Python for Data Science

PyCon Finland 17.10.2011
Harri Hämäläinen
harri.hamalainen aalto fi
#sgwwx

What is Data Science

- Data science ~ computer science + mathematics/statistics + visualization
- Most web companies are actually doing some kind of Data Science:
    - Facebook, Amazon, Google, LinkedIn, Last.fm...
    - Social network analysis, recommendations, community building, decision making, analyzing emerging trends, ...

Scope of the talk

- What is in this presentation
  - Pythonized tools for retrieving and dealing with data
  - Methods, libraries, ethics
- What is not included
  - Dealing with very large data
  - Math or detailed algorithms behind library calls

Outline

- Harvesting
- Cleaning
- Analyzing
- Visualizing
- Publishing

Data

Data harvesting

Authorative borders for data sources

1. Data from your system
   - e.g. user access log files, purchase history, view counts
   - e.g. sensor readings, manual gathering
2. Data from the services of others
   - e.g. view counts, tweets, housing prizes, sport results, financing data

Data sources

- Locally available data
- Data dumps from Web
- Data through Web APIs
- Structured data in Web documents

API

- Available for many web applications accessible with general Python libraries
  - urllib, soaplib, suds, ...
- Some APIs available even as application specific Python libraries
  - python-twitter, python-linkedin, pyfacebook, ...

Data sources

- Locally available data
- Data dumps in Web
- Data through Web APIs
- Structured data in Web documents

Crawling

- Crawlers (spiders, web robots) are used to autonomously navigate the Web documents

```
import urllib
# queue holds urls, managed by some other component
def crawler(queue):
    url = queue.get()
    fd = urllib.urlopen(url)
    content = fd.read()
    links = parse_links(content)
    # process content
    for link in links:
        queue.put(link)
```

Web Scraping

- Extract information from structured documents in Web
- Multiple libraries for parsing XML documents
- But in general web documents are rarely valid XML
- Some candidates who will stand by you when data contains “dragons”
  - BeautifulSoup
  - lxml

urllib & BeautifulSoup

```
>>> import urllib, BeautifulSoup
>>> fd = urllib.urlopen('http://mobile.example.org/
foo.html')
>>> soup = BeautifulSoup.BeautifulSoup(fd.read())
>>> print soup.prettify()
...
<table>
    <tr><td>23</td></tr>
    <tr><td>24</td></tr>
    <tr><td>25</td></tr>
    <tr><td>22</td></tr>
    <tr><td>23</td></tr>
</table>
...
```

```
>>> values = [elem.text for elem in soup.find('table').findChildren('td')]
>>>> values
[u'23', u'24', u'25', u'22', u'23']
```

Scrapy

- A framework for crawling web sites and extracting structured data
- Features
  - extracts elements from XML/HTML (with XPath)
  - makes it easy to define crawlers with support more specific needs (e.g. HTTP compression headers, robots.txt, crawling depth)
  - real-time debugging
- http://scrapy.org

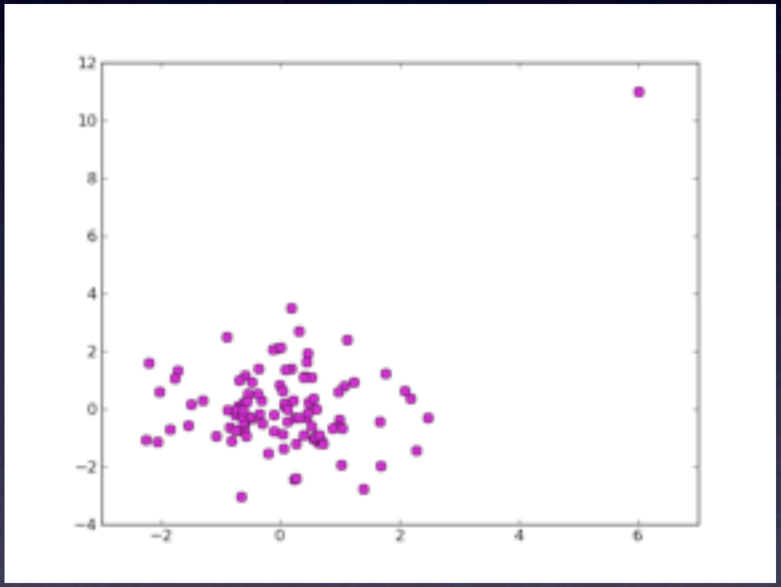
Tips and ethics

- Use the mobile version of the sites if available
- No cookies
- Respect robots.txt
- Identify yourself
- Use compression (RFC 2616)
- If possible, download bulk data first, process it later
- Prefer dumps over APIs, APIs over scraping
- Be polite and request permission to gather the data
- Worth checking: https://scraperwiki.com/

Data cleansing
(preprocessing)

