

30 minuti con... Pandas



Corso di Quality Outsourcing Management
Lezione 1 di 3

Roberto Nai (Dipartimento di Informatica – UNITO)



Agenda

- Introduzione alla libreria Pandas
- Utilità della libreria Pandas
- Esempi pratici con Python in Google Colaboratory (Colab)
 - Leggere dati da file (CSV) tramite la libreria Pandas e creare un dataset (`DataFrame`)
 - Visualizzare i dati contenuti nel dataset
 - Panoramica dei dati contenuti nel dataset
- Conclusioni



Materiale della lezione: <https://github.com/roberto-nai/SUISS>



Pandas, chi era costui?

- **Pandas** è una *libreria* di Python utilizzata per lavorare con i **big data** (organizzati in **dataset**).
- Il nome "Pandas" fa riferimento sia a "Panel Data" che a "Python Data Analysis" ed è stato creato da Wes McKinney nel 2008.



<https://pandas.pydata.org>

Pandas, perché?

- Pandas permette di *analizzare* i dataset e trarre conclusioni basate su teorie *statistiche*.
- Pandas può inoltre *ripulire* i dataset disordinati (*rumorosi*) e renderli *leggibili e rilevanti*.
- I dati rilevanti sono molto importanti in **Data Science**.



Data Science: è una branca dell'Informatica in cui si studia come memorizzare, utilizzare e analizzare i dati per ricavarne informazioni.

Pandas, cosa fai?

- Pandas fornisce *risposte* sui dati. Ad esempio:
 - Esiste una correlazione tra due o più colonne?
 - Qual è il valore medio?
 - Qual è il valore massimo?
 - Qual è il valore minimo?
 - Qual è la moda?

Pandas, cosa fai?

- Pandas è anche in grado di *eliminare* o *correggere* le righe che non sono rilevanti o che contengono valori errati, come valori mancanti (detti *nulli* o NaN). Questa operazione si chiama **data cleaning**.



In Informatica, **NaN** significa *Not a Number* ed è un particolare valore che indica un dato *indefinito* (mancante) o *non rappresentabile* (es.: il risultato di 0 diviso 0).

Pandas, cosa leggi?

- Un modo semplice per archiviare grandi dataset è utilizzare i file CSV (Comma Separated Values).
- I file CSV contengono testo semplice e sono un formato ben noto che può essere letto da molte applicazioni (app), compreso Pandas.
- Nella prima lezione verrà utilizzato un file CSV chiamato "data.csv".



I file CSV possono essere letti e/o creati tramite varie applicazioni per ufficio tra cui Excel, Numbers, Calc, ecc. nonché dalle varie app che si utilizzano tutti i giorni.

Pandas, cosa leggi?

- Il file CSV chiamato "data.csv" preso in considerazione per questa lezione contiene i dati sulle pulsazioni e calorie consumate, registrati da uno smartwatch durante sessioni di corsa che hanno avuto una specifica durata (in minuti).
- I dati sono *separati* da una virgola (si possono utilizzare altri simboli).

```
data.csv x
1 durata,pulsazione,pulsazione_max,calorie
2 60,110,130,409.1
3 60,117,145,479.0
4 60,103,,340.0
5 45,109,175,282.4
6 45,117,148,406.0
```


Pandas, cosa leggi?

CSV

```
data.csv
1 durata,pulsazione,pulsazione_max,calorie
2 60,110,130,409.1
3 60,117,145,479.0
4 60,103,,340.0
5 45,109,175,282.4
6 45,117,148,406.0
```

Separatore ,

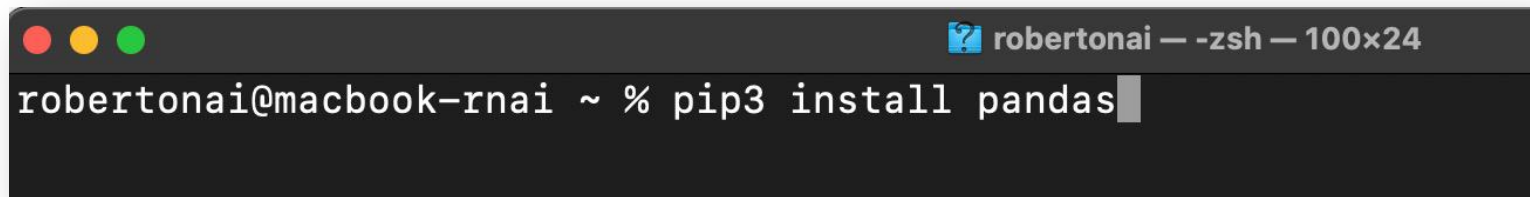


DataFrame

	durata	pulsazione	pulsazione_max	calorie
0	60	110	130	409.1
1	60	117	145	479.0
2	60	103	NaN	340.0
3

Pandas, iniziamo?

