Unione Parmense degli Industriali

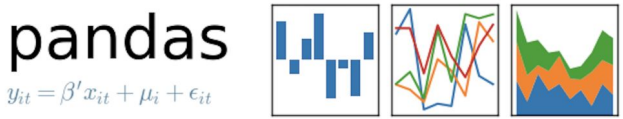
Gruppo Imprese Artigiane

Pandas and Data visualization

Internet of Things e Analisi predittiva

Gianfranco Lombardo MEng, Ph.D Candidate in ICT
gianfranco.lombardo@unipr.it

Pandas



- Library for data structures and data analysis
  - High performance
  - Easy to use
- Useful to import datasets, for data cleaning, data splitting
- Easy import of CSV files and their management
- Base module for several projects in statistical modeling and machine learning
- Install with : pip install pandas

- Main concept : DataFrame (can be seen as an Excel worksheet)
- Values in a DataFrame are in the form of numpy arrays with indexes for rows and columns

Creating a DataFrame

Py

```
>>> import pandas as pd
>>> df = pd.DataFrame([[1, 2], [4, 5], [7, 8]],
                      index=['cobra', 'viper',
'sidewinder'],
                      columns=['max_speed', 'shield'])
>>> df
             max_speed  shield
cobra                1       2
viper                4       5
sidewinder           7       8
```

Selection of columns

Py

```
>>> df['max_speed']  # or simply df.max_speed
cobra         1
viper         4
sidewinder    7
Name: max_speed, dtype: int64

>>> df[['max_speed', 'shield']]
            max_speed  shield
cobra               1       2
viper               4       5
sidewinder          7       8
```

Selection of rows

- With brackets, a slice of rows can be selected
  - Provide both start and stop, not a single position or label
  - With labels, *the stop label is included!*

Py

```
>>> df[1:3]
            max_speed  shield
viper               4       5
sidewinder          7       8

>>> df['viper':'sidewinder']
            max_speed  shield
viper               4       5
sidewinder          7       8
```

Selection in both axis

- Slicing by labels (*the stop label is included!*)

```
>>> df.loc['viper':'sidewinder', 'max_speed':'shield']
            max_speed  shield
viper                4       5
sidewinder           7       8
```

- Slicing by positions (*the stop position is not included!*)

```
>>> df.iloc[1:3, 0:2]
            max_speed  shield
viper                4       5
sidewinder           7       8
```

Set a value

- Create a copy of the DataFrame with copy()
- Use assignment operator and loc to set new values

Py

```
>>> df2 = df.copy()
>>> df2.loc[['viper', 'sidewinder'], ['shield']] = 50
>>> df2
             max_speed  shield
cobra                1       2
viper                4      50
sidewinder           7      50
```

Adding a column and isin method

```
>>> df2 = df.copy()
>>> df2['label'] = ['one','two','three']
>>> df2
            max_speed  shield  label
cobra               1       2    one
viper               4       5    two
sidewinder          7       8  three


#isin as another way for slicing on both axis


>>> df2[df2['label'].isin(['one','two'])]
            max_speed  shield  label
cobra               1       2    one
viper               4       5    two
```

Sorting by value or index

```
>>> df.sort_values(by='shield')   # asc values of shield
            max_speed  shield
cobra               1       2
viper               4       5
sidewinder          7       8

>>> df.sort_index(axis=1, ascending=False)   # desc column names
            shield  max_speed
cobra            2          1
viper            5          4
sidewinder       8          7
```

Sorting by value or index

```
>>> df.sort_values(by='shield')   # asc values of shield
            max_speed  shield
cobra               1       2
viper               4       5
sidewinder          7       8

>>> df.sort_index(axis=1, ascending=False)   # desc column names
            shield  max_speed
cobra            2          1
viper            5          4
sidewinder       8          7
```