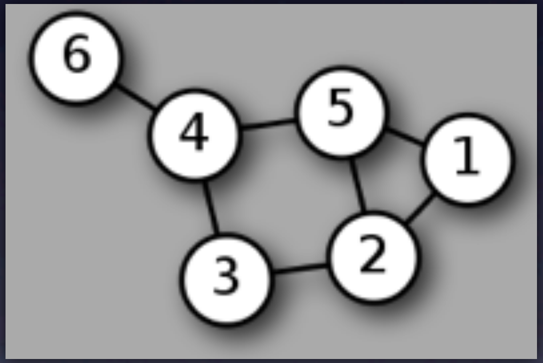
Data cleansing

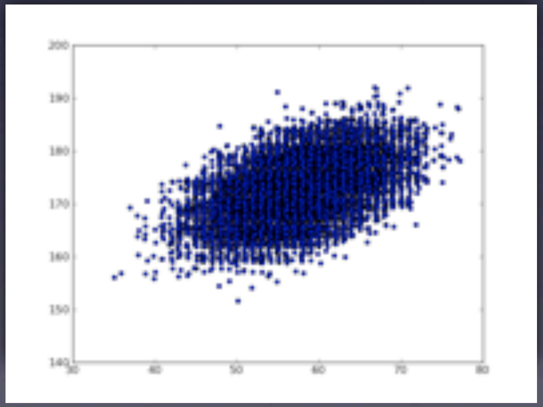
- Harvested data may come with lots of noise
- ... or interesting anomalies?
- Detection
  - Scatter plots
  - Statistical functions describing distribution



Data preprocessing

- Goal: provide structured presentation for analysis
  - Network (graph)
  - Values with dimension





Network representation

- Vast number of datasets are describing a network
  - Social relations
  - Plain old web pages with links
  - Anything where some entities in data are related to each other

