *Short paper on project management by Economists, read it all today*
- Athey, S. and G. Imbens (ARE 2019): “Machine Learning Methods That Economists Should Know About” - *Well-written overview that introduces all the technical terms for meachine learning, read it until 3rd day*
- Gentzkow, M., B. Kelly and M. Taddy (JEL 2019): “Text as Data" - *Well-written introduction to language processing, read it until last day*

How to use Python

Why Python?

- Interpreted, high-level, general-purpose programming language
- Can be object-oriented, imperative, functional and procedural
- Free (= no licenses)
- Large (= support and many packages)
- Centralized development
- Very good first language

Why Python?

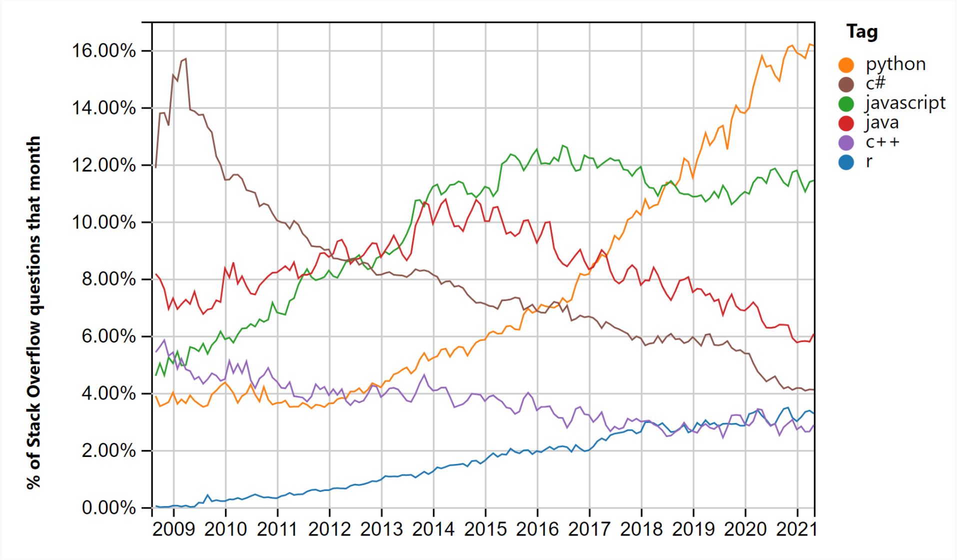
- Interpreted, high-level, general-purpose programming language
- Can be object-oriented, imperative, functional and procedural
- Free (= no licenses)
- Large (= support and many packages)
- Centralized development
- Very good first language

There should be one– and preferably only one –obvious way to do it. Although that way may not be obvious at first unless you're Dutch. (Tim Peters - The Zen of Python)

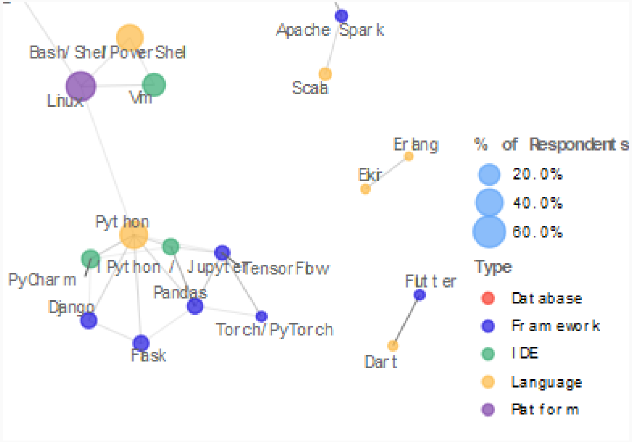
Credit where Credit is due

- Guido van Rossum
  created Python in his Christmas holidays 1989 as
  "a descendant of ABC that would appeal to Unix/C hackers. I chose Python as a working title for the project, being in a slightly irreverent mood (and a big fan of Monty Python's Flying Circus)."
- Since 2019 5-member steering committee at the Python Foundation heads the development of Python

Python is popular and increasing in popularity



Python's local technology cluster



StackOverflow.com: "Developer Survey Results 2019"

