UNIVERSITÀ DI TORINO

30 minuti con… Pandas

Corso di **Quality Outsourcing Management**
Lezione 2 di 3

Roberto Nai (Dipartimento di Informatica – UNITO)

Agenda

- Introduzione al data cleaning
- Correggere le celle vuote
- Correggere le date
- Corregegre dati
- Conclusioni

Materiale della lezione: https://github.com/roberto-nai/SUISS_22-23

Pandas, dove eravamo rimasti?

- Nella lezione si è visto come gestire un **dataset** (insieme di dati) in formato CSV (Comma-Separated Value) 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.

https://pandas.pydata.org

Pandas, puliamo i dati?

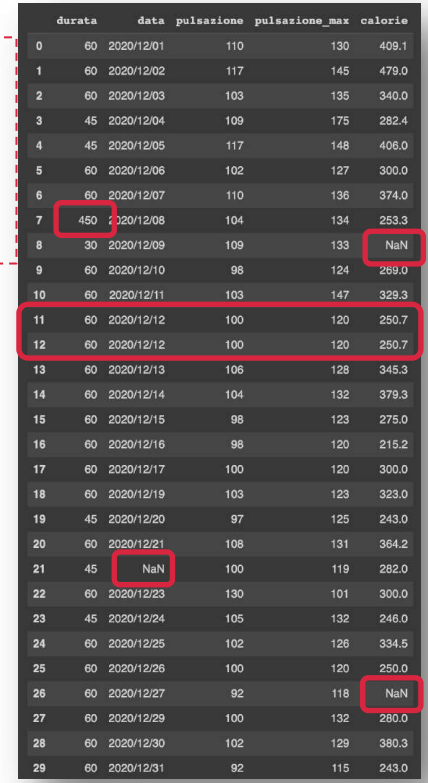
- L'attività di **data cleaning** (*pulizia dei dati*) significa correggere i dati errati presenti nel proprio dataset.
- I dati errati (*rumorosi*) potrebbero essere:
  - celle vuote
  - dati in formato errato
  - dati errati
  - duplicati

Pandas, puliamo i dati?

- Le celle con dati di formato errato (*rumoroso*) possono rendere difficile, o addirittura impossibile, l'analisi dei dati.

Dataset contenenti dati rumorosi

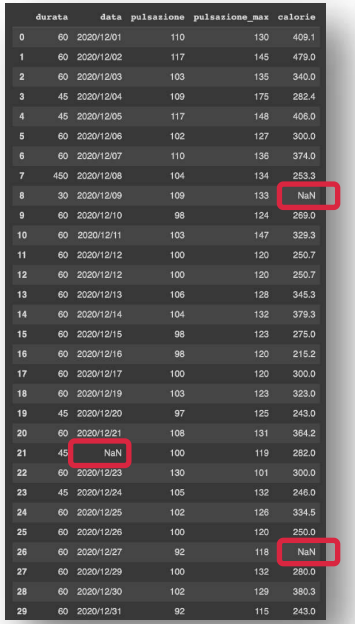
Sono presenti:
- dati mancanti (righe 8, 21, 26);
- dati duplicati (righe 11 e 12);
- dati fuori scala (riga 7).



	durata	data	pulsazione	pulsazione_max	calorie
0	60	2020/12/01	110	130	409.1
1	60	2020/12/02	117	145	479.0
2	60	2020/12/03	103	135	340.0
3	45	2020/12/04	109	175	282.4
4	45	2020/12/05	117	148	406.0
5	60	2020/12/06	102	127	300.0
6	60	2020/12/07	110	136	374.0
7	450	2020/12/08	104	134	253.3
8	30	2020/12/09	109	133	NaN
9	60	2020/12/10	98	124	269.0
10	60	2020/12/11	103	147	329.3
11	60	2020/12/12	100	120	250.7
12	60	2020/12/12	100	120	250.7
13	60	2020/12/13	106	128	345.3
14	60	2020/12/14	104	132	379.3
15	60	2020/12/15	98	123	275.0
16	60	2020/12/16	98	120	215.2
17	60	2020/12/17	100	120	300.0
18	60	2020/12/19	103	123	323.0
19	45	2020/12/20	97	125	243.0
20	60	2020/12/21	108	131	364.2
21	45	NaN	100	119	282.0
22	60	2020/12/23	130	101	300.0
23	45	2020/12/24	105	132	246.0
24	60	2020/12/25	102	126	334.5
25	60	2020/12/26	100	120	250.0
26	60	2020/12/27	92	118	NaN
27	60	2020/12/29	100	132	280.0
28	60	2020/12/30	102	129	380.3
29	60	2020/12/31	92	115	243.0