Analyzing the Data

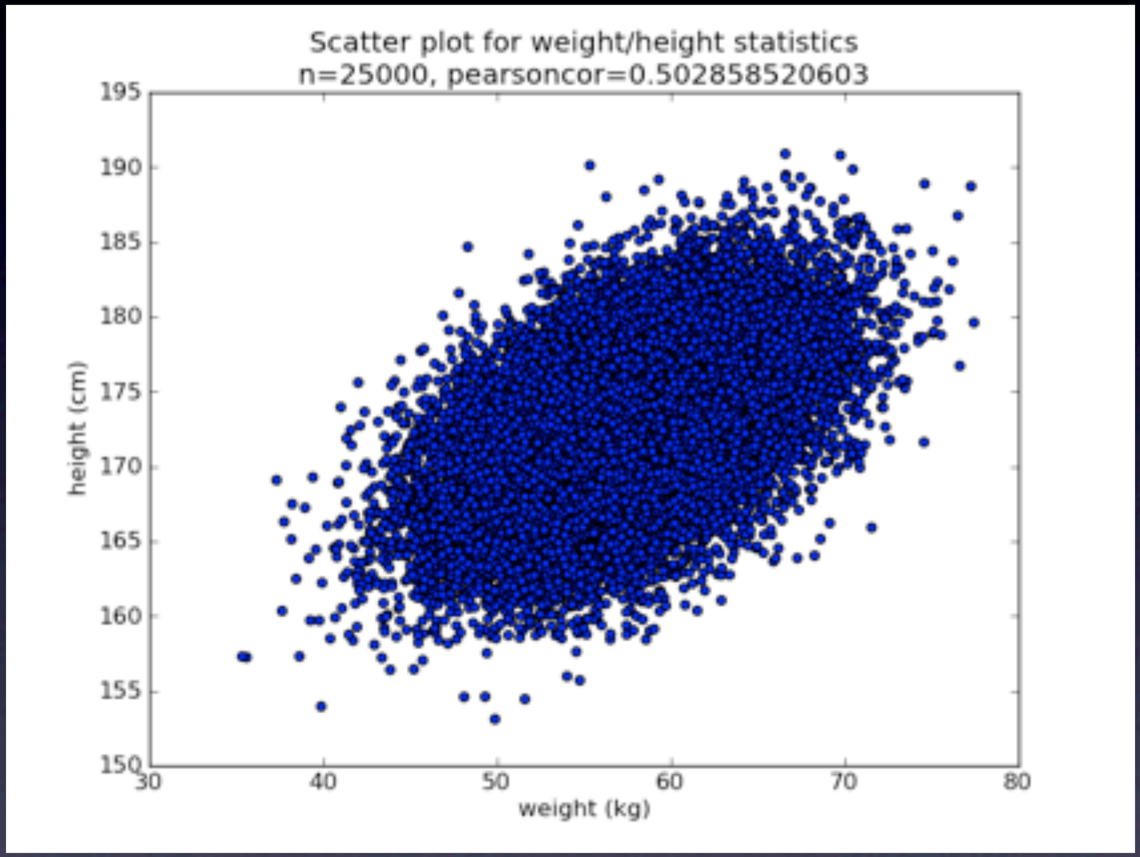
NumPy

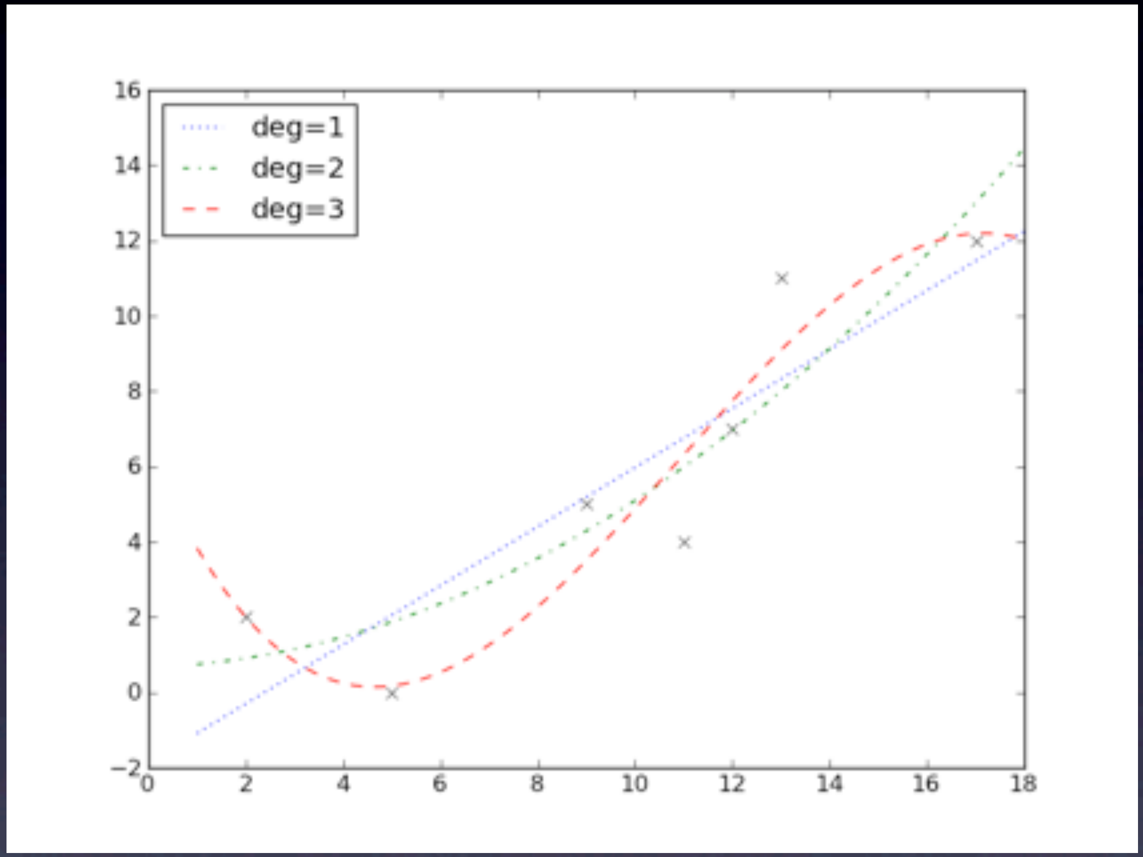
- Offers efficient multidimensional array object, *ndarray*
- Basic linear algebra operations and data types
- Requires GNU Fortran

SciPy.org

- Builds on top of NumPy
- Modules for
  - statistics, optimization, signal processing, ...
- Add-ons (called SciKits) for
  - machine learning
  - data mining
  - ...

matplotlib





```
>>> pylab.scatter(weights, heights)
>>> xlabel("weight (kg)")
>>> ylabel("height (cm)")
>>> pearsonr(weights, heights)
(0.5028, 0.0)
```

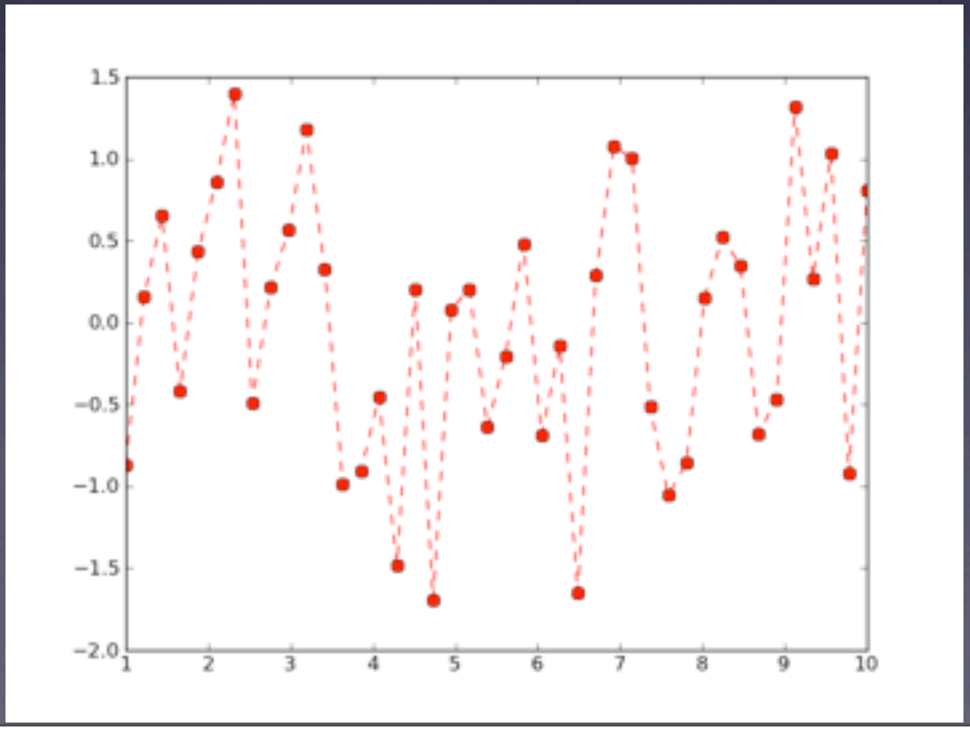
Curve fitting with numpy polyfit & polyval functions

NumPy + SciPy + Matplotlib + IPython

- Provides Matlab ”-ish” environment
- ipython provides extended interactive interpreter (tab completion, magic functions for object querying, debugging, ...)