- Se sul proprio computer sono già installati Python (e PIP), l'installazione della libreria Pandas è molto semplice.
- Dal prompt dei comandi (o terminale), digitare
 - `pip install pandas` (in alternativa `pip3 install pandas`)



```
robertonai — -zsh — 100x24
robertonai@macbook-rnai ~ % pip3 install pandas
```



Python è scaricabile a questo indirizzo: <https://www.python.org/downloads/>
Per scrivere programmi in Python, si consigliano **Sublime Text** o **Visual Studio Code**.

Pandas, iniziamo?

- Una volta installata la libreria Pandas, importarla nelle applicazioni aggiungendo la parola chiave `import`:
 - `import pandas` (in alternativa `import pandas as pd`)



Alias: in Python gli alias sono un metodo alternativo per riferirsi alla stessa cosa, utilizzando la parola chiave `as`.

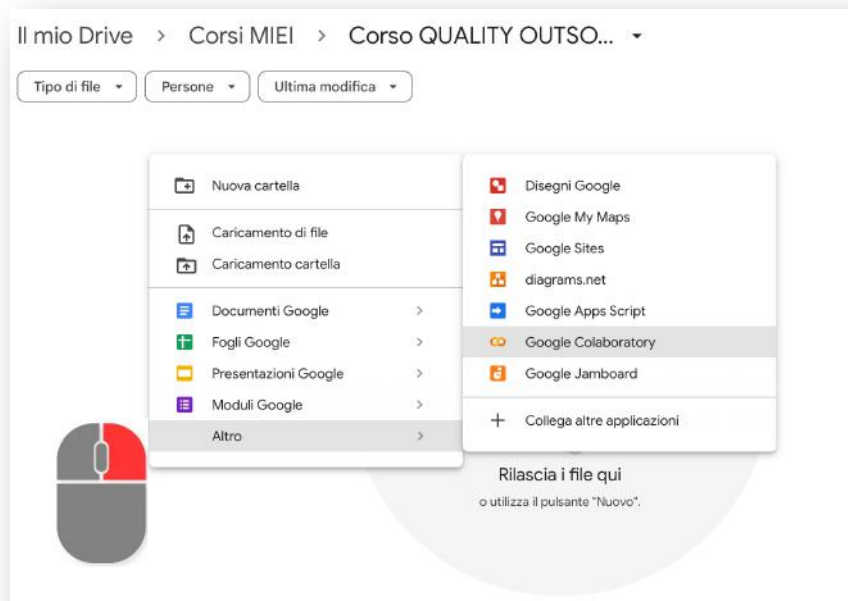
Pandas, iniziamo?

- In alternativa, Python e Pandas possono essere utilizzati tramite **Google Colaboratory** (detto anche **Colab**), che permette di scrivere ed eseguire *codice* (o *script*) Python nel browser senza alcuna configurazione necessaria.
 - <https://colab.research.google.com/>
 - È necessario avere un account Google in quanto **Colab salva il codice su Drive**.



Pandas, iniziamo?

- Dal proprio account di Google **Drive**, creare un nuovo file di Python tramite Colab.



