University of Zurich^UZH

Department of Physics

Python in the Wild

Scientific Programming with Python

Nicola Chiapolini, Andreas Weiden, Christian Elsasser & Federica Lionetto

Agenda

We want to give a couple of concrete examples from our work how python is used to foster ideas and also shed light on additional functionalities not covered so far in the course

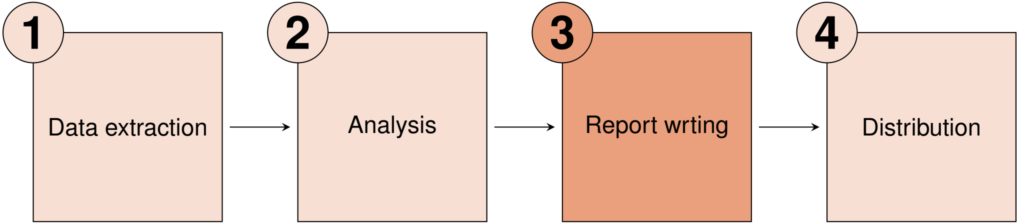
- Python to automate reports
- Python to automate testing a webpage
- Python to interpret ML decisions
- Python to design efficient class hierarchies

Python to automate reports

Let’s meet Konstantin-Maximilian – Core skill: slide writing

Python can easily do what Konstantin-Maximilian can do!

Process chain



1 Data extraction

2 Analysis

3 Report wrting

4 Distribution

Suitable libraries

1. `requests`, `BeautifulSoup` and particular modules for certain data repositories (*e.g.* Quandl)
2. What ever you want to do (if geo-data is a topic, have a look at `geopandas`)
3. `python-pptx`, `Pandas`
4. `smtplib`, `email`, `pywin32`

Mimicking of a Web user

Automated Testing of a Web-Application

Problem

- PhD progress tracked via Web-Application
- many wishes for improvments and extensions
- we need tests

Solution

- Selenium (tool to control web-browsers)
- controled with python-unittests

Interpretation of Machine-Learning Decisions

Interpretation of ML decisions - 1/1

Problem

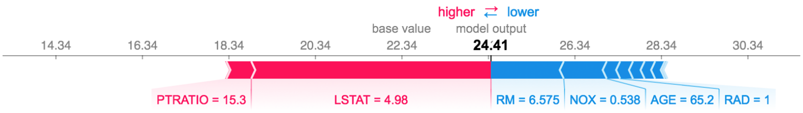
- After months of hard work, you developed a very cool ML algorithm, which is able to predict whether a new startup will be successful
- You are ready to make money with it, but why should your potential customers trust your “black box”?

Solution

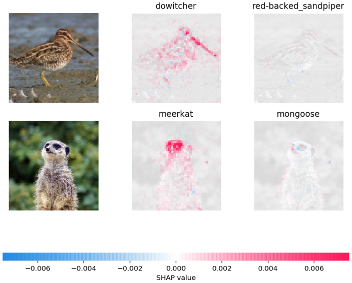
- SHAP: SHapley Additive exPlanations (Lundberg and Lee, https://pypi.org/project/shap/)
- You make your ML algorithm understandable by explaining which features (and how) are affecting each decision
  - for the whole sample as well as for individual observations

Interpretation of ML decisions - 2/2

tree ensemble ⟹



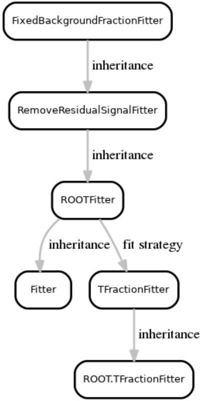
deep learning ⟹



Class Hierarchies in Real World

Class hierarchy for fitting

- Application of OOP
- Each subclass has a `fit` method that is overriden. It re-uses the method of the parent class
- Strategy pattern for the actual fitter underneath (`ROOT.TFractionFitter`)
- Uses `argparse` to take the name of the fitting class as argument


FixedBackgroundFractionFitter
inheritance
RemoveResidualSignalFitter
inheritance
ROOTFitter
inheritance
fit strategy
Fitter
TFractionFitter
inheritance
ROOT.TFractionFitter