DataFrame from Python dictionaries

```
df = pd.DataFrame({
            'shield': np.array([2, 5, 8], dtype='int32'),
            'max_speed': np.array([1, 4, 7], dtype='int32') },
        index=['cobra', 'viper', 'sidewinder'])

df1 = pd.DataFrame('Animal': ['Falcon', 'Falcon',
                              'Parrot', 'Parrot'],
                              'Max Speed': [380., 370., 24., 26.]})

>>> df1
   Animal  Max Speed
0  Falcon      380.0
1  Falcon      370.0
2  Parrot       24.0
3  Parrot       26.0
```

Read and export DataFrame from csv or Excel

```
df2 = pd.read_csv('data.csv')
df2.to_csv('data.csv')

df3 = pd.read_excel('data.xlsx')
      df3.to_excel('data.xlsx')
```

Group by

- Split data into groups based on some criteria
- Apply a function to each group independently
- Combine the results into a data structure

```
>>> df1
   Animal  Max Speed
0  Falcon      380.0
1  Falcon      370.0
2  Parrot       24.0
3  Parrot       26.0

df1.groupby(['Animal']).mean()  ## try also other function like sum()
        Max Speed
Animal
Falcon      375.0
Parrot       25.0
```

Py

CSV EXAMPLE: registry.csv

```
Name,Gender,Age,Job,City                    #Header
George,male,34,Waiter,Chicago
Alice,female,27,Developer,New York
Mario,male,57,Plumber, New York
Lauren,female,42,Teacher,Chicago
Robert,male,29,Engineer,London
```

Example:

```
import pandas as pd
import numpy as np
df = pd.read_csv("registry.csv")
df = df.sort_values(by='Age') #sort by age
print(df)
df_2 = df.groupby(['Gender']).mean() #group by gender and get average
value

#Retrieve a column in a numpy array
ages = np.array(df['Age'])
print(ages)
print(np.mean(ages))
```

Example:

```
#Add random salary in a column
salary = np.random.choice(50000,len(df.index))
print(salary)
df['Salary']=salary
print(df)

#Group by Gender and get average salary and age
df_3 = df.groupby(['Gender']).mean()
print(df_3)
```

Exercise: Prices of houses

- Starting from “house.csv” dataset
  - Keep only these columns in a new dataset:
    - price,bedrooms,bathrooms,sqft_living,sqft_lot,floors,sqft_above,sqft_basement,yr_built,yr_renovated,city
  - Compute the average price for the entire dataset
  - Compute the average price group by City
  - Find where the price is the highest and where the smallest

Exercise: Prices of houses

- Add a column “sqft_total” = sqft_living + sqft_lot + sqt_above + sqft_basement
- Group houses by sqft_total and get the average price group by City

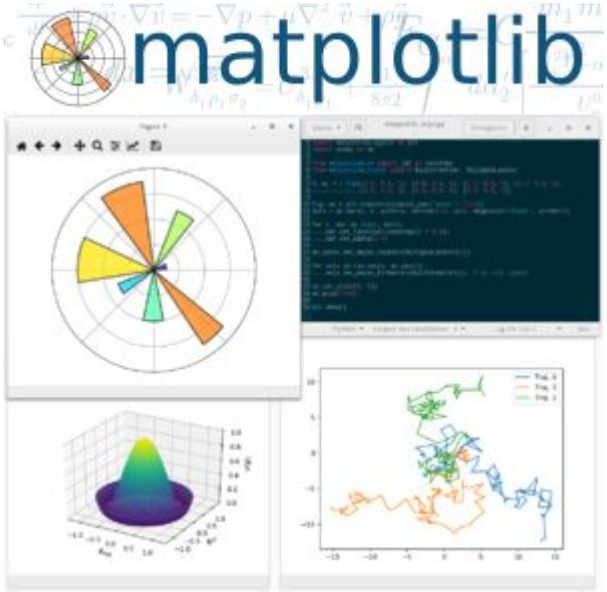
BREAK

Data visualization

- Data visualization is a branch of Data analysis that aims to provide graphical tools to get a representation of information
  - Visual elements: charts,histograms,graphs and maps
- It provides an accessible way to detect and understand trends, outliers and patterns among data
- Fundamental tool in the world of Big Data for data-driven decisions

