Data Science in Python

ESC 2K18

Hi, I’m Stefania Delprete, nice to meet you!

Data Scientist in TOP-IX, BIG DIVE co-organizer

stefania.delprete@top-ix.org // @astrastefania

NumFOCUS ambassador, PyCon / EuroPython volunteer

Python, PyCon, EuroPython...

NumFOCUS

The mission of NumFOCUS, 501(c)3 non-profit organization, is to promote sustainable high-level programming languages, open code development, and reproducible scientific research.

- https://www.numfocus.org
- https://pydata.org

NumFOCUS

Filter Projects

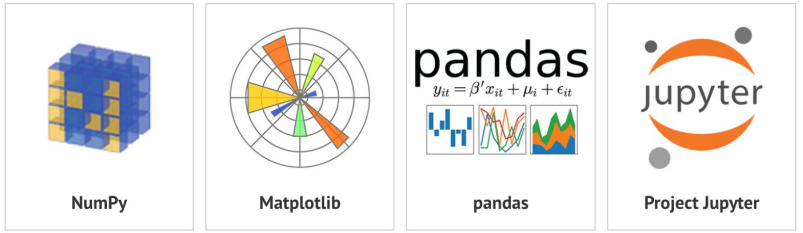
Show All Projects

Language

- Python
- R
- Julia
- JavaScript
- Multiple
- Other

Features

- Data Wrangling
- Modeling
- Visualization
- High Performance Computing
- Big Data
- Statistical Computing
- Numerical Computing
- Data Mining
- Text Processing



Jupyter Project

Notebooks to run present your code and results, great tool to learn and teach

2001 IPython Fernand Pérez, physicist

> Jupyter Notebook > Jupyter Lab

You can try it online too:

- https://jupyter.org/try

NumPy

Scientific computation, random numbers, mathematical functions, linear algebra...

1995 Numeric, Jim Hugunin, developer

2006 Travis Oliphant, data scientist

- http://www.numpy.org
- https://github.com/numpy/numpy

NumPy

Generating unidimentional arrays

```
np.zeros(9) # array of 9 zeros
```

```
array([0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

```
nine_ones = np.ones(9) # array of 9 ones
nine_ones
```

```
array([1., 1., 1., 1., 1., 1., 1., 1., 1.])
```

```
np.append(nine_ones,4) # add 4 at the end
```

```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 4.])
```

```
np.insert(nine_ones,8,3.14) # insert at position 8, np.insert(np-array,index,value)
```

```
array([1.  , 1.  , 1.  , 1.  , 1.  , 1.  , 1.  , 1.  , 3.14, 1.  ])
```

```
np.arange(8)
```

```
array([0, 1, 2, 3, 4, 5, 6, 7])
```

```
np.arange(0,15,3)
```

```
array([ 0,  3,  6,  9, 12])
```

NumPy

```
# We can round the randomly generated elements
mat4 = np.random.random((4,4))
mat4
```

```
array([[0.88011078, 0.69001378, 0.17171111, 0.07590845],
       [0.60435452, 0.37537713, 0.25559777, 0.79863126],
       [0.67484935, 0.69988719, 0.26587466, 0.41250546],
       [0.03134224, 0.41878734, 0.49221183, 0.38166006]])
```

```
mat4_round = np.round(mat4, 2)
mat4_round
```

```
array([[0.88, 0.69, 0.17, 0.08],
       [0.6 , 0.38, 0.26, 0.8 ],
       [0.67, 0.7 , 0.27, 0.41],
       [0.03, 0.42, 0.49, 0.38]])
```