ipython --pylab

```
In [1]: x = linspace(1, 10, 42)
In [2]: y = randn(42)
In [3]: plot(x, y, 'r--.', markersize=15)
```



Analyzing networks

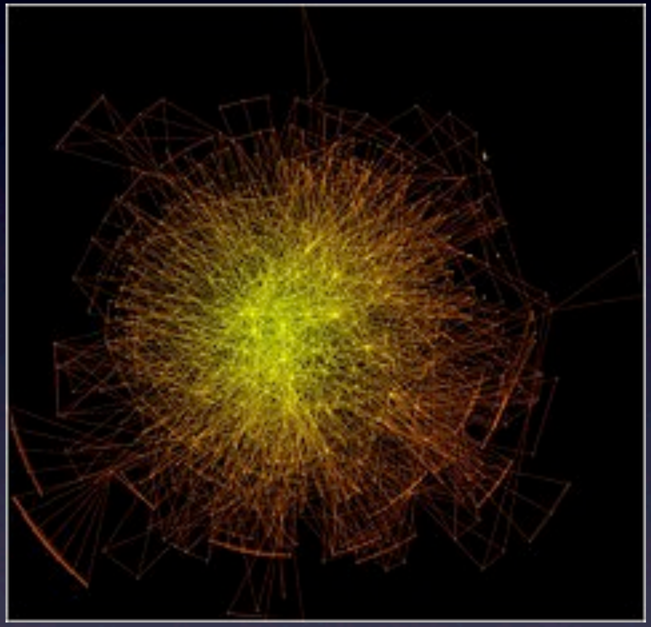
- Basicly two bigger alternatives
  - NetworkX
  - igraph

NetworkX

- Python package for dealing with complex networks:
  - Importers / exporters
  - Graph generators
  - Serialization
  - Algorithms
  - Visualization

NetworkX

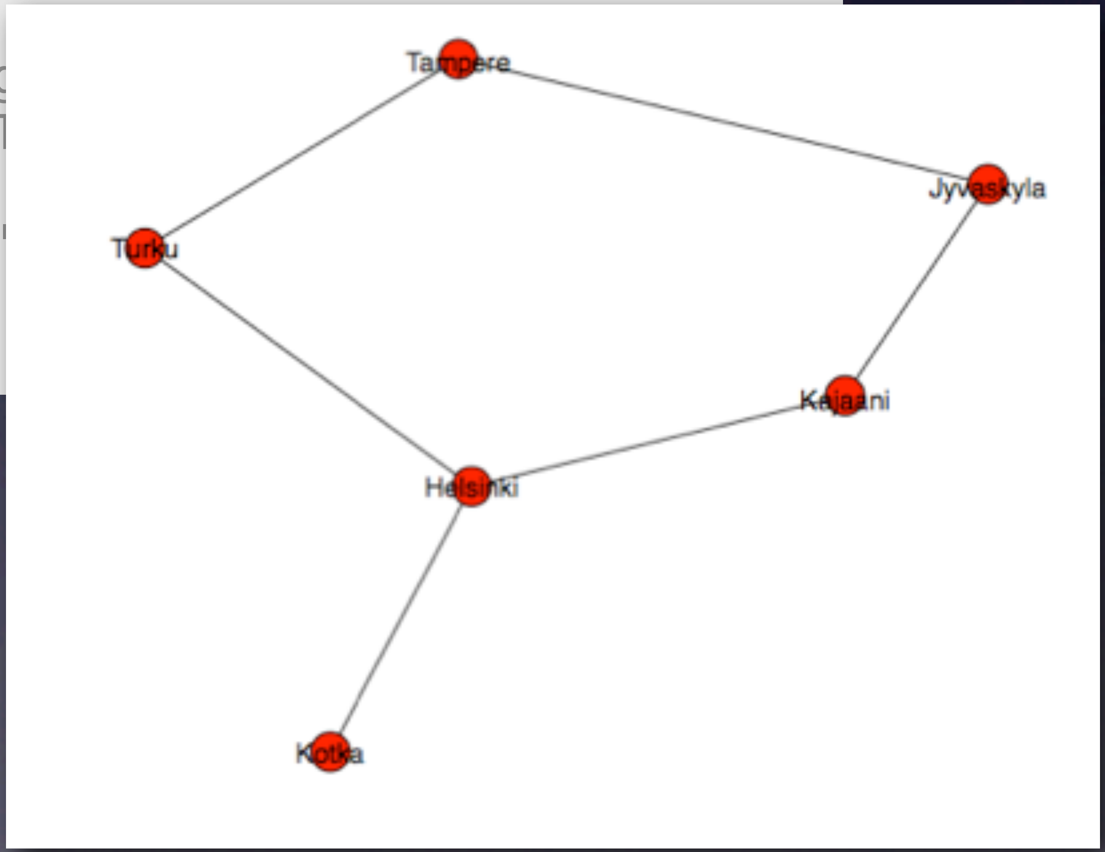
```
>>> import networkx
>>> g = networkx.Graph()
>>> data = {'Turku':     [('Helsinki', 165), ('Tampere', 157)],
            'Helsinki':  [('Kotka', 133), ('Kajaani', 557)],
            'Tampere':   [('Jyväskylä', 149)],
            'Jyväskylä': [('Kajaani', 307)] }
>>> for k,v in data.items():
...     for dest, dist in v:
...         g.add_edge(k, dest, weight=dist)
>>> networkx.astar_path_length(g, 'Turku', 'Kajaani')
613
>>> networkx.astar_path(g, 'Kajaani', 'Turku')
['Kajaani', 'Jyväskylä', 'Tampere', 'Turku']
```