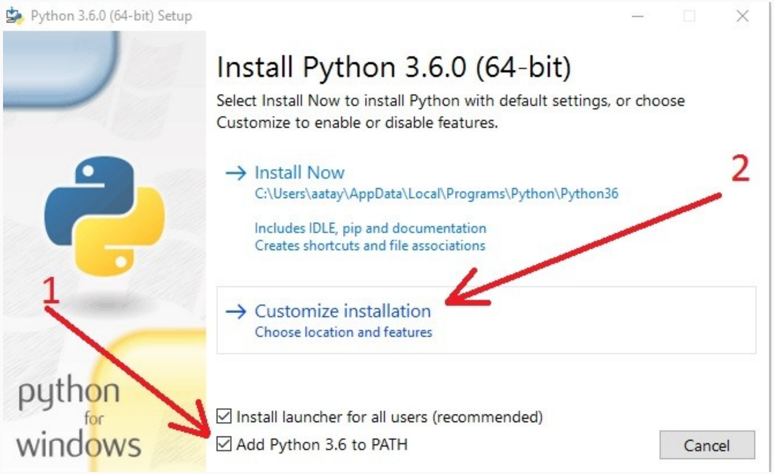
Why I discourage anaconda

- packages provided by anaconda need to be installed with `conda install` (they will ONLY be in the conda environment)
- packages tend to be outdated
- Overkill/Unnecessary software
- Jupyter and spyder run without anaconda as well
- Actually not *that* popular: 19% of Python installations via Anaconda[1]

[1] Python Developers Survey 2020 Results

Installing Python and pip

https://www.python.org/downloads/



Python 3.6.0 (64-bit) Setup

Install Python 3.6.0 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

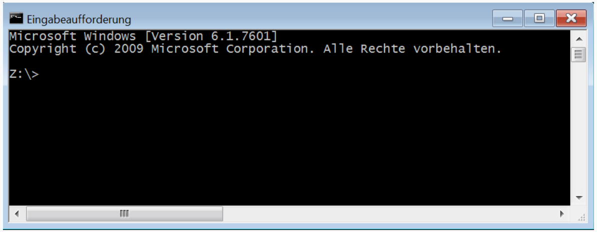
→ Install Now
C:\Users\aatay\AppData\Local\Programs\Python\Python36

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

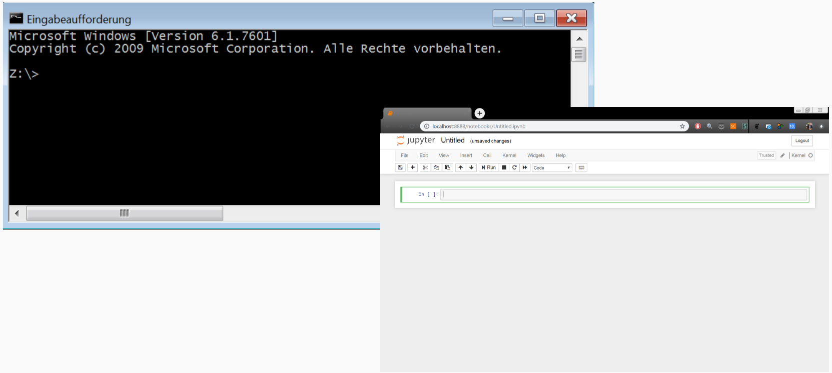
☑ Install launcher for all users (recommended)
☑ Add Python 3.6 to PATH