```
np.sort(mat4_round)
```

```
array([[0.08, 0.17, 0.69, 0.88],
       [0.26, 0.38, 0.6 , 0.8 ],
       [0.27, 0.41, 0.67, 0.7 ],
       [0.03, 0.38, 0.42, 0.49]])
```

Find a dataset...

Datasets

Example of resources to find datasets:

- Kaggle https://www.kaggle.com/unsdsn/world-happiness/data
- Nasa open data https://data.nasa.gov
- Gov data (https://data.gov https://dataportal.daf.teamdigitale.it)
- Commons like OMS https://www.openstreetmap.org and Common Voice https://voice.mozilla.org/en/data
- Surveys (example https://eahub.org/survey)
- ...or make your own!

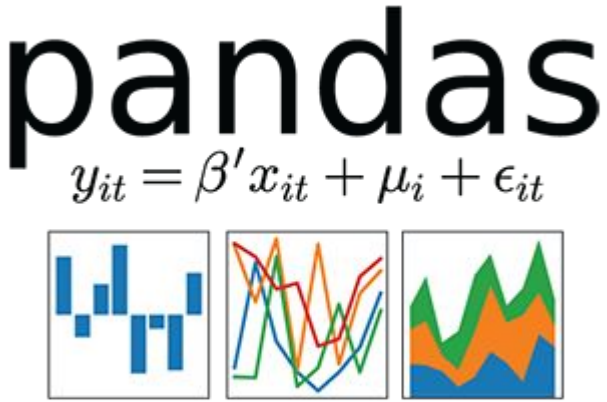
...and start explore it!

Pandas

Great tool to manipulate datasets (pandas DataFrame), powerful indexing, management of missing values…

2008 Wes McKinney, statistician

- http://pandas.pydata.org
- https://github.com/pandas-dev/pandas



Pandas sprints

- https://python-sprints.github.io/pandas
- https://github.com/astrastefania/pandas-sprint-Turin

Pandas

```
agencies_dict = {'agency_code': ['ASI', 'ESA', 'NASA', 'ROSCOSMOS', 'ISRO', 'ISA', 'JAXA'],
          'agency_name': ['Agenzia Spaziale Italiana',
                          'European Space Agency',
                          'National Aeronautics and Space Administration',
                          'Russian Federal Space Agency',
                          'Indian Space Research Organisation',
                          'Israeli Space Agency', 'Japan Aerospace Exploration Agency'],
          'country': ['Italy', 'Europe', 'United States', 'Russia', 'India', 'Israel', 'Japan'],
          'year_founded': [1988, 1975, 1958, 1992, 1969, 1983, 2003]}
```

```
# We also call the columns 'variables' and the rows 'observables'

agencies_df = pd.DataFrame(agencies_dict)
agencies_df
```

	agency_code	agency_name	country	year_founded
0	ASI	Agenzia Spaziale Italiana	Italy	1988
1	ESA	European Space Agency	Europe	1975
2	NASA	National Aeronautics and Space Administration	United States	1958
3	ROSCOSMOS	Russian Federal Space Agency	Russia	1992
4	ISRO	Indian Space Research Organisation	India	1969
5	ISA	Israeli Space Agency	Israel	1983
6	JAXA	Japan Aerospace Exploration Agency	Japan	2003

Pandas

```
agencies_df['years'] = agencies_df['year_founded'].apply(lambda year: 2018 - year)
agencies_df
```

	agency_code	agency_name	country	year_founded	years
0	ASI	Agenzia Spaziale Italiana	Italy	1988	30
1	ESA	European Space Agency	Europe	1975	43
2	NASA	National Aeronautics and Space Administration	United States	1958	60
3	ROSCOSMOS	Russian Federal Space Agency	Russia	1992	26
4	ISRO	Indian Space Research Organisation	India	1969	49
5	ISA	Israeli Space Agency	Israel	1983	35
6	JAXA	Japan Aerospace Exploration Agency	Japan	2003	15

```
agencies_df.describe()
```

	year_founded	years
count	7.000000	7.000000
mean	1981.142857	36.857143
std	15.093360	15.093360
min	1958.000000	15.000000
25%	1972.000000	28.000000
50%	1983.000000	35.000000
75%	1990.000000	46.000000
max	2003.000000	60.000000

Pandas

```
happy = pd.read_csv('data/happiness/2017.csv') # Importing a .csv file
```

```
happy.columns
```

```
Index(['Country', 'Happiness.Rank', 'Happiness.Score', 'Whisker.high',
       'Whisker.low', 'Economy..GDP.per.Capita.', 'Family',
       'Health..Life.Expectancy.', 'Freedom', 'Generosity',
       'Trust..Government.Corruption.', 'Dystopia.Residual'],
      dtype='object')
```

```
happy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 155 entries, 0 to 154
Data columns (total 12 columns):
Country                        155 non-null object
Happiness.Rank                 155 non-null int64
Happiness.Score                155 non-null float64
Whisker.high                   155 non-null float64
Whisker.low                    155 non-null float64
Economy..GDP.per.Capita.       155 non-null float64
Family                         155 non-null float64
Health..Life.Expectancy.       155 non-null float64
Freedom                        155 non-null float64
Generosity                     155 non-null float64
Trust..Government.Corruption.  155 non-null float64
Dystopia.Residual              155 non-null float64
dtypes: float64(10), int64(1), object(1)
memory usage: 14.6+ KB
```

Pandas