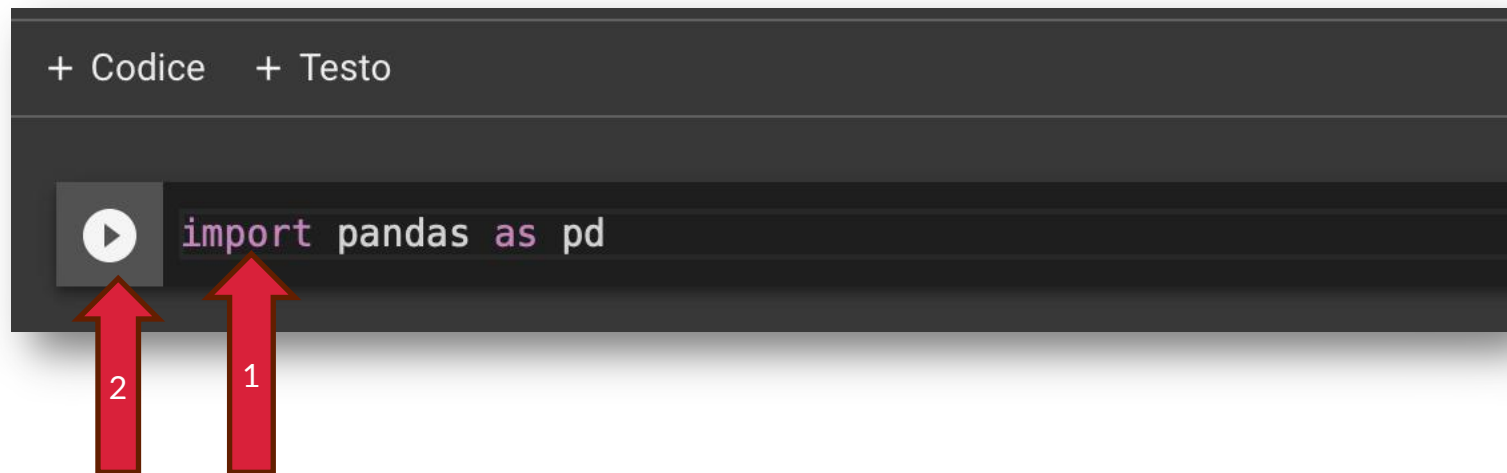
Pandas, iniziamo?

- Prime operazioni da eseguire:
 - 1) dare un nome al programma che conterrà il codice Python;
 - 2) caricare il file di dati (`data.csv`) nella cartella `sample_data`.



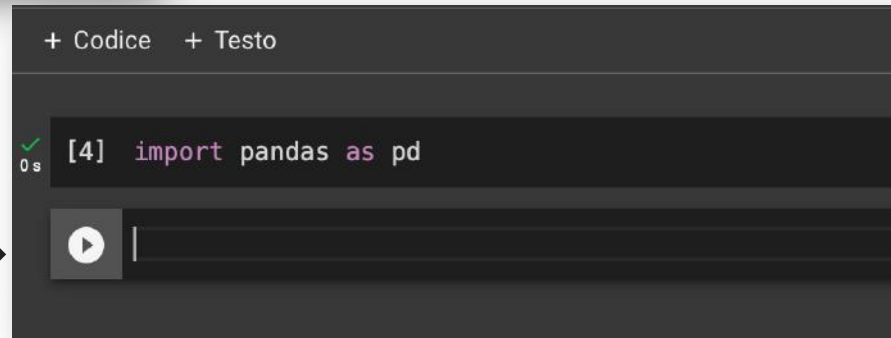
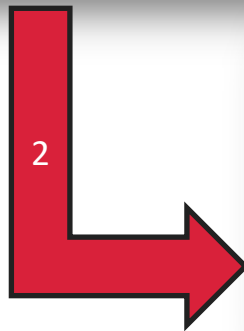
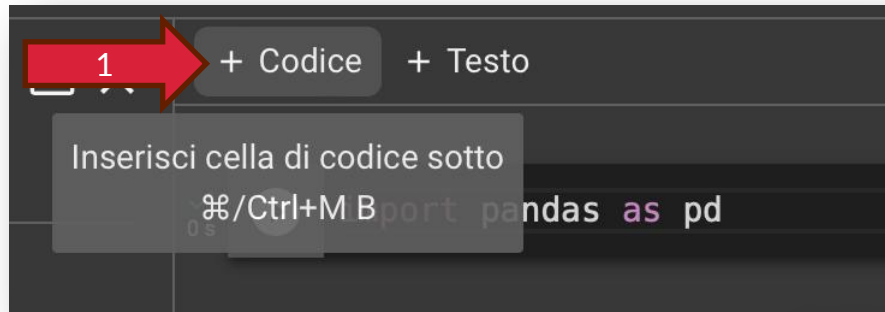
Pandas, iniziamo?

- Nello spazio per il codice Python, importare la libreria Pandas ed eseguire l'istruzione premendo il tasto *play* / *esegui cella*.