Different ways to use Python



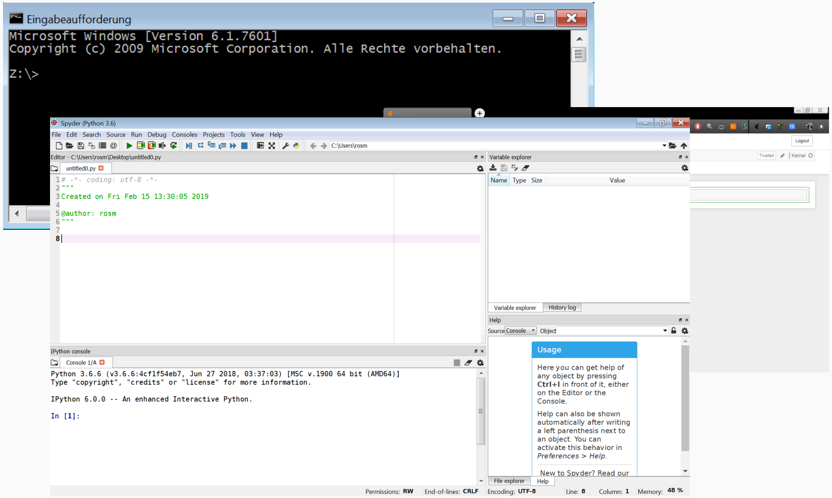
```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Alle Rechte vorbehalten.

Z:\>
```

Different ways to use Python


Eingabeaufforderung
localhost:8888/notebooks/Untitled.ipynb
jupyter Untitled (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Kernel
Run
Code
In [ ]:

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Alle Rechte vorbehalten.

Z:\>
```

Different ways to use Python


Eingabeaufforderung
Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\rosm
Editor - C:\Users\rosm\Desktop\untitled0.py
untitled0.py
Variable explorer
Name Type Size Value
Variable explorer History log
Help
Source Console Object
Usage
IPython console
Console 1/A
File explorer Help
Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 8 Column: 1 Memory: 48 %
Logout
Trusted
Kernel

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Alle Rechte vorbehalten.

Z:\>
```

```python
# -*- coding: utf-8 -*-
"""
Created on Fri Feb 15 13:30:05 2019

@author: rosm
"""

```

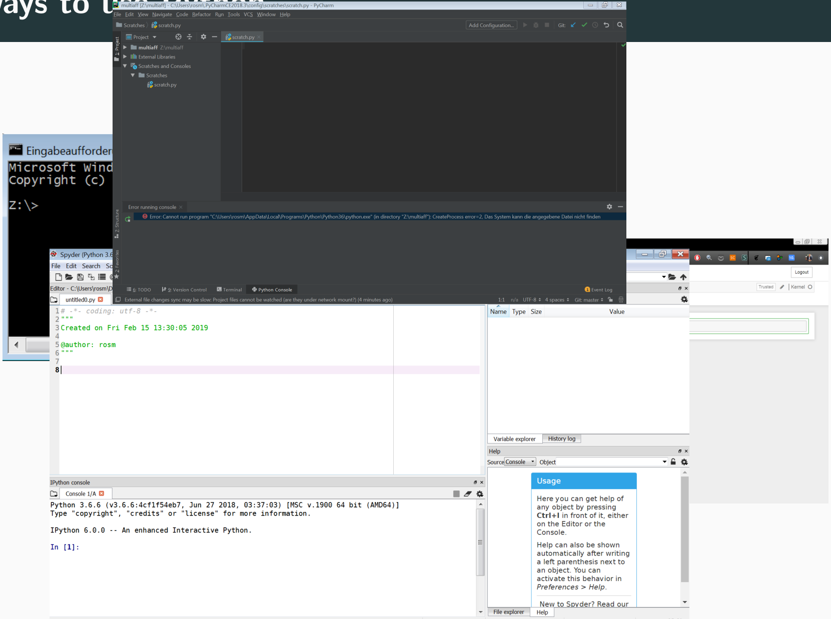
```
Python 3.6.6 (v3.6.6:4cf1f54eb7, Jun 27 2018, 03:37:03) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 6.0.0 -- An enhanced Interactive Python.

In [1]:
```

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our

Different ways to use Python



```
Microsoft Wind
Copyright (c)

Z:\>
```

multiaff [Z:\multiaff] - C:\Users\rosm\.PyCharmCE2018.3\config\scratches\scratch.py - PyCharm

```
Error: Cannot run program "C:\Users\rosm\AppData\Local\Programs\Python\Python36\python.exe" (in directory "Z:\multiaff"): CreateProcess error=2, Das System kann die angegebene Datei nicht finden
```

```
External file changes sync may be slow: Project files cannot be watched (are they under network mount?) (4 minutes ago)
```