Visualizing Data

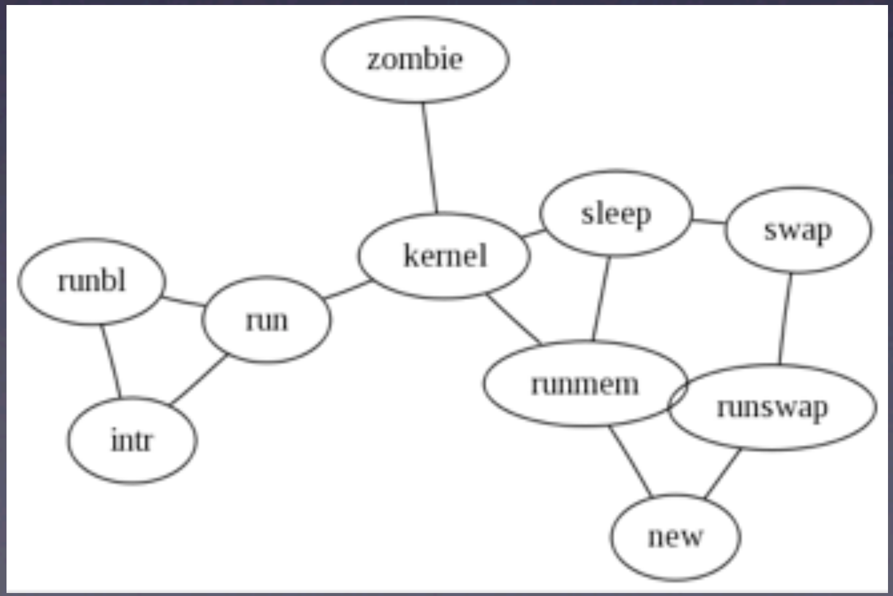
NetworkX

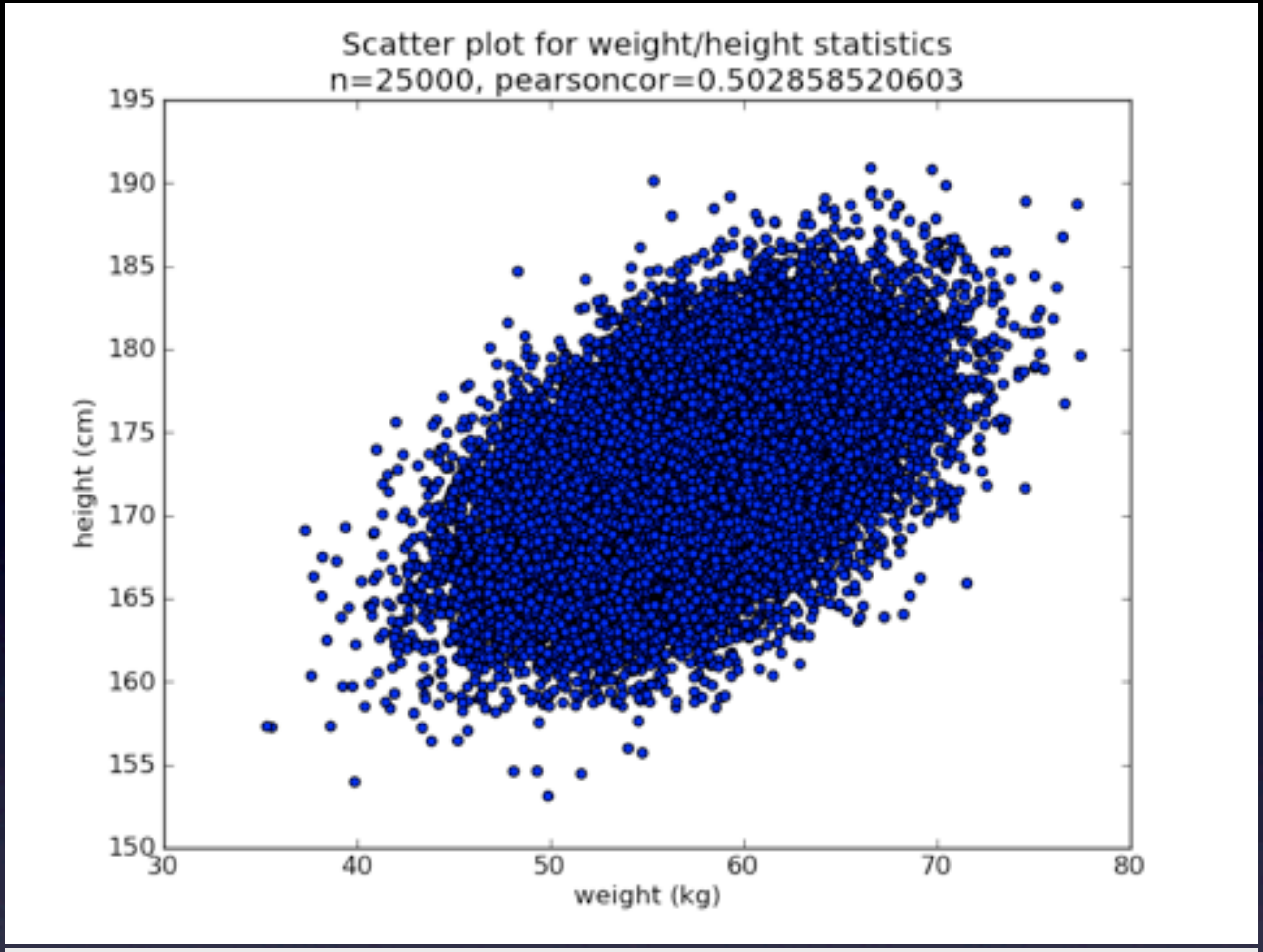
```
>>> import networkx
>>> g = networkx.Graph()
>>> data = {'Turku':     [('Helsinki', 165), ('Tampere', 157)],
            'Helsinki':  [('Kotka', 133), ('Kajaani', 557)],
            'Tampere':   [('Jyväskylä', 149)],
            'Jyväskylä': [('Kajaani', 307)] }
>>> for k,v in data.items():
...     for dest, dist in v:
...         g.add_edge(k, dest, weig
>>> networkx.astar_path_length(g, 'T
613
>>> networkx.astar_path(g, 'Kajaani'
['Kajaani', 'Jyväskylä', 'Tampere',
>>> networkx.draw(g)
```