Pandas, sistema le celle vuote

- Le celle vuote (*nulle* o `NaN`) possono potenzialmente fornire un risultato sbagliato quando si analizzano i dati.



	durata	data	pulsazione	pulsazione_max	calorie
0	60	2020/12/01	110	130	409.1
1	60	2020/12/02	117	145	479.0
2	60	2020/12/03	103	135	340.0
3	45	2020/12/04	109	175	282.4
4	45	2020/12/05	117	148	406.0
5	60	2020/12/06	102	127	300.0
6	60	2020/12/07	110	136	374.0
7	450	2020/12/08	104	134	253.3
8	30	2020/12/09	109	133	NaN
9	60	2020/12/10	98	124	269.0
10	60	2020/12/11	103	147	329.3
11	60	2020/12/12	100	120	250.7
12	60	2020/12/12	100	120	250.7
13	60	2020/12/13	106	128	345.3
14	60	2020/12/14	104	132	379.3
15	60	2020/12/15	98	123	275.0
16	60	2020/12/16	98	120	215.2
17	60	2020/12/17	100	120	300.0
18	60	2020/12/19	103	123	323.0
19	45	2020/12/20	97	125	243.0
20	60	2020/12/21	108	131	364.2
21	45	NaN	100	119	282.0
22	60	2020/12/23	130	101	300.0
23	45	2020/12/24	105	132	246.0
24	60	2020/12/25	102	126	334.5
25	60	2020/12/26	100	120	250.0
26	60	2020/12/27	92	118	NaN
27	60	2020/12/29	100	132	280.0
28	60	2020/12/30	102	129	380.3
29	60	2020/12/31	92	115	243.0

Pandas, sistema le celle vuote

- Un modo per gestire le celle vuote consiste nel *rimuovere le righe che le contengono*.
- Di solito questa soluzione va bene quando i dataset sono molto grandi e la rimozione di alcune righe non ha impatto sul risultato.
- Il metodo `dropna()`, applicato al DataFrame, cancella le righe contenenti celle vuote e restituisce un nuovo DataFrame senza modificare l'originale.
  - `df_new = df.dropna()`

Pandas, sistema le celle vuote

```
# 1) Cancella le righe che contengono celle vuote e restituisce un nuovo DataFrame
df_2 = df.dropna()

# 2) Cancella le righe che contengono celle vuote direttamente nel DataFrame
df.dropna(inplace = True)
```

Pandas, sistema le celle vuote

	durata	data	pulsazione	pulsazione_max	calorie
0	60	2020/12/01	110	130	409.1
1	60	2020/12/02	117	145	479.0
2	60	2020/12/03	103	135	340.0
3	45	2020/12/04	109	175	282.4
4	45	2020/12/05	117	148	406.0
5	60	2020/12/06	102	127	300.0
6	60	2020/12/07	110	136	374.0
7	450	2020/12/08	104	134	253.3
8	30	2020/12/09	109	133	NaN
9	60	2020/12/10	98	124	269.0
10	60	2020/12/11	103	147	329.3
11	60	2020/12/12	100	120	250.7
12	60	2020/12/12	100	120	250.7
13	60	2020/12/13	106	128	345.3
14	60	2020/12/14	104	132	379.3
15	60	2020/12/15	98	123	275.0
16	60	2020/12/16	98	120	215.2
17	60	2020/12/17	100	120	300.0
18	60	2020/12/19	103	123	323.0
19	45	2020/12/20	97	125	243.0
20	60	2020/12/21	108	131	364.2
21	45	NaN	100	119	282.0
22	60	2020/12/23	130	101	300.0
23	45	2020/12/24	105	132	246.0
24	60	2020/12/25	102	126	334.5
25	60	2020/12/26	100	120	250.0
26	60	2020/12/27	92	118	NaN
27	60	2020/12/29	100	132	280.0
28	60	2020/12/30	102	129	380.3
29	60	2020/12/31	92	115	243.0

```
df.dropna()
```