Editor - C:\Users\rosm\D

```python
# -*- coding: utf-8 -*-
"""
Created on Fri Feb 15 13:30:05 2019

@author: rosm
"""

```

```
Python 3.6.6 (v3.6.6:4cf1f54eb7, Jun 27 2018, 03:37:03) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 6.0.0 -- An enhanced Interactive Python.

In [1]:
```

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 8 Column: 1 Memory: 48 %

Terminal/Console

- Console uses DOS language () or `shell` and `bash` (and)
- Starts python environment, Jupyter, and executes scripts

Terminal/Console

- Console uses DOS language ( ) or `shell` and `bash` (and)
- Starts python environment, Jupyter, and executes scripts
- Install packages here:

```
python -m pip install pandas seaborn
```

```
python3 -m pip install pandas seaborn
```

- Shortcut (which is not platform-independent)

```
pip install pandas seaborn
```

```
pip3 install pandas seaborn
```

Jupyter Notebook on your computer

- Create a folder for this course and navigate there in your terminal (alternatively, open the "PowerShell" via context menu after ⇧+rightclick)

Jupyter Notebook on your computer

- Create a folder for this course and navigate there in your terminal (alternatively, open the "PowerShell" via context menu after ⇧+rightclick)
- Install the jupyter notebook if necessary

```
python3 -m pip install notebook
jupyter notebook
```

- Your browser will fire up (i.e., you started your own server)

Jupyter Notebook on your computer

- Create a folder for this course and navigate there in your terminal (alternatively, open the "PowerShell" via context menu after ⇧+rightclick)
- Install the jupyter notebook if necessary

```
python3 -m pip install notebook
jupyter notebook
```

- Your browser will fire up (i.e., you started your own server)
- Click on New in the upper right corner to start a new notebook

Notebooks will be saved in the folder where you invoked the jupyter server

Jupyter notebook in the ☁

- colab.research.google.com: requires Google account; stores notebooks in your Drive; integrates with GitHub; potentially older packages
- kaggle.com/code: requires Kaggle account; allows for R as well
- mybinder.org: requires GitHub account; builds from a GitHub repository

Recap some Python basics

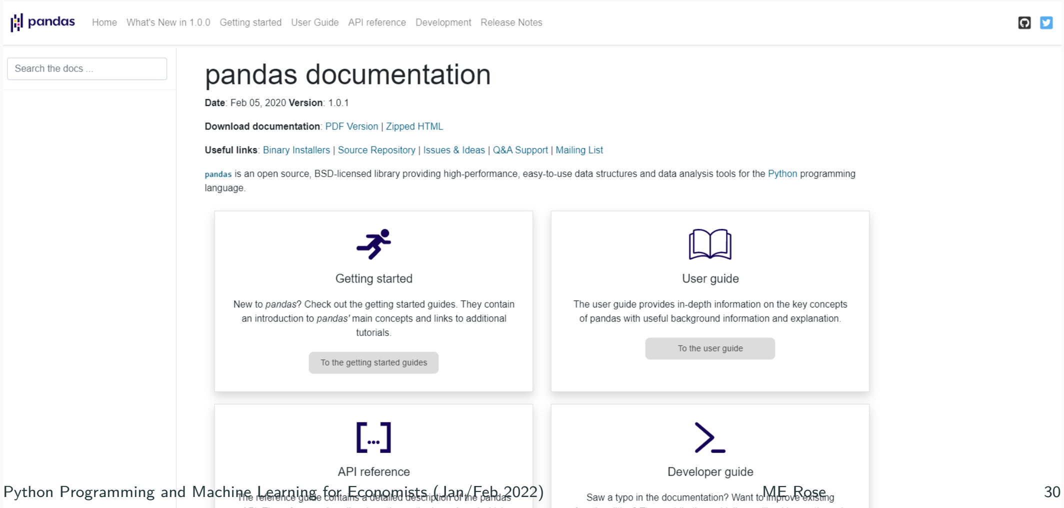
What matters in Python?