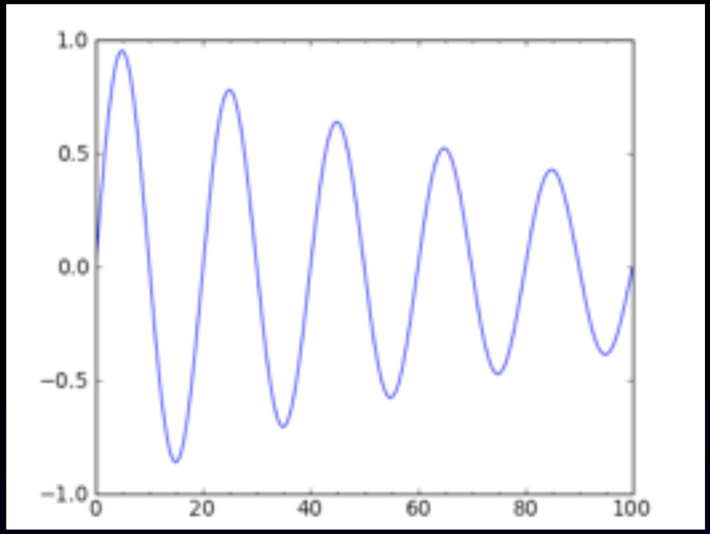
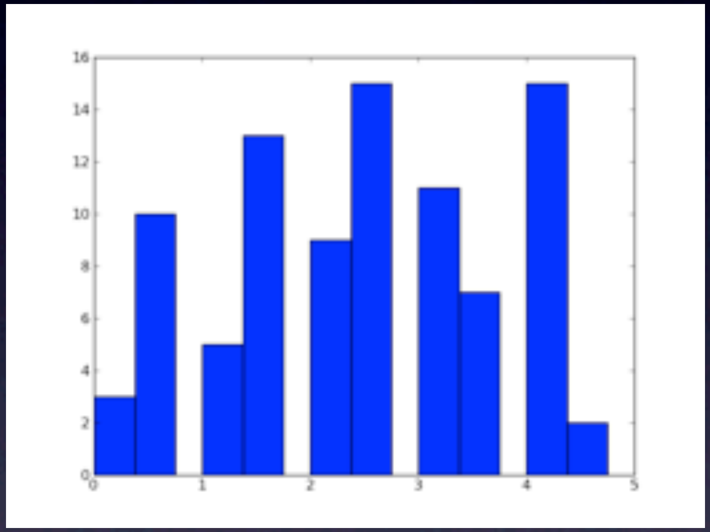
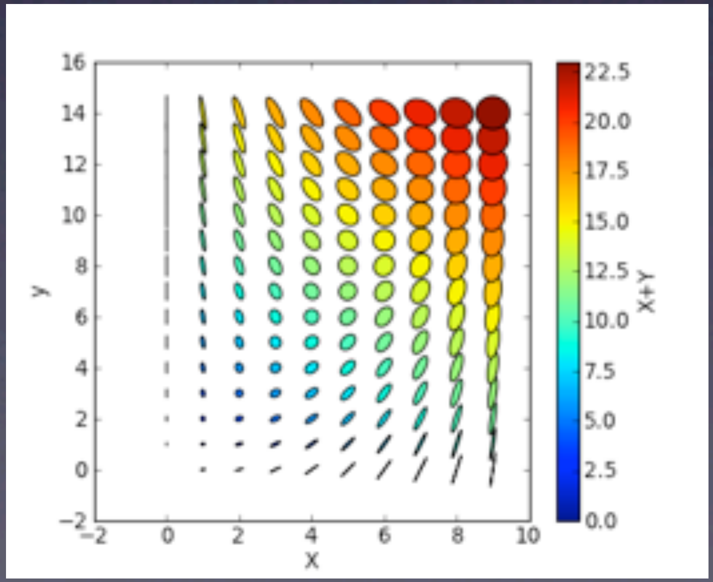
PyGraphviz

- Python interface for the Graphviz layout engine
  - Graphviz is a collection of graph layout programs