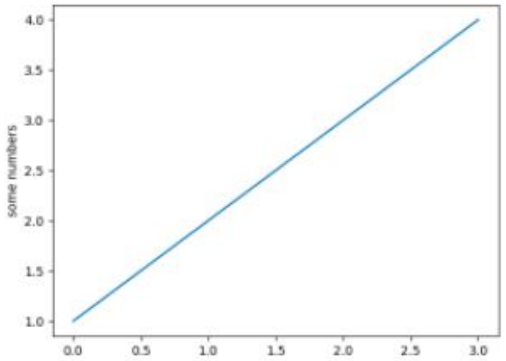
Matplotlib

- Python module for data visualization (not the only one)
- It supports numpy and Pandas
- First release: 2003
- It provides an interface Matlab-like
- Install with pip install matplotlib
- Import:
  - import matplotlib as plt

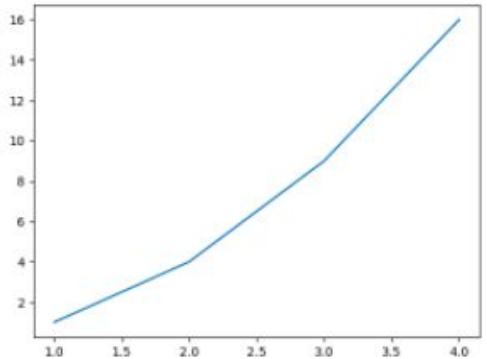


Basic plot

```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4])
plt.ylabel('some numbers')
plt.show()
```

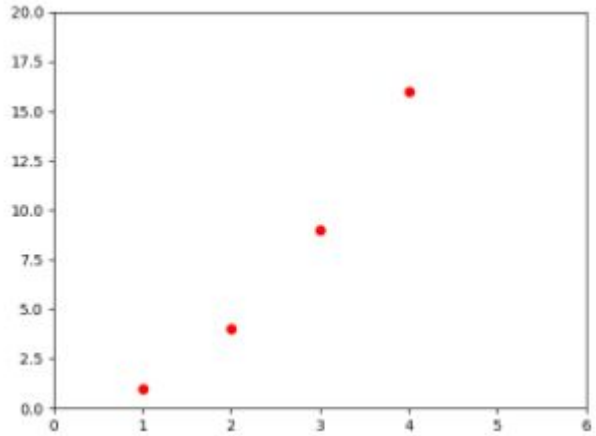



```
# plot x versus y
plt.plot([1, 2, 3, 4], [1, 4, 9, 16])
plt.show()
```



Formatting the plot

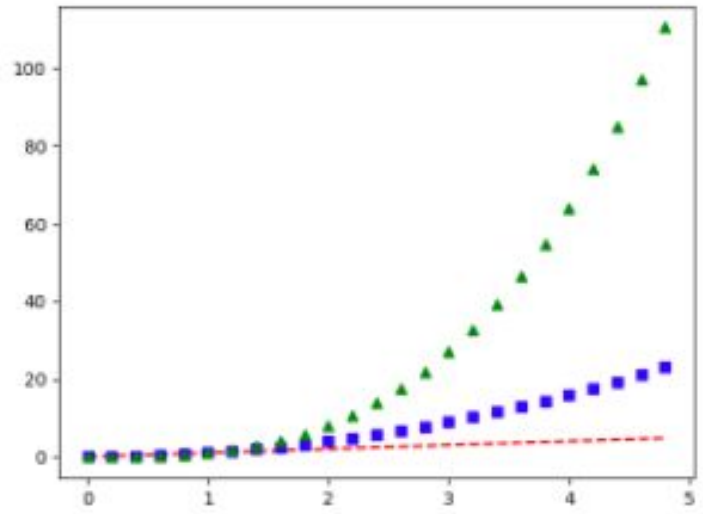
```
plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'ro')
plt.axis((0, 6, 0, 20)) # xmin, xmax, ymin, ymax
plt.show()
```




```
# example format strings
'b'    # blue markers with default shape
'ro'   # red circles
'g-'   # green solid line
'--'   # dashed line with default color
'k^:'  # black triangle_up markers connected by a dotted line
```