```
happy_ = happy.loc[:, ['Country', 'Happiness.Rank', 'Happiness.Score','Freedom']]
happy_.head(3)
```

	Country	Happiness.Rank	Happiness.Score	Freedom
0	Norway	1	7.537	0.635423
1	Denmark	2	7.522	0.626007
2	Iceland	3	7.504	0.627163

```
# Changing columns name
happy_.columns = ['country', 'rank', 'score', 'freedom']
happy_.head(3)
```

	country	rank	score	freedom
0	Norway	1	7.537	0.635423
1	Denmark	2	7.522	0.626007
2	Iceland	3	7.504	0.627163

```
# Changing only one column name
happy_ = happy_.rename(columns = {'freedom':'free'})
happy_.head(3)
```

	country	rank	score	free
0	Norway	1	7.537	0.635423
1	Denmark	2	7.522	0.626007
2	Iceland	3	7.504	0.627163

Leverage the power of visualizations

Matplotlib

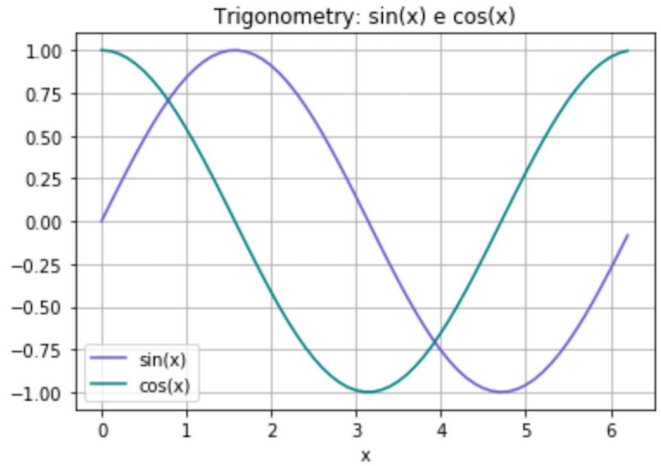
Statistical graph. Powerful and light, easy to customize.

2003 John D. Hunter, neurobiologist

- https://matplotlib.org
- https://github.com/matplotlib/matplotlib

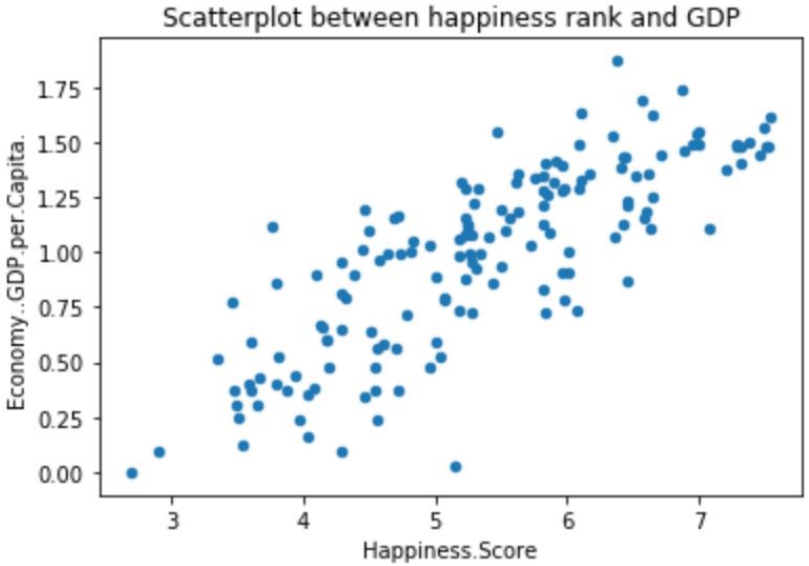
Matplotlib

```
# Generating points with NumPy
x = np.arange(0, np.pi*2, 0.1)
y_sin = np.sin(x)
y_cos = np.cos(x)

plt.plot(x, y_sin,'slateblue', label="sin(x)")
plt.plot(x, y_cos,'teal', label="cos(x)")