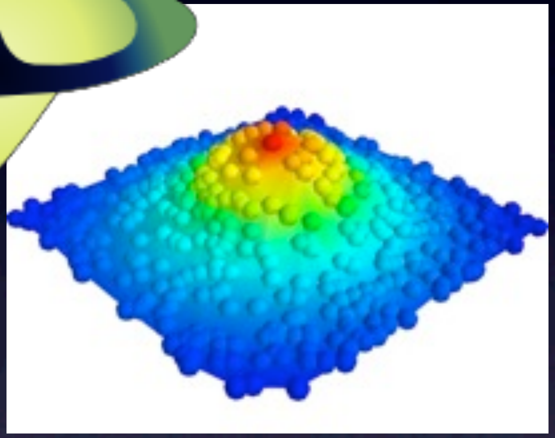


```
>>> pylab.scatter(weights, heights)
>>> xlabel("weight (kg)")
>>> ylabel("height (cm)")
>>> pearsonr(weights, heights)
(0.5028, 0.0)
>>> title("Scatter plot for weight/height
statistics\nn=25000, pearsoncor=0.5028")
>>> savefig("figure.png")
```

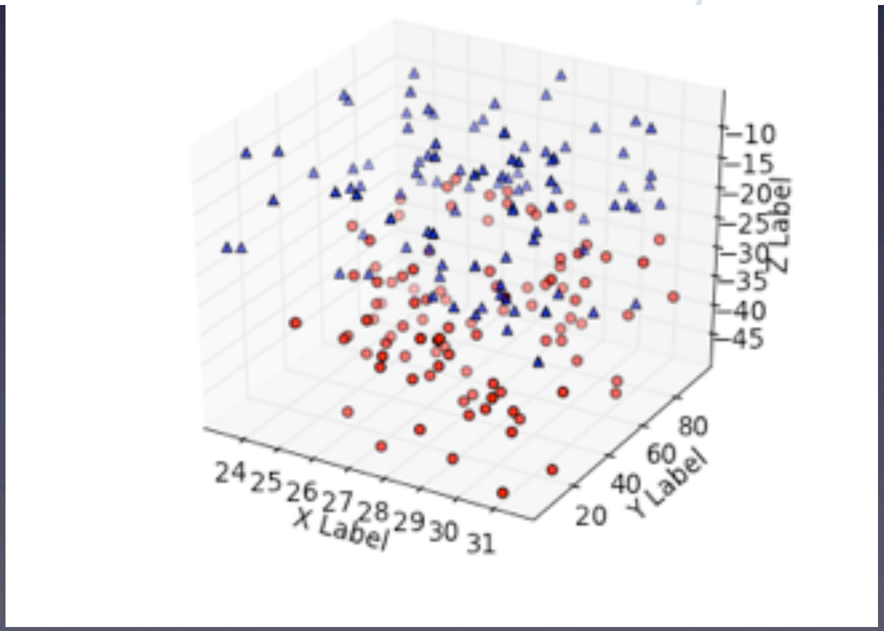
3D visualizations

Mayavi



matplotlib

with mplot3d toolkit



Data publishing

Open Data

- Certain data should be open and therefore available to everyone to use in a way or another
- Some open their data to others hoping it will be beneficial for them or just because there’s no need to hide it
- Examples of open dataset types
  - Government data
  - Life sciences data
  - Culture data
  - Commerce data
  - Social media data
  - Cross-domain data

The Zen of Open Data

Open is better than closed.
Transparent is better than opaque.
Simple is better than complex.
Accessible is better than inaccessible.
Sharing is better than hoarding.
Linked is more useful than isolated.
Fine grained is preferable to aggregated.

Optimize for machine readability — they can translate for humans.
...
“Flawed, but out there” is a million times better than “perfect, but unattainable”.
...

Chris McDowall & co.

Sharing the Data

- Some convenient formats
  - JSON (import simplejson)
  - XML (import xml)
  - RDF (import rdflib, SPARQLWrapper)
  - GraphML (import networkx)
  - CSV (import csv)

Resource Description Framework (RDF)

- Collection of W3C standards for modeling complex relations and to exchange information
- Allows data from multiple sources to combine nicely
- RDF describes data with triples
  - each triple has form *subject - predicate - object* e.g. *PyconFi2011 is organized in Turku*

RDF Data

```
@prefix poi:     <http://schema.onki.fi/poi#> .
@prefix skos:    <http://www.w3.org/2004/02/skos/core#> .

...

<http://purl.org/finnonto/id/rky/p437>
    a       poi:AreaOfInterest ;
    poi:description """Turun rautatieasema on maailmansotien välisen ajan merkittävimpiä asemarakennushankkeita Suomessa..."""@fi ;
    poi:hasPolygon "60.454833421,22.253543828 60.453846032,22.254787430 60.453815665,22.254725349..." ;
    poi:history """Turkuun suunniteltiin rautatietä 1860-luvulta lähtien, mutta ensimmäinen rautatieyhteys Turkuun..."""@fi ;
    poi:municipality kunnat:k853 ;
    poi:poiType poio:tuotantorakennus , poio:asuinrakennus , poio:puisto ;
    poi:webPage "http://www.rky.fi/read/asp/r_kohde_det.aspx?KOHDE_ID=1865" ;
    skos:prefLabel "Turun rautatieympäristöt"@fi .

...
```

