FONDI STRUTTURALI E DI INVESTIMENTO EUROPEI 2014/2020

fondo sociale europeo

Elementi di Programmazione con Python e Analisi dei Dati
Lezione 3: Pandas

Stefano Andreozzi, PhD

Formazione continua individuale – Id Attività: 2530775 Codice Corso: B341-1-2019-0

30 novembre 2020

UNIONE EUROPEA

REGIONE PIEMONTE

per una crescita intelligente, sostenibile ed inclusiva

www.regione.piemonte.it/europa2020

INIZIATIVA CO-FINANZIATA CON FSE

- Pandas: libreria per la manipolazione di dati in formato sequenziale o tabellare
- Caratteristiche principali
  - Caricamento e salvataggio di formati standard per dati tabellari, quali CSV (Comma-separated Values), TSV (Tab-separated Values), file Excel e formati per database
  - Semplicità nella esecuzione di operazioni di indicizzazione e aggregazione di dati
  - Semplicità nella esecuzione di operazioni numeriche e statistiche
  - Semplicità nella visualizzazione dei risultati delle operazioni
- sito ufficiale del progetto: info aggiuntive, documentazione etc.
- sintassi comune `import pandas as pd`
- tutorial: https://www.learndatasci.com/tutorials/python-pandas-tutorial-complete-introduction-for-beginners/
- repository del tutorial

Per ricapitolare

- obiettivo: analizzare un dataset CSV con l'aiuto della libreria Pandas in python
  - quali sono le evidenze estratte dai dati?
  - tenere in considerazione la distribuzione delle variabili e le informazioni contestuali
  - dati spesso sparsi e sporchi
  - importanza della compattazione e pulizia
- csv semplice tabella di dati, con un record per riga e una variabile per colonna
- terminologia di pandas: la tabella è chiamata Dataframe e ognuna delle colonne è detta Series

- open dataset per esplorarne i contenuti
- csv sui posti letto negli ospedali italiani preso da dati.salute.gov.it
- È un buon dataset per due motivi:
  - contiene informazioni concrete sulla sanità nazionale (dati provenienti dal mondo reale)
  - dati disordinati, incompleti e non abbastanza documentati

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3 import requests
4 url = 'http://www.dati.salute.gov.it/imgs/C_17_dataset_96_0_upFile.csv'
5 r = requests.get(url, allow_redirects=True)
6 open('C_17_dataset_96_0_upFile.csv', 'wb').write(r.content)
7 csv=pd.read_csv('C_17_dataset_96_0_upFile.csv', sep=';')
```

- l'istruzione `pd.read_csv()` dà un errore `UnicodeDecodeError: 'utf-8' codec can't decode byte 0xb0 in position 1: invalid start byte`
- che significa?

Scelta del dataset II

- i file di testo non sono tutti uguali
- *non coperto dal corso:* approfondimenti nella documentazione
  - Standard encodings
  - Unicode HOWTO
- il più delle volte pandas si aspetta Unicode Transformation Format a 8 bit (UTF-8)
- si usa l'opzione `encoding= 'unicode_escape'`

```
import pandas as pd
import matplotlib.pyplot as plt
import requests
url = 'http://www.dati.salute.gov.it/imgs/C_17_dataset_96_0_upFile.csv'
r = requests.get(url, allow_redirects=True)
open('C_17_dataset_96_0_upFile.csv', 'wb').write(r.content)
csv=pd.read_csv('C_17_dataset_96_0_upFile.csv', sep=';',encoding='unicode_escape')
```

Ispezione di dati e tipo

- lanciamo le istruzioni `print(csv.head())` e `print(csv.dtypes)`
- otteniamo

```
   Anno  Codice Regione Descrizione Regione  ...  Posti letto Day Hospital  Posti letto Day Surgery  Totale posti l
0  2010              10            PIEMONTE  ...                        15                        0
1  2010              10            PIEMONTE  ...                         0                        0
2  2010              10            PIEMONTE  ...                         0                        0
3  2010              10            PIEMONTE  ...                         0                        0
4  2010              10            PIEMONTE  ...                         0                        0
```