# Describing the graph
plt.title('Trigonometry: sin(x) e cos(x)')
plt.xlabel('x')

# Creating a legend
plt.legend()

# Showing the grid
plt.grid(True, which='both')

plt.show()
```



Matplotlib

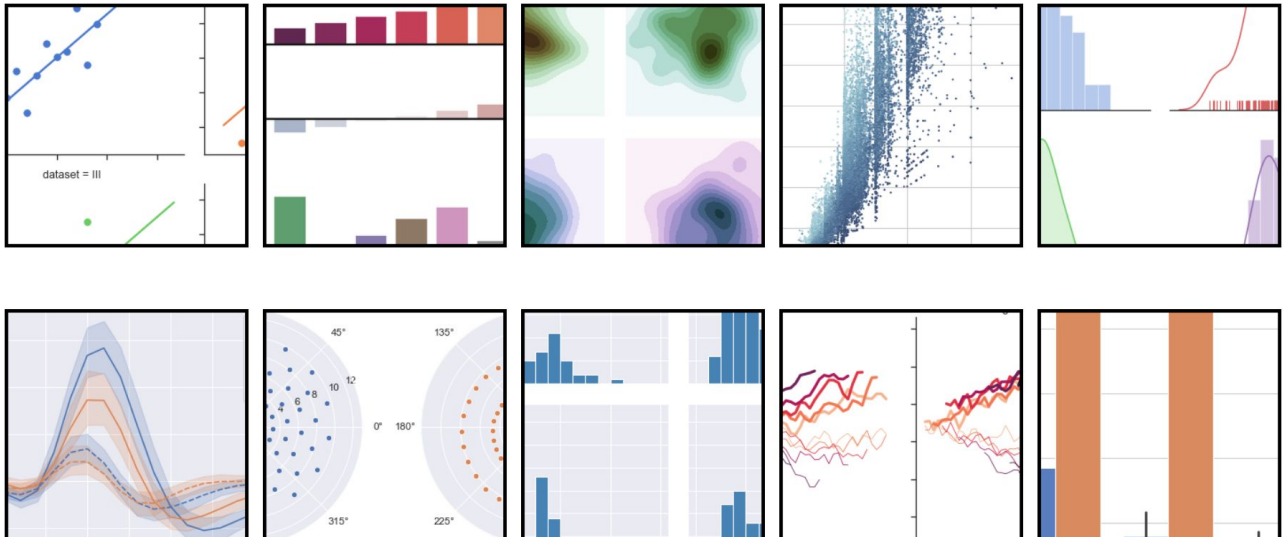
```
happy.plot(x='Happiness.Score', y='Economy..GDP.per.Capita.', kind='scatter')
plt.title('Scatterplot between happiness rank and GDP')

plt.show()
```



Seaborn

Built on Matplotlib “making attractive and informative statistical graphics in Python”

- https://seaborn.pydata.org
- https://github.com/mwaskom/seaborn



Interactive graphs

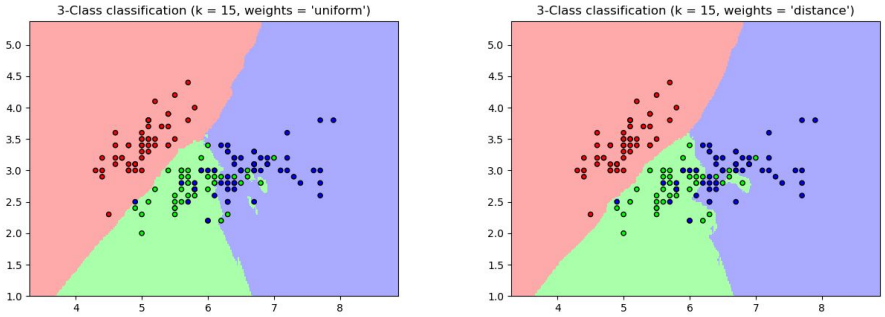
Bokeh

- https://bokeh.pydata.org
- https://demo.bokehplots.com/apps/gapminder
- https://github.com/bokeh/bokeh.git

Go deeper and predict

Machine Learning and Deep Learning

- http://scikit-learn.org
- https://www.tensorflow.org
- https://keras.io
- https://pytorch.org



Examples:

- Nearest Neighbors Classification: an example of classification using nearest neighbors.

1.6.3. Nearest Neighbors Regression

Neighbors-based regression can be used in cases where the data labels are continuous rather than discrete variables. T
label assigned to a query point is computed based the mean of the labels of its nearest neighbors.

scikit-learn implements two different neighbors regressors: `KNeighborsRegressor` implements learning based on the k
nearest neighbors of each query point, where k is an integer value specified by the user. `RadiusNeighborsRegressor`
implements learning based on the neighbors within a fixed radius r of the query point, where r is a floating-point value

Thank you!

Stefania Delprete

stefania.delprete@top-ix.org

@astrastefania