	durata	data	pulsazione	pulsazione_max	calorie
0	60	2020/12/01	110	130	409.1
1	60	2020/12/02	117	145	479.0
2	60	2020/12/03	103	135	340.0
3	45	2020/12/04	109	175	282.4
4	45	2020/12/05	117	148	406.0
5	60	2020/12/06	102	127	300.0
6	60	2020/12/07	110	136	374.0
7	450	2020/12/08	104	134	253.3
9	60	2020/12/10	98	124	269.0
10	60	2020/12/11	103	147	329.3
11	60	2020/12/12	100	120	250.7
12	60	2020/12/12	100	120	250.7
13	60	2020/12/13	106	128	345.3
14	60	2020/12/14	104	132	379.3
15	60	2020/12/15	98	123	275.0
16	60	2020/12/16	98	120	215.2
17	60	2020/12/17	100	120	300.0
18	60	2020/12/19	103	123	323.0
19	45	2020/12/20	97	125	243.0
20	60	2020/12/21	108	131	364.2
22	60	2020/12/23	130	101	300.0
23	45	2020/12/24	105	132	246.0
24	60	2020/12/25	102	126	334.5
25	60	2020/12/26	100	120	250.0
27	60	2020/12/29	100	132	280.0
28	60	2020/12/30	102	129	380.3
29	60	2020/12/31	92	115	243.0

Pandas, sistema le celle vuote

- Un altro metodo comune per sostituire le celle vuote è quello di:
  - **calcolare** il valore *medio*, *mediano* o di *moda* della colonna;
  - **sostituire** le celle vuote con tale valore.

Media: il valore medio (la somma di tutti i valori divisa per il numero di valori).
Mediana: il valore al centro, dopo aver ordinato tutti i valori in modo crescente.
Moda: il valore che compare più frequentemente.

Pandas, sistema le celle vuote

- Il valore *medio* si calcola applicando il metodo `mean()` alla colonna.
- Esempio:
  - `media = df['calorie'].mean()`

L'accesso ad una colonna si ottiene inserendo il nome della colonna tra `['...']`

Pandas, sistema le celle vuote

- Il valore *mediano* si calcola applicando il metodo `median()` alla colonna.
- Esempio:
  - `mediana = df['calorie'].median()`

L'accesso ad una colonna si ottiene inserendo il nome della colonna tra `['...']`

Pandas, sistema le celle vuote

- Il valore della *moda* si calcola applicando il metodo `mode()` alla colonna.
- Esempio:
  - `moda = df['calorie'].mode()[0]`

L'accesso ad una colonna si ottiene inserendo il nome della colonna tra `['...']`

Pandas, sistema le celle vuote

```
[38] # Calcola la media dei valori della colonna 'calorie'
     media = df['calorie'].mean()
     media

     310.86785714285713
```

```
# Calcola la mediana dei valori della colonna 'calorie'
mediana = df['calorie'].median()
mediana

300.0
```

```
# Calcola la moda dei valori della colonna 'calorie'
moda = df['calorie'].mode()[0]
moda

300.0
```

Pandas, sistema le celle vuote

- Una volta deciso il valore che si desidera utilizzare per le celle vuote (media, mediana o moda), si applica alla colonna del dataset il metodo `fillna()` che consente di sostituire le celle vuote con un determinato valore.
  - `df['calorie'].fillna(media, inplace = True)`

Modifica direttamente il dataset senza crearne una copia.

Pandas, sistema le celle vuote

```
# Sostituise le celle vuote della colonna 'calorie' con il valore contenuto in media
df['calorie'].fillna(media, inplace = True)
```

Pandas, sistema le date

- Per correggere le colonne con le **date**, è necessario applicare due metodi:
  - cercare di correggere le date;
  - eliminare le righe senza date.

Pandas, sistema le date

	durata	data	pulsazione	pulsazione_max	calorie
0	60	2020/12/01	110	130	409.1
1	60	2020/12/02	117	145	479.0
2	60	2020/12/03	103	135	340.0
3	45	20201204	109	175	282.4
4	45	2020/12/05	117	148	406.0
5	60	2020/12/06	102	127	300.0
6	60	2020/12/07	110	136	374.0
7	450	2020/12/08	104	134	253.3
8	30	2020/12/09	109	133	NaN
9	60	2020/12/10	98	124	269.0
10	60	2020/12/11	103	147	329.3
11	60	2020/12/12	100	120	250.7
12	60	2020/12/12	100	120	250.7
13	60	2020/12/13	106	128	345.3
14	60	2020/12/14	104	132	379.3
15	60	2020/12/15	98	123	275.0
16	60	2020/12/16	98	120	215.2
17	60	2020/12/17	100	120	300.0
18	60	2020/12/19	103	123	323.0
19	45	2020/12/20	97	125	243.0
20	60	2020/12/21	108	131	364.2
21	45	NaN	100	119	282.0

Pandas, sistema le date

- Per cercare di correggere le date, è possibile applicare il metodo della libreria Pandas chiamato `to_datetime()` che riceve come parametro tra le parentesi la colonna da correggere.
  - `df['data'] = pd.to_datetime(df['data'])`