```
Anno                                    int64
Codice Regione                          int64
Descrizione Regione                    object
Codice Azienda                          int64
Tipo Azienda                            int64
Codice struttura                        int64
Subcodice                              object
Denominazione Struttura/Stabilimento   object
Indirizzo                              object
Codice Comune                           int64
Comune                                 object
Sigla Provincia                        object
Codice tipo struttura                 float64
Descrizione tipo struttura             object
Codice disciplina                       int64
Descrizione disciplina                 object
Tipo di Disciplina                     object
N° Reparti                              int64
Posti letto degenza ordinaria           int64
Posti letto degenza a pagamento         int64
Posti letto Day Hospital                int64
Posti letto Day Surgery                 int64
Totale posti letto                      int64
dtype: object
```

Controllo del tipo

- importante controllare che il tipo di dato sia quello che ci si aspetta per ogni campo
- specialmente in versioni legacy di Pandas alcune colonne numeriche potrebbero essere scambiate per testuali (`object`)
- perché valori come N.D. in colonne che dovrebbero contenere solo numeri

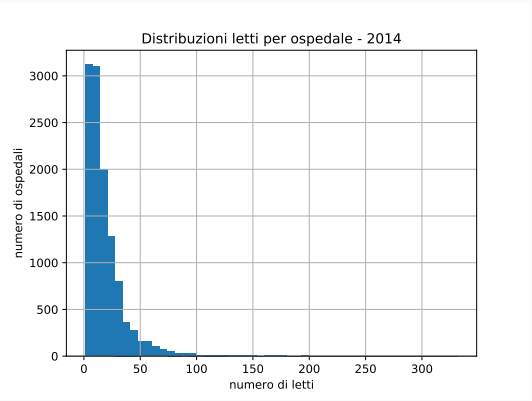
Prime analisi:

- Per adesso ci limitiamo all'anno 2014
- per iniziare ci concentriamo sulla variabile "Totale posti letto" (numero totale di letti nell'ospedale)
- utilizziamo la funzione `describe()` per ottenere delle statistiche descrittive
- costruiamo un istogramma per vedere come la variabile è distribuita.

```
csv2014 = csv[ csv['Anno'] == 2014 ]
beds2014 = csv2014['Totale posti letto']
print(beds2014.describe())
histogram = beds2014.hist( bins=50 )
histogram.set_title( 'Distribuzioni letti per ospedale - 2014' )
histogram.set_xlabel( 'numero di letti' )
histogram.set_ylabel( 'numero di ospedali' )
plt.show()
```

Risultati

```
count    11696.000000
mean        18.859268
std         21.572361
min          1.000000
25%          7.000000
50%         13.000000
75%         23.000000
max        332.000000
```

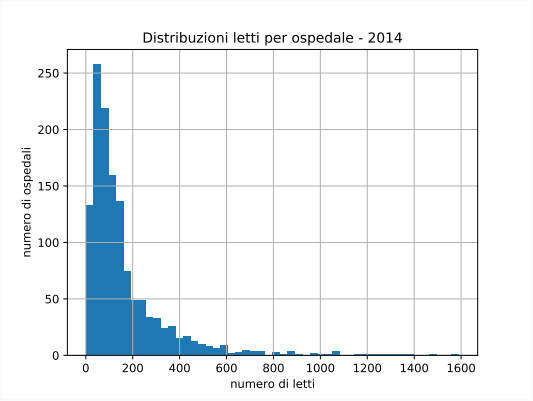


- c'è qualcosa che non va
- chi mi sa spiegare?

- abbiamo visualizzato i posti per reparto e non per ospedale!!!
- usiamo `groupby`