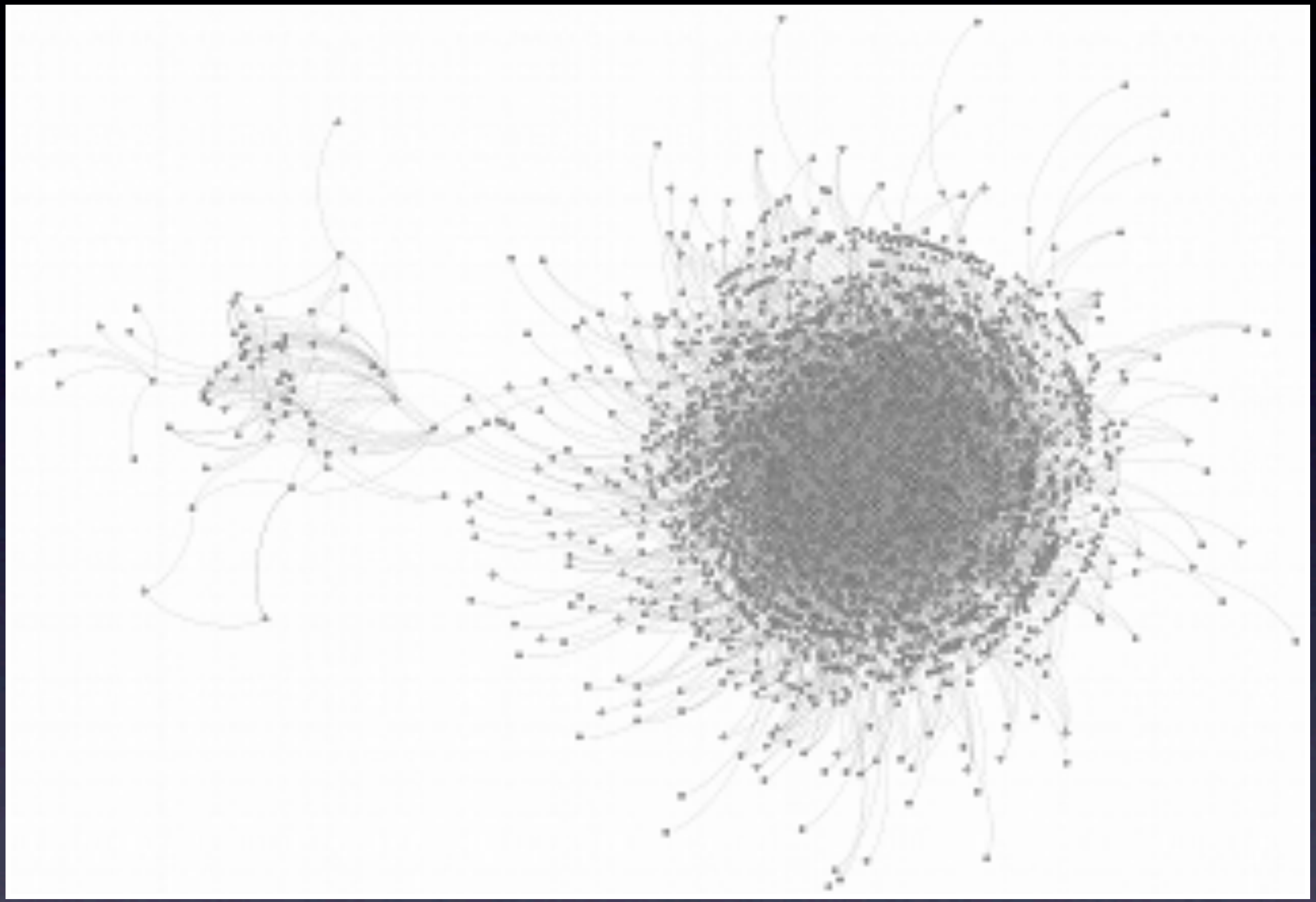
```
from SPARQLWrapper import SPARQLWrapper, JSON
QUERY = """
   Prefix lgd:<http://linkedgeodata.org/>
   Prefix lgdo:<http://linkedgeodata.org/ontology/>
   Prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>

   Select distinct ?label ?g From <http://linkedgeodata.org> {
      ?s rdf:type <http://linkedgeodata.org/ontology/Library> .
      ?s <http://www.w3.org/2000/01/rdf-schema#label> ?label.
      ?s geo:geometry ?g .
      Filter(bif:st_intersects (?g, bif:st_point (24.9375, 60.170833), 1)) .
   }"""

sparql = SPARQLWrapper("http://linkedgeodata.org/sparql")
sparql.setQuery(QUERY)
sparql.setReturnFormat(JSON)
results = sparql.query().convert()['results']['bindings']
for result in results:
    print result['label']['value'].encode('utf-8'), result['g']['value']
```

Query for libraries in Helsinki located within 1 kilometer radius from the city centre

```
German library POINT(24.9495 60.1657)
Metsätalo POINT(24.9497 60.1729)
Opiskelijakirjasto POINT(24.9489 60.1715)
Topelia POINT(24.9493 60.1713)
Eduskunnan kirjasto POINT(24.9316 60.1725)
Rikhardinkadun kirjasto POINT(24.9467 60.1662)
Helsinki 10 POINT(24.9386 60.1713)
```

RDF Data sources

- CKAN
- DBpedia
- LinkedGeoData
- DBTune
- http://semantic.hri.fi/



harri.hamalainen aalto fi