Pandas, sistema le date

```
# Cerca di correggere la colonna 'data'
df['data'] = pd.to_datetime(df['data'])
```

	durata	data	pulsazione	pulsazione_max	calorie
0	60	2020/12/01	110	130	409.1
1	60	2020/12/02	117	145	479.0
2	60	2020/12/03	103	135	340.0
3	45	20201204	109	175	282.4
4	45	2020/12/05	117	148	406.0

	durata	data	pulsazione	pulsazione_max	calorie
0	60	2020-12-01	110	130	409.1
1	60	2020-12-02	117	145	479.0
2	60	2020-12-03	103	135	340.0
3	45	2020-12-04	109	175	282.4
4	45	2020-12-05	117	148	406.0

Pandas, sistema le date

- Le date che non possono essere corrette, vengono segnalate come celle contenenti il valore `NaT` (Not a Time).

20	60	2020-12-21	108	131	364.2
21	45	NaT	100	119	282.0
22	60	2020-12-23	130	101	300.0
23	45	2020-12-24	105	132	246.0

Pandas, sistema le date

- Le righe con date mancanti possono essere rimosse, utilizzando il metodo già visto `dropna()` ma, anziché applicarlo su tutte le righe, è possibile indicare al metodo di cancellare solamente le righe con date mancanti (ovvero si applica il metodo solo ai dati mancanti in una determinata colonna).
- `df.dropna(subset=['data'], inplace = True)`

Pandas, sistema le date

```
# Cancella le righe con date mancanti (NaT)
df.dropna(subset = ['data'], inplace = True)
```

Cancella le righe vuote solo della colonna `data`.

Pandas, sistema i dati errati

- I *dati errati* non devono essere necessariamente *celle vuote* o *formato errato* (come le date), ma possono essere semplicemente errati, come se qualcuno avesse inserito il valore "199" invece di "1.99" oppure "450" anziché "45".
- A volte è possibile individuare i dati errati osservando il dataset, avendo un'aspettativa di ciò che dovrebbe contenere.

Pandas, sistema i dati errati

- Esempio: se si osserva il dataset di esempio, si può notare che nella riga 7 la durata è di 450, ma per tutte le altre righe la durata è compresa tra 30 e 60.

	durata	data	pulsazione	pulsazione_max	calorie
0	60	2020/12/01	110	130	409.1
1	60	2020/12/02	117	145	479.0
2	60	2020/12/03	103	135	340.0
3	45	2020/12/04	109	175	282.4
4	45	2020/12/05	117	148	406.0
5	60	2020/12/06	102	127	300.0
6	60	2020/12/07	110	136	374.0
7	450	2020/12/08	104	134	253.3
8	30	2020/12/09	109	133	NaN
9	60	2020/12/10	98	124	269.0

Pandas, sistema i dati errati

- Un metodo per correggere i valori errati è sostituirli con altri.
  - Nell'esempio precedente, è molto probabile che si tratti di un errore di battitura e che il valore debba essere "45" invece di "450".
- Attraverso il metodo `loc`, è possibile accedere ad una riga e colonna e modificarne il contenuto.
  - `df.loc[7, 'durata'] = 45`

Pandas, sistema i dati errati

```
# Modifica la colonna 'durata' alla riga 7
df.loc[7, 'durata'] = 45
```

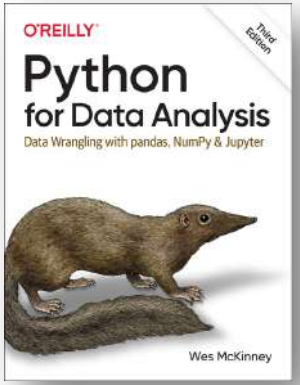
	durata	data	pulsazione	pulsazione_max	calorie
0	60	2020/12/01	110	130	409.1
1	60	2020/12/02	117	145	479.0
2	60	2020/12/03	103	135	340.0
3	45	2020/12/04	109	175	282.4
4	45	2020/12/05	117	148	406.0
5	60	2020/12/06	102	127	300.0
6	60	2020/12/07	110	136	374.0
7	450	2020/12/08	104	134	253.3
8	30	2020/12/09	109	133	NaN
9	60	2020/12/10	98	124	269.0

Conclusioni

- In questa lezione è stato affrontato il *data cleaning* di un 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 – metodo `dropna()`
  - https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.html
- Pandas – metodo `mean()`
  - https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.mean.html
- Pandas – metodo `median()`
  - https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.median.html
- Pandas – metodo `mode()`
  - https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.mode.html
- Pandas – metodo `fillna()`
  - https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.fillna.html

Bibliografia

- Pandas – metodo `to_datetime()`
  - https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.to_datetime.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