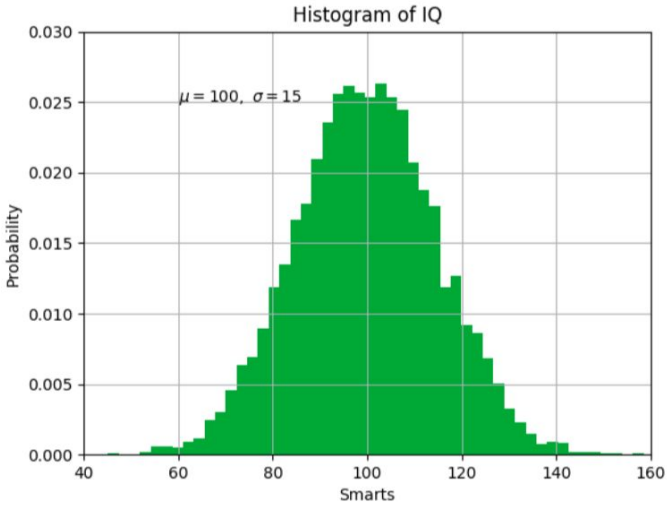
Plotting Numpy arrays

```
t = np.arange(0, 5, 0.2)

# red dashes, blue squares and green triangles
plt.plot(t, t, 'r--',
         t, t**2, 'bs',
         t, t**3, 'g^')
plt.show()
```



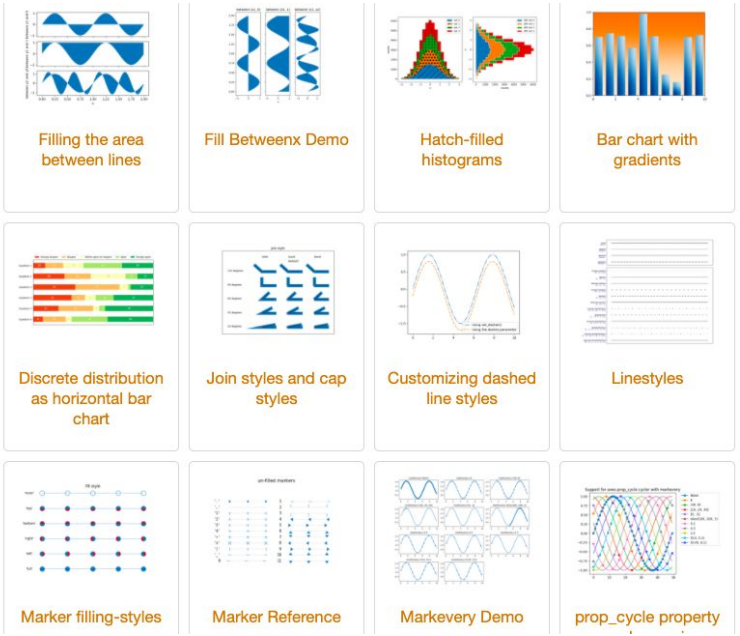
Histogram and text

```
mu, sigma = 100, 15
x = mu + sigma * np.random.randn(10000)

# histogram of data, with 50 “bins”
plt.hist(x, 50, facecolor='g')

plt.xlabel('Smarts')
plt.ylabel('Probability')
plt.title('Histogram of IQ')
plt.text(60, .025, r'$\mu=100,\ \sigma=15$')
plt.grid(True, alpha=0.75)
plt.show()
```



Matplotlib documentation

- Matplotlib provides thousands of possibilities and customizations
- Visit https://matplotlib.org/ and find the visualization tool you need!



Exercise

- Starting from “house.csv” dataset
  - Plot price column as y of a scatter plot with sqr_living as x
  - Plot price as y, yr_build as x
  - Ask the user the name of a variable and plot an histogram
- If available plot price VS sqr_total and check for the differences with the previous point