```
csv2014 = csv[ csv['Anno'] == 2014 ]
beds2014= csv2014.groupby('Denominazione Struttura/Stabilimento')['Totale posti letto'].
    sum()
print(beds2014.describe())
histogram = beds2014.hist( bins=50 )
histogram.set_title( 'Distribuzioni letti per ospedale - 2014' )
histogram.set_xlabel( 'numero di letti' )
histogram.set_ylabel( 'numero di ospedali' )
plt.show()
```

count	1320.000000
mean	167.104545
std	191.253464
min	1.000000
25%	56.750000
50%	104.500000
75%	198.000000
max	1591.000000



- Il sistema ospedaliero italiano è quindi composto di pochi grossi ospedali e molti piccoli ospedali
- Questo rispecchia la presenza di grandi città in Italia, nelle quali la densità abitativa è maggiore e così la necessità di ospedali capienti, mentre nelle tante zone meno popolate ci sono altrettante strutture più piccole.
- Media e varianza di una variabile possono dare falsi indizi, è buona pratica guardare la distribuzione.

Letti di ospedale: sorting

- regione con più ospedali?
- `beds2014= csv2014.groupby('Descrizione Regione')['Totale posti letto'].sum()`
- ottengo un oggetto di tipo `Series`
- ripristino un `DataFrame`: `beds2014_2=beds2014.to_frame().reset_index()`
- ordino dalla regione con più posti letto a quella con meno posti letto: `beds2014_2=beds2014_2.sort_values(beds2014_2.columns[1], ascending=False)`

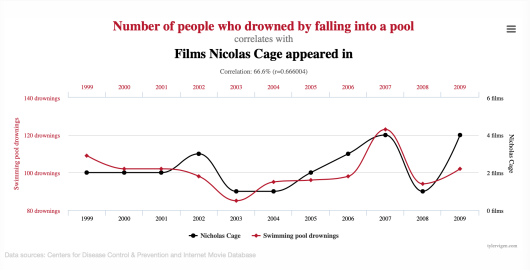
	Descrizione Regione	Totale posti letto
8	LOMBARDIA	38519
6	LAZIO	21955
4	EMILIA ROMAGNA	18876
3	CAMPANIA	18504
20	VENETO	18400
11	PIEMONTE	17703
16	SICILIA	16689
14	PUGLIA	13046
17	TOSCANA	12619
15	SARDEGNA	5924
2	CALABRIA	5857
7	LIGURIA	5833
9	MARCHE	5772
5	FRIULI VENEZIA GIULIA	5028
0	ABRUZZO	4483
18	UMBRIA	3164
13	PROV. AUTON. TRENTO	2132
12	PROV. AUTON. BOLZANO	2106
1	BASILICATA	2029
10	MOLISE	1422
19	VALLE D'AOSTA	517

Per approfondire

- **proposte:**
  - FACILE: plottare la distribuzione della variabile “Posti letto Day Hospital”
  - MEDIO: raggruppare i “Posti letto Day Hospital” in base al “Comune”.
  - DIFFICILE: trovare gli ospedali più capienti in base al “Tipo di Disciplina”
- tutorial
- altri approfondimenti su Pandas
- Summarising, Aggregating, and Grouping data in Python Pandas
- gestire Excel su Python, 1: xlwt
- gestire Excel su Python, 2: openpyxl
- gestire Excel su Python, 3: XlsxWriter
- **avanzato:** parallelizzazione dei DataFrame e di altre strutture dati (es. numpy) con Dask

- http://data.dft.gov.uk/road-accidents-safety-data/Stats19-Data1979-2004.zip (242 Mb)
- estrarre incidenti avvenuti a Londra nel 2000
- provare ad aprire il file `Accidents7904.csv` in Excel (737 Mb)
- il vs computer verosimilmente si pianta oppure vi dice che non riesce a caricare tutti i record
- come fare?
- `pandas_accidents.py`

- motivazione: nell'analizzare un dataset (es. sistemi biologici) è possibile che solo poche colonne abbiano senso, le altre magari sono *combinazioni lineari* di altre colonne (variabili latenti)
- controllare se si correlano *linearmente* con qualche altra variabile
- $\rho_{X,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y}$ (coefficiente di Pearson, proporzionale alla variabilità *congiunta* delle due variabili aleatorie X e Y)
  - 0 nessuna correlazione, 1 correlazione massima (stessi valori)
  - correlazione negativa: se una var aumenta, l'altra diminuisce
  - Guess the correlation!
- correlazione non significa legame causa-effetto!



Esempio, giocatori NBA