- Indentation is key (convention: four spaces)
- Case-sensitive
- Variables must not start with numbers
- It's a language, *not* a program

Pandas

pandas: the library for data manipulation

- Documentation: http://pandas.pydata.org/pandas-docs/stable/



pandas Home What's New in 1.0.0 Getting started User Guide API reference Development Release Notes

Search the docs ...

pandas documentation

Date: Feb 05, 2020 **Version**: 1.0.1

Download documentation: PDF Version | Zipped HTML

Useful links: Binary Installers | Source Repository | Issues & Ideas | Q&A Support | Mailing List

`pandas` is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

Getting started

New to *pandas*? Check out the getting started guides. They contain an introduction to *pandas'* main concepts and links to additional tutorials.

To the getting started guides

User guide

The user guide provides in-depth information on the key concepts of pandas with useful background information and explanation.

To the user guide

API reference

The reference guide contains a detailed description of the pandas

Developer guide

Saw a typo in the documentation? Want to improve existing

Let’s start with a dataset on twins...

```python
import pandas as pd

FNAME = "http://www.stat.ucla.edu/~rgould/datasets/twins.dat"

df = pd.read_csv(FNAME, sep='\t')
```

- Documentation at http://www.stat.ucla.edu/~rgould/datasets/twinsexplain.txt

pandas functionality relevant for the course

- 10 minutes to pandas
- IO tools (text, CSV, HDF5, ...)
- Indexing and selecting data
- Reshaping and pivot tables
- Working with missing data
- Computational tools

Let’s inspect our data

```python
df.shape  # Dimensions
df.head()  # First 5 lines (by default)
df.tail(7)  # Last 7 lines
df.columns  # List of variables
df.describe()  # Summary statistics
```

1. How many observations do you have?
2. How many variables do you have?
3. Which variables are numeric?
4. What is the mean of variable `"DEDUC1"`?

Slicing the DataFrame

```python
# Selecting columns
df["DEDUC1"]  # Column by column name
df[["AGE", "LHRWAGEH"]]  # Columns by list of column names
df.iloc[:, 5:7]  # Column range by column indices

# Selecting rows
df.loc[0]  # Row by index name (also accepts lists)
df.iloc[0]  # Row by row number (also accepts lists)

# Selecting values
df.loc[18, "AGE"]  # Name of row and column
df.iloc[18, 2]  # Index of row and column
```

Slicing the DataFrame

```python
# Selecting columns
df["DEDUC1"]  # Column by column name
df[["AGE", "LHRWAGEH"]]  # Columns by list of column names
df.iloc[:, 5:7]  # Column range by column indices

# Selecting rows
df.loc[0]  # Row by index name (also accepts lists)
df.iloc[0]  # Row by row number (also accepts lists)

# Selecting values
df.loc[18, "AGE"]  # Name of row and column
df.iloc[18, 2]  # Index of row and column
```

1. What is the 6th entry of the 5th column?
2. What is the 5th entry of column `"DTEN"`?
3. What is the last entry of column `"LHRWAGEL"`?

Understanding dtypes

```
df.info()
```

Understanding dtypes

```python
df.info()
```

Pandas	Python	Purpose
object	unicode	Text
int64	int	Integers
float64	float	Floating numbers
bool	bool	True & False values
datetime64		Date and time values
timedelta[ns]		Differences between two datetimes
category		Finite list of text values

Changing dtypes

```python
df["WHITEH"] = df["WHITEH"].astype(bool)
df["DMARRIED"] = df["DMARRIED"].astype("category")
df["LHRWAGEH"] = pd.to_numeric(df["LHRWAGEH"], errors="coerce")
```

Optimising dtypes

```python
df.info(memory_usage=True)
```

Optimising dtypes

```
df.info(memory_usage=True)
```

```
bools = ['WHITEH', 'MALEH', 'WHITEL', 'MALEL']
df[bools] = df[bools].astype(bool)
df['DMARRIED'] = df['DMARRIED'].astype('int8')
df.info(memory_usage=True)
```

Boolean indexing

```python
df[df["AGE"] > 20]  # One condition
df[(df["AGE"] > 20) & (df["WHITEL"] == 1)]  # Multiple conditions
df[~(df["AGE"] > 20)]  # Tilde inverses boolean
values = (20, 21, 22, 23)
df[df["AGE"].isin(values)]  # Select specific values
```