Pandas, iniziamo?

- Aggiungere una riga di codice premendo il pulsante + Codice



Pandas, leggi un CSV

- Il metodo `read_csv()` di Pandas permette di leggere un file CSV e salvarlo in memoria (nello specifico, viene letto il file e creato un oggetto di tipo `DataFrame`);

caso 1: `df = pd.read_csv('data.csv')`

caso 2: `df = pd.read_csv('sample_data/data.csv')`

caso 3: `df = pd.read_csv('sample_data/data.csv', sep = ';')`

- Il file CSV deve essere nella stessa cartella (caso 1) del file con il codice Python, oppure in una sotto-cartella (caso 2).
- Nel caso 3 viene utilizzato anche il parametro `sep` per indicare qual è il *separator* utilizzato nel CSV (';' o ', ', ecc.).

Pandas, leggi un CSV



Tutti i *metodi* di Pandas devono essere eseguiti dalla libreria (digitando `pd.metodo()`) o dal DataFrame (dataset) creato (digitando `df.metodo()`).



Tutti gli *attributi* di un DataFrame (dataset) si richiamano senza parentesi `()` (digitando `df.attributo`).

Pandas, leggi un CSV

CSV

```
data.csv
1 durata,pulsazione,pulsazione_max,calorie
2 60,110,130,409.1
3 60,117,145,479.0
4 60,103,,340.0
5 45,109,175,282.4
6 45,117,148,406.0
```

`df = pd.read_csv('data.csv')`

DataFrame
(df)

	[0]			[3]	
	durata	pulsazione	pulsazione_max	calorie	Colonne
0	60	110	130	409.1	
1	60	117	145	479.0	Riga
2	60	103	NaN	340.0	
3	

Dato nullo nella cella [2,2]

Pandas, leggi un CSV

- Leggere un file CSV per creare un `DataFrame` (dataset) chiamato `df`.

```
✓ [3] import pandas as pd
0 s

✓ # Legge il file 'data.csv' salvato in 'sample_data' e lo salva in df (DataFrame)
0 s df = pd.read_csv('sample_data/data.csv')
```



Commenti: in Python è possibile inserire commenti al proprio codice tramite il simbolo cancelletto `#`

Pandas, visualizza un DataFrame

- Dopo aver letto un file CSV ed averlo memorizzato in un `DataFrame`, uno dei metodi più utilizzati per ottenere una rapida panoramica dei dati è `head()`.
- Di default, il metodo `head()` mostra le *prime* cinque righe del dataset.
- È possibile modificare il numero di righe da mostrare definendolo all'interno delle parentesi del metodo `head()`;
 - `head(10)` mostra le prime dieci righe.

```
[13] # Mostra le prime cinque righe del DataFrame
      df.head()
```

Pandas, visualizza un DataFrame

```
[13] # Mostra le prime cinque righe del DataFrame
df.head()
```

	durata	pulsazione	pulsazione_max	calorie
0	60	110	130.0	409.1
1	60	117	145.0	479.0
2	60	103	NaN	340.0
3	45	109	175.0	282.4
4	45	117	148.0	406.0

```
[15] # Mostra le prime dieci righe del DataFrame
df.head(10)
```

	durata	pulsazione	pulsazione_max	calorie
0	60	110	130.0	409.1
1	60	117	145.0	479.0
2	60	103	NaN	340.0
3	45	109	175.0	282.4
4	45	117	148.0	406.0
5	60	102	127.0	300.0
6	60	110	136.0	374.0
7	45	104	134.0	253.3
8	30	109	133.0	195.1
9	60	98	124.0	269.0

Pandas, visualizza un DataFrame

- Come nel metodo `head()`, è possibile visualizzare le ultime righe di un dataset con il metodo `tail()`.
- Di default, il metodo `tail()` mostra le *ultime* cinque righe del dataset.
- È possibile modificare il numero di righe da mostrare definendolo all'interno delle parentesi del metodo `tail()`;
 - `tail(10)` mostra le ultime dieci righe.



```
# Mostra le ultime cinque righe del DataFrame  
df.tail()
```

Pandas, analizza un DataFrame

- Sul dataset creato è possibile applicare il metodo chiamato `info()`, che fornisce importanti informazioni sul dataset stesso (numero di righe, numero di colonne, campi vuoti, tipologia delle colonne, ecc.).
- Per eseguirlo, scrivere `df.info()`.

```
[8] # Mostra il numero di righe, colonne, la tipologia dei dati e il numero di dati mancanti  
df.info()
```