```
# importing pandas as pd
import pandas as pd
# Making data frame from the csv file
df = pd.read_csv("nba.csv")
# Printing the first 10 rows of the data frame for visualization
df[:10]
```

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	NaN	12000000.0
6	Jordan Mickey	Boston Celtics	55.0	PF	21.0	6-8	235.0	LSU	1170960.0
7	Kelly Olynyk	Boston Celtics	41.0	C	25.0	7-0	238.0	Gonzaga	2165160.0
8	Terry Rozier	Boston Celtics	12.0	PG	22.0	6-2	190.0	Louisville	1824360.0
9	Marcus Smart	Boston Celtics	36.0	PG	22.0	6-4	220.0	Oklahoma State	3431040.0

tutorial

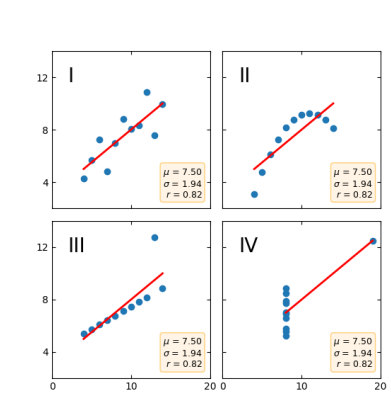
dataframe.corr() II

```
# To find the correlation among
# the columns using pearson method
df.corr(method ='pearson')
```

	Number	Age	Weight	Salary
Number	1.000000	0.028724	0.206921	-0.112386
Age	0.028724	1.000000	0.087183	0.213459
Weight	0.206921	0.087183	1.000000	0.138321
Salary	-0.112386	0.213459	0.138321	1.000000

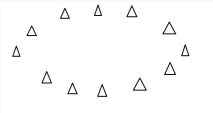
Che possiamo dire?

Intermezzo: Quartetto di Anscombe

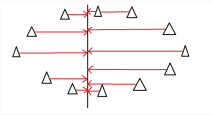


- Graphs in Statistical Analysis: “Graphs are essential to good statistical analysis”, The American Statistician, Vol. 27, No. 1. (Feb., 1973), pp. 17-21
- quattro set di dati con stesse statistiche descrittive, ma grafici totalmente differenti
- qui trovate il dataset e il codice

- **PCA** (Principal Component Analysis, Pearson e Hotelling, 1933): riduzione di numero di variabili (caratteristiche del fenomeno analizzato) in poche variabili latenti, che da sole sono considerate informative (feature reduction)

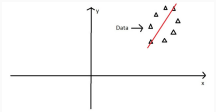


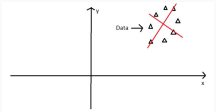
- Componente principale: trovare la retta che ha le proiezioni dei dati spalmati in maniera più ampia



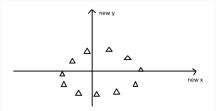
- la retta è detta **componente principale**: i dati proiettati su quella retta hanno una varianza maggiore
- in maniera sistematica: calcolo degli autovettori e autovalori
- autovettore: direzione scelta per disegnare uno degli assi con cui descrivere i dati
- autovalore: quanta varianza c'è lungo quella direzione

- componente principale: autovettore avente autovalore maggiore di tutti gli altri



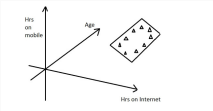


- numero di autovettori pari alle dimensioni del dataset originario, es. due dimensioni
- ogni autovettore deve essere perpendicolare agli altri (indipendenti fra loro)

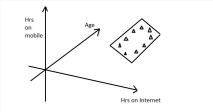


- riferire i dati in questi nuovi assi (prospettiva maggiormente informativa)

- eventuali assi con autovalori piccoli possono essere eliminati, cioè la dimensione viene scartata



- esempio tridimensionale: età, ore uso internet, ore uso smartphone



- dati raccolti giacciono come su un piano: PCA con terzo autovettore trascurabile
- riduzione da dimensione 3 a dimensione 2

- **Perché la PCA necessita eventualmente di una normalizzazione dei dati, quando dati differenti hanno scale differenti?**
- Una spiegazione un po' più rigorosa della PCA è qui
- Esempio con visualizzazione di dataset a dimensione elevata (1): qui
- Esempio con visualizzazione di dataset a dimensione elevata (2): qui