Boolean indexing

```python
df[df["AGE"] > 20]  # One condition
df[(df["AGE"] > 20) & (df["WHITEL"] == 1)]  # Multiple conditions
df[~(df["AGE"] > 20)]  # Tilde inverses boolean
values = (20, 21, 22, 23)
df[df["AGE"].isin(values)]  # Select specific values
```

1. How many observations have `"WHITEL"` equal to 0?
2. How many observations have `"WHITEH"` equal to 1 and `"DEDUC1` unequal to 0?
3. In how many rows do the values for `"WHITEH"` and `"WHITEL"` differ?
4. What is the mean age of twins whose L-sibling is a non-white male with either 12 or 14 years of education? (Use `"WHITEL"`, `"MALEL"` and `"EDUCHL"`,)

Aggregate data

```
df["WHITEL"].value_counts()
pd.crosstab(df["WHITEH"], df["WHITEL"])
```

Aggregate data

```python
df["WHITEL"].value_counts()
pd.crosstab(df["WHITEH"], df["WHITEL"])
```

1. What is the most common value in `"EDUCL"`?
2. What is the most common combination of `"MALEH"` and `"MALEL"`?

Manipulation

```python
# Representation
df = df.sort_values(by='HRWAGEH')  # Sorting by column
df = df[sorted(df.columns)]  # Re-order columns alphabetically
# Work on columns
df = df.drop('AGESQ', axis=1)  # Drop a column
df['new'] = 9  # Add new column
df['AGETR'] = df['AGE']**3
df['combined'] = df['MALEH'] + df['EDUCH']
# Missing data
df["HRWAGEH_new"] = df["HRWAGEH"].fillna(0)  # Fill missings with 0
df = df.dropna(subset=["HRWAGEH"])  # Drop rows missing in "HRWAGEH"
```

Grouping

```python
grouped = df.groupby(['MALEH'])
print(grouped['AGE'].mean())
print(grouped['EDUCH'].agg(['mean', 'sum']))
print(grouped[['EDUCH', 'AGE']].agg(['mean', 'std']))
```

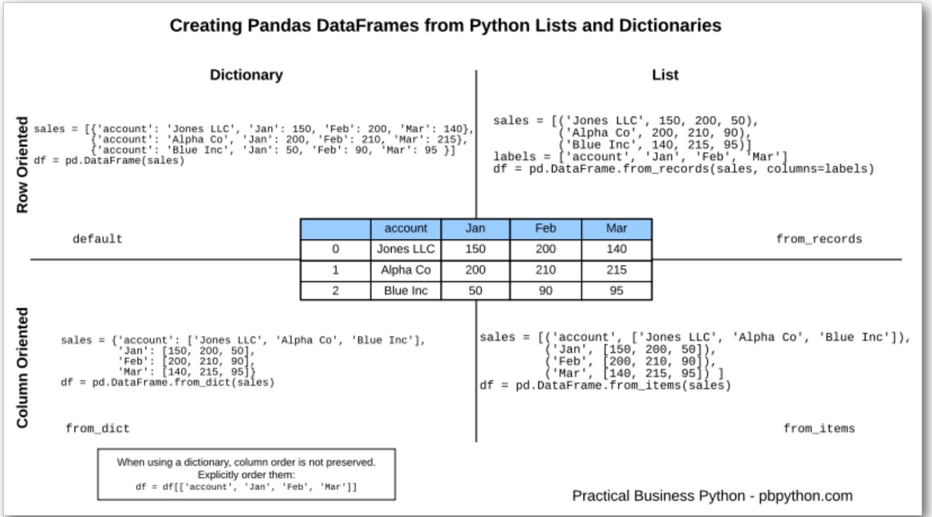
Grouping

```python
grouped = df.groupby(['MALEH'])
print(grouped['AGE'].mean())
print(grouped['EDUCH'].agg(['mean', 'sum']))
print(grouped[['EDUCH', 'AGE']].agg(['mean', 'std']))
```

→ Full list at https://pandas.pydata.org/pandas-docs/stable/user_guide/groupby.html#aggregation

- What is the `"AGE"` variance for `"MALEL" == 0` individuals?
- What are the second and the third quartile of years of schooling for female L-siblings? (Use `"EUDCL"` and `"MALEL" == 0`)
- What is the average `"AGE"` for twins where both siblings are female?