Pandas, analizza un DataFrame

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 169 entries, 0 to 168
Data columns (total 4 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   durata          169 non-null    int64
1   pulsazione      169 non-null    int64
2   pulsazione_max  168 non-null    float64
3   calorie         164 non-null    float64
dtypes: float64(2), int64(2)
memory usage: 5.4 KB
```

Il dataset ha
169 righe

Il dataset ha 4
colonne

Nomi delle colonne


Nella colonna `calorie` mancano cinque dati
(164 dati presenti su 169)

Nella colonna `pulsazioni_max` manca un
dato (168 dati presenti su 169)

Pandas, analizza un DataFrame

- Ogni dataset creato dispone di alcune *attributi* (proprietà), tra cui:
 - `shape` che mostra il numero di righe e colonne;
 - `shape[0]` che mostra il numero di righe;
 - `shape[1]` che mostra il numero di colonne;
 - `size` che mostra il numero di celle (righe x colonna);
 - `columns` che mostra i nomi delle colonne;
 - `index` che mostra l'intervallo delle righe presenti.

Pandas, analizza un DataFrame



```
# Mostra il numero di righe e colonne
```

```
df.shape
```


```
(169, 4)
```



```
# Mostra il numero di righe
```

```
df.shape[0]
```


```
169
```



```
# Mostra il numero di colonne
```

```
df.shape[1]
```


```
4
```



```
# Mostra il numero totale di celle (righe x colonne)
```

```
df.size
```

```
676
```



```
# Mostra la lista delle colonne
```

```
df.columns
```

```
Index(['durata', 'pulsazione', 'pulsazione_max', 'calorie'], dtype='object')
```



```
# Mostra le righe (indici) presenti
```

```
df.index
```

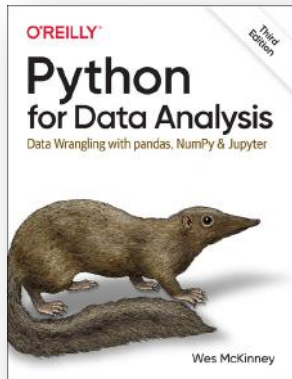
```
RangeIndex(start=0, stop=169, step=1)
```

Conclusioni

- In questa lezione si è visto come gestire un dataset in formato CSV tramite la libreria Pandas di Python.
- I dataset (detti `DataFrame` in Pandas), possono essere creati in Python partendo dal CSV corrispondente, tramite il metodo `read_csv()`.
- I dataset possono essere visualizzati tramite i metodi `head()` e `tail()`.
- I dataset, spesso, contengono dati mancanti; è possibile avere una visione dei valori *nulli* (o `NaN`) tramite il metodo `info()`, che mostra anche il numero di righe e colonne presenti nel dataset.

Bibliografia

- Python for Data Analysis: Data Wrangling With Pandas, Numpy, and Jupyter, 3a edizione, Wes McKinney, O'Reilly.



Bibliografia

- Pandas – Introduzione
 - <https://pandas.pydata.org/pandas-docs/stable/index.html>
- Pandas – Installazione
 - https://pandas.pydata.org/pandas-docs/stable/getting_started/index.html#getting-started
- Pandas – Leggere un file CSV
 - https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html
- Pandas – metodo `head()`
 - <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.head.html>
- Pandas – metodo `tail()`
 - <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.tail.html>
- Pandas – metodo `info()`
 - <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.info.html>

Bibliografia


- Pandas – attributo `shape`
 - <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.shape.html>
- Pandas – attributo `size`
 - <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.size.html>
- Pandas – attributo `columns`
 - <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.columns.html>
- Pandas – attributo `index`
 - <https://pandas.pydata.org/docs/reference/api/pandas.Index.html>

Bibliografia

- Python
 - <https://www.python.org>
- SublimeText (programma per sviluppare codice in Python)
 - <https://www.sublimetext.com>
- Visual Studio Code (programma per sviluppare codice in Python)
 - <https://code.visualstudio.com>
- Google Colaboratory (Colab):
 - <https://colab.research.google.com>

Fine presentazione

Grazie per l'attenzione

 roberto.nai@unito.it

