Quando il saggio indica il cielo, pandas guarda l'indice

Pietro Battiston

Firenze, 7 aprile 2017

PYCON OTTO

Prima di cominciare...

I file di dati che utilizzeremo si trovano all'indirizzo `http://pietrobattiston.it/c/pycon`

In `ipython`, la documentazione di qualsiasi funzione/metodo si può consultare con `funzione?` o `ogg.metodo?`.

Cos'è pandas?

- *La* libreria Python per manipolare ed analizzare dati

Cos'è pandas?

- *La* libreria Python per manipolare ed analizzare dati
- *Il* modo per i data analist Pythonisti di non sentirsi inferiori agli "**R**isti", con i loro data-frame (anzi!)

Cos'è pandas?

- *La* libreria Python per manipolare ed analizzare dati
- *Il* modo per i data analist Pythonisti di non sentirsi inferiori agli "**R**isti", con i loro data-frame (anzi!)
- In pratica, "semplicemente":
  - due strutture dati (`Series` e `DataFrame`) che estendono (pesantemente) gli `array` di `numpy`

Cos'è pandas?

- *La* libreria Python per manipolare ed analizzare dati
- *Il* modo per i data analist Pythonisti di non sentirsi inferiori agli "**R**isti", con i loro data-frame (anzi!)
- In pratica, "semplicemente":
  - due strutture dati (`Series` e `DataFrame`) che estendono (pesantemente) gli `array` di `numpy`
  - (parecchie) funzionalità aggiuntive (IO, datetime...)

Vi spiegherò tutto di `pandas`?

Vi spiegherò tutto di pandas?

No, non basterebbe l'intera PyCon

Vi spiegherò tutto di pandas?

No, non basterebbe l'intera PyCon

. . . quindi ci concentreremo sui concetti fondamentali.

Vi spiegherò tutto di pandas?

No, non basterebbe l'intera PyCon

. . . quindi ci concentreremo sui concetti fondamentali.

(**SPOT** Domani, ore 9:00: *How to use pandas the wrong way*)

. . . passo indietro: cos'è un `array`?

`numpy` fornisce a Python strutture di dati efficienti (RAM) con metodi efficienti (in C).

Es: `0.1 - numpy.ipynb`

. . . passo indietro: cos'è un `array`?

`numpy` fornisce a Python strutture di dati efficienti (RAM) con metodi efficienti (in C).

Es: `0.1 - numpy.ipynb`

Le operazioni possono essere vettoriali o *elementwise*.

Es: →**Elementwise**

. . . passo indietro: cos'è un `array`?

`numpy` fornisce a Python strutture di dati efficienti (RAM) con metodi efficienti (in C).

Es: `0.1 - numpy.ipynb`

Le operazioni possono essere vettoriali o *elementwise*.

Es: →**Elementwise**

. . . passo indietro: cos'è un `array`?

`numpy` fornisce a Python strutture di dati efficienti (RAM) con metodi efficienti (in C).

Es: `0.1 - numpy.ipynb`

Le operazioni possono essere vettoriali o *elementwise*.

Es: →**Elementwise**

Scopo del gioco: *vettorializzare* la manipolazione dei dati (incluse le operazioni elementwise).

Cos'è una `pandas.Series`?

È un `numpy.array` con

Cos'è una `pandas.Series`?

È un `numpy.array` con

- un *indice* (una lista di "etichette")

a	3
b	7
c	1
...	...

Cos'è una `pandas.Series`?

È un `numpy.array` con

- un *indice* (una lista di "etichette")

a	3
b	7
c	1
...	...

- ...circa 200 metodi in più.

Cos'è una `pandas.Series`?

È un `numpy.array` con

- un *indice* (una lista di "etichette")

a	3
b	7
c	1
...	...

- ...circa 200 metodi in più.

simile ad un `dict` molto potente.

Es: `1.1 - Series`

Indexing

Diversi meccanismi:

- `s.iloc[idx]`: *position-based* (come s.values[idx])

Indexing

Diversi meccanismi:

- `s.iloc[idx]`: *position-based* (come s.values[idx])
- `s.loc[lab]`: *label-based*

Indexing

Diversi meccanismi:

- `s.iloc[idx]`: *position-based* (come s.values[idx])
- `s.loc[lab]`: *label-based*
- `s[idx]`: alias (per ora!) di `s.loc[idx]`

Indexing

Diversi meccanismi:

- `s.iloc[idx]`: *position-based* (come s.values[idx])
- `s.loc[lab]`: *label-based*
- `s[idx]`: alias (per ora!) di `s.loc[idx]`
- (`s.ix[...]`: obsoleto)

Es: $\rightarrow$**Sfruttiamo l'indice**

Operazioni tra `Series`

. . . funzionano come tra `array`... ma "allineando" in base all'indice!

Es: `1.3 Operazioni tra Series`

Fate largo a `pd.DataFrame`!

Un `DataFrame` è una *tabella*

Fate largo a `pd.DataFrame`!

Un `DataFrame` è una *tabella*

	A	B
a	3	4
b	7	2
c	1	15
...	...	...

...ovvero una "lista" di `Series`

Fate largo a `pd.DataFrame`!

Un `DataFrame` è una *tabella*

	A	B
a	3	4
b	7	2
c	1	15
...	...	...

...ovvero una "lista" di `Series`

...con l'indice in comune.

Fate largo a `pd.DataFrame`!

Un `DataFrame` è una *tabella*

	A	B
a	3	4
b	7	2
c	1	15
...	...	...

...ovvero una "lista" di `Series`
...con l'indice in comune.

...ovvero un `array` **bi**dimensionale con **due** indici.

Es: `2.1 - DataFrame`

pandas e matplotlib

Sia `pd.Series` che `pd.DataFrame` hanno un metodo `.plot(...)`, che si appoggia su `matplotlib` (ed inoltra a `matplotlib` gli argomenti che non conosce).