Creating DataFrames from other objects



Creating Pandas DataFrames from Python Lists and Dictionaries

```
sales = [{'account': 'Jones LLC', 'Jan': 150, 'Feb': 200, 'Mar': 140},
         {'account': 'Alpha Co',  'Jan': 200, 'Feb': 210, 'Mar': 215},
         {'account': 'Blue Inc',  'Jan': 50,  'Feb': 90,  'Mar': 95 }]
df = pd.DataFrame(sales)
```

```
sales = [('Jones LLC', 150, 200, 50),
         ('Alpha Co', 200, 210, 90),
         ('Blue Inc', 140, 215, 95)]
labels = ['account', 'Jan', 'Feb', 'Mar']
df = pd.DataFrame.from_records(sales, columns=labels)
```

	account	Jan	Feb	Mar
0	Jones LLC	150	200	140
1	Alpha Co	200	210	215
2	Blue Inc	50	90	95

```
sales = {'account': ['Jones LLC', 'Alpha Co', 'Blue Inc'],
         'Jan': [150, 200, 50],
         'Feb': [200, 210, 90],
         'Mar': [140, 215, 95]}
df = pd.DataFrame.from_dict(sales)
```

```
sales = [('account', ['Jones LLC', 'Alpha Co', 'Blue Inc']),
         ('Jan', [150, 200, 50]),
         ('Feb', [200, 210, 90]),
         ('Mar', [140, 215, 95]) ]
df = pd.DataFrame.from_items(sales)
```

When using a dictionary, column order is not preserved.
Explicitly order them:
`df = df[['account', 'Jan', 'Feb', 'Mar']]`

Practical Business Python - pbpython.com

To become a Master...

- 10 minutes to pandas
- Wes McKinney: "Python for Data Analysis. Data Wrangling with Pandas, NumPy, and IPython", O'Reilly (2017)
- Fabio Nelli: "Python Data Analytics. Data Analysis and Science Using Pandas, matplotlib, and the Python Programming Language", Apress (2015)

Plotting w/ pandas (matplotlib), and w/ seaborn

matplotlib

pandas

seaborn

Visualization with pandas

- Straightforward plotting as DataFrame methods for all kinds: barplots, areas, histograms, violin plots, timeseries, etc.: `https://pandas.pydata.org/pandas-docs/stable/visualization.html`
- Has `matplotlib` under the hood - for aesthetics
  ```python
  import matplotlib.pyplot as plt
  ```
- Set global styles with `plt.style.use('<style>')` (list all styles with `plt.style.available`)

! Beware: Have DataFrame in correct format (long vs. wide)

Statistical plotting with seaborn

- seaborn: wrapper for `matplotlib`, optimized for quick statistical plotting: Error bars, distributions, regressions, etc.

- Use seaborn's toy datasets using `.load_dataset()`

☞ If downloading example datasets via `.load_dataset()` doesn't work, get them from github.com/mwaskom/seaborn-data and store them in `~./seaborn-data/`

Seaborn's plotting philosophy

- Statistical relation between numeric values?
  - ➔ `relplot()` for Scatter and Line (→ Documentation)
- Categorical data?
  - ➔ `catplot()` for Scatter-like (Swarm and Strip), Distributions (Box, Violin, Boxen) and Estimations (Point, Bar, Count) (→ Documentation)
- Linear relationships?
  - ➔ `regplot()` (→ Documentation)

Pandas plotting vs. seaborn

- In Jupyter, remember to write and execute `%matplotlib inline` in first cell to show figures
- Use pandas when you do the aggregations yourself
- Use seaborn when you use raw data – seaborn will aggregate itself

Excourse: colormaps

Color maps

List of named colors

To become a Master...

- Fabio Nelli: "Python Data Analytics. Data Analysis and Science Using Pandas, matplotlib, and the Python Programming Language", Apress (2015)
- matplotlib Tutorials
- seaborn User guide and tutorial