PyLadies

Vienna 15.2.2020

Who?

International mentorship group with a focus on helping more women become active participants and leaders in the Python open-source community.

Our mission is to promote, educate and advance a diverse Python community through outreach, education, conferences, events and social gatherings.

Agenda for today

---

1. Data analysis Libraries - numpy, pandas, matplotlib
2. Data analysis project
3. Exercises, networking, discussion of own projects
4. Join us for a beer (or anything else) later

Goals

- Understand your data
- Display your data
- Learn from data
- Statistics
- future steps: use data in ML algorithms

Numpy - numerical Python

- main object is **multidimensional array** (**ndarray**)
- basically n-dimensional table of elements
- indexing by tuples or positive numbers
- dimensions called axes
- heavily optimized for numerical operations on arrays
- official documentation a bit heavy, tutorials usually better for basics

Numpy - initialize first array

```
import numpy as np # import package as (shortcut or consensus)
a=np.array([[1, 2], [3, 4]])
print a
```

- numpy objects created from arrays or functions returning them

```
np.array(object, dtype = None, copy = True, order = None, subok = False, ndmin
= 0)
b = np.array([1, 2, 3, 4, 5], dtype = int, ndmin = 2)
print b
# first and second arguments are positional, rest are keyword arguments
c = np.array(ndmin = 2, object=[[1, 2, 3], [4, 5, 6]]) # works fine too
```

Array attributes and creation, stick to 2D for now

```
d = np.array([[1,2,3],[4,5,6]])
d.shape # returns a tuple with array dimensions
e = d.reshape(3,2) # changes dimensions, does not modify initial object
e.flatten() # collapses array to one dimensions
f = np.arange(24) # creates evenly spaced array
# np.arange([start,] stop[, step,], dtype=None) - just like range()
g = np.empty([3,2], dtype = int) # prefilled with random numbers
h = np.zeros([3,2], dtype = float) # good for initialization
i = np.ones([2,3], dtype = bool) # can be used for boolean masking
j = [(1,2,3),(4,5)]
k = np.asarray(j) # accepts not only array, but any sequence - tuple etc
```

Indexing, slicing

```
m = np.arange(10)
m[2:7:2] # returns array([2, 4, 6])
m[4:] # returns array([4, 5, 6, 7, 8, 9])
m[4] # returns 4
m[:4] # returns array([0, 1, 2, 3])
# how about multidimensional arrays? same principle
n = np.array([[1, 2, 3], [3, 4, 5], [4, 5, 6]])
n[1:] # returns [[3 4 5], [4 5 6]]
n[2, 2] # returns 6
# elipsis (...) can be used to make a selection same length as dim of array
n[1, ...] # returns items of a second row [3, 4, 5]
n[..., 1] # returns items of a second column [2, 4, 5]
```

Some random picks

- Broadcasted arithmetic, dim of smaller array fits larger

```
a = np.array([[0,0,0],[10,10,10],[20,20,20],[30,30,30]])
b = np.array([1,2,3])
a + b # returns array([[ 1,  2,  3], [11, 12, 13], [21, 22, 23], [31, 32,
33]])
```

- np functions **add**, **subtract**, **multiply**, **divide**, **power**
- Iterating over array:

```
for i in array:  # per row
for x in np.nditer(a):  # per item
```

- np functions **sort(a, axis=0), c = np.where(a > 15)**

Pandas

- `import pandas as pd`
- new data structures for easier handling of data
- loading functions - **read_csv**
- data transformation
- data handling
- joins and merge

Series

- One-dimensional array-like object containing data and labels (or index)
- `s = pd.Series([1,2,2,4])`
- Index can be specified -> `s.index=['a', 'b', 'c', 'd']`
- `s[['a']]`
- `s[['a', 'c']]`
- unlike dictionaries, index don’t have to be unique

Series

- Numeric operations - like in numpy
- s >= 4
- pandas can work with incomplete data
- s2 = pd.Series(s, list('abcde'))
- data are automatically aligned

Data Frame (df)

- data structure collecting ordered **collection of columns**
- “table-like”
- has row and column index
- Creation from dictionary:
  `pd.DataFrame(dict.items())`
- Can be also created by dictionary of dictionaries
- access same way as Series -> df['col1']

Data Frame (df)

- df.describe() - display basic **stats** about df
- new columns can be easily added by direct assign or derived - df['col3'] = df['col1'] * 2
- Pandas and dataframes are great with statistic :
  - df.sum()
  - df.mean()
  - df.dtypes
  - etc..
- comparisons: df < 3
- you can **stack** the methods together - data.groupby(by ='col1')['number'].sum().sort_values(ascending =False).head(10)

loc, iloc, index

- data selection options
- iloc - integer-location based indexing
  - `data.iloc[0]`
  - `data.iloc[:,1]`
  - `data.iloc[0:5, 5:8]`
- loc - selection by name
  - `data.loc[2] - by index of a row (name)`
  - `df.loc[[2,3], ['gender', 'state']]`
  - `df.loc[df['state'] == 'dc']`
- index - df.ix[[3]] - not supported anymore, use methods above

Load data with pandas

- pd.read_csv
- works well with JSON and other structured data
- pd.read_sql - for connection to database
- with additional library also reads excel files

Matplotlib

- `import matplotlib.pyplot as plt`
- multiple types of plots, subplots
- good documentation and tutorials

Matplotlib

Basic plot:

```
x = np.linspace(0, 10, 100)
plt.plot(x, x, label='linear')
plt.legend()
plt.show()
```

Two main components: figures and axis

Matplotlib

- barplot
- line plot
- scatter plot
- pie charts
- box plots
- https://matplotlib.org/gallery.html
- for more fancy plots: use **seaborn**

Plots with pandas

- `df=pd.DataFrame({'name':['john','mary','peter','jeff','bill','lisa','jose'],'age':[23,78,22,19,45,33,20],'gender':['M','F','M','M','M','F','M'],'state':['california','dc','california','dc','california','texas','texas'],'num_children':[2,0,0,3,2,1,4],'num_pets':[5,1,0,5,2,2,3]})`
- `df.plot(kind='scatter',x='num_children',y='num_pets',color='red')`
- `plt.show()`

Project - fires in Brazil

- data - https://www.kaggle.com/gustavomodelli/forest-fires-in-brazil
- goal:
  - understand the data and create nice visualisations
  - practice different plots
  - practice pandas data handling

Exercises

---

1. load data into python
2. get minimal and maximal year
3. display statistics
4. get total number of fires in each state
5. visualize number of fires by year for chosen state
6. which 3 states have highest number of fires?
7. create pie chart with percentage of fires per state

Spotify Dataset

- https://www.kaggle.com/leonardopena/top-50-spotify-songs-by-each-country

1. Choose one country you like and find most popular song
2. Find most popular genres
3. Display average popularity of artist
4. Discover most popular genre among countries
5. Find something you find interesting :)
6. Have fun exploring

Resources

- Kaggle: https://www.kaggle.com/
- Geeks for geeks: https://www.geeksforgeeks.org/
- Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython

Resources and materials general

- advent of code - adventofcode.org
- hackerrank - hackerrank.com
- Django Girls - django tutorial (25th of April in Vienna)
- https://www.practicepython.org
- Nice Python exercices at one place https://github.com/tystar86/python_exercises/tree/master/Tasks
- https://automatetheboringstuff.com
- https://diveintopython3.problemsolving.io

Next topics

Databases

Object oriented programming

Testing

Flask

fill the form from last time please :) → https://forms.gle/UtfgVGe6AhhRwx539

Thank you and see you next time

Coding session - 27.2.2020

Next workshop - 21.3.2020