Kung Fu Pandas

“La mente intelligente è una mente curiosa.”

-Bruce Lee

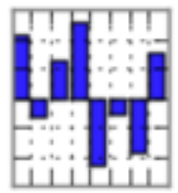
PYCONSEI

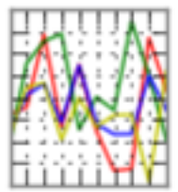
About me

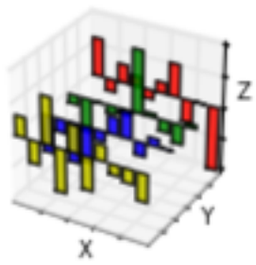
```
name = "Christian Barra"
location = "Often Milan or around the world"
blog = "chrisbarra.me"
github = "github.com/barrachri"
actually = "I'm studying statistics and
 looking for something interesting to do !"

print("Do you need some help ?")
print("{}@{}.com".format("barrachri", "gmail"))
```

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$







“**pandas** is an **open source**, **BSD-licensed library** providing **high-performance**, **easy-to-use data structures** and **data analysis tools** for **Python**.”

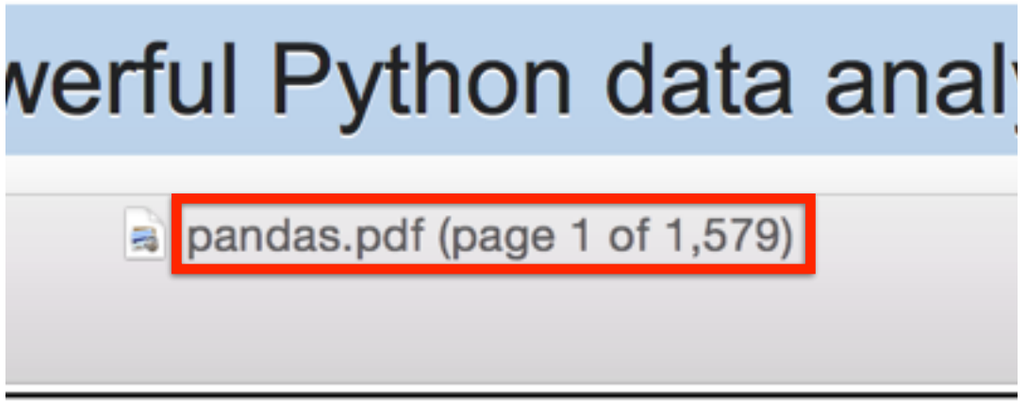
Pandas in 3 righe.

More info

- Si base sulla libreria **numpy**
- Il boss e' Wes McKinney
- pandas.pydata.org
- Compatibile con python 3
- Fortemente utilizzato in ambito finanziario

1579 pagine di docs.

Decisamente tanta documentazione.



ndas: powerful I

Per iniziare

- Tutto e' praticamente una matrice
- SERIES one-dimensional labeled array (tipo dati unico)
- DATAFRAME 2-dimensional labeled data structure (tipo dati misto)
- Il miglior modo per iniziare e' seguire il tutorial da 10 min disponibile su pandas.pydata.org

Quindi cosa facciamo adesso ?

Analizziamo dei dati con pandas + jupyter.

pandas + jupyter (ipython)

SuperPandas di 5th livello

	A	B	C	D	E	F
1	Anno	Mese	NaturaIncidente	Incidenti	Feriti	Morti
2	2001	1	Scontro frontale	32	49	1
3	2001	1	Scontro frontale-laterale	482	758	1
4	2001	1	Scontro laterale	66	75	1
5	2001	1	Tamponamento	151	227	0
6	2001	1	Investimento pedone	204	232	4
7	2001	1	Urto con veicolo in fermata o sosta	113	146	0
8	2001	1	Urto con ostacolo	86	99	1
9	2001	1	Fuoriuscita, sbandamento	62	67	0
10	2001	1	Altre cause	36	36	0
11	2001	2	Scontro frontale	32	52	0
12	2001	2	Scontro frontale-laterale	483	734	3
13	2001	2	Scontro laterale	107	127	0
14	2001	2	Tamponamento	137	193	1
15	2001	2	Investimento pedone	147	169	4
16	2001	2	Urto con veicolo in fermata o sosta	131	172	2

Ecco i nostri dati.

Quando si parla di Kung Fu purtroppo ci sono incidenti, feriti e morti.

About our data

- Open data del comune di Milano
- Formato CSV

- Dati storici (dal 2001 al 2013) del numero di incidenti, feriti e morti per ogni mese, suddivisi per tipologia.

“Se cerchi qualcosa senza sapere cosa cerchi perderai solo tempo. Parti dalle domande e poi cerchi le risposte nei tuoi dati.”

–Kung Fu Data Master

Le nostre domande

1. Qual e' l'incidente con la maggior frequenza ?
2. Qual e' la tipologia di incidente con la mortalita' piu' alta ?
3. Il numero di incidenti, feriti e morti e' diminuito durante gli anni ?
4. Esiste una qualche stagionalita' nel numero di incidenti ?
5. Come e' cambiato il numero e la composizione degli incidenti nel tempo ?
6. Esiste qualche correlazione tra la piovosita' e il numero di incidenti, feriti, morti nel 2013 ?

0# Importiamo Pandas

```
In [97]: import pandas as pd
         import numpy as np
         %matplotlib inline
```

- Avete installato pandas giusto ? (pip install pandas)

1# Importiamo i dati

```
#Importiamo i dati
datami = pd.read_csv("data/incidenti.csv", sep=";")
```

- read_csv accetta diversi parametri
- Nel nostro caso dove si trova il file e il separator dei campi del nostro file csv
- Ovviamente ci sono altri 1000 parametri.

2.1# Tempo di sbirciare

```
#Visualizziamo le prime 5 rows
datami.head()
```

	Anno	Mese	NaturaIncidente	Incidenti	Feriti	Morti
0	2001	1	Scontro frontale	32	49	1
1	2001	1	Scontro frontale-laterale	482	758	1
2	2001	1	Scontro laterale	66	75	1

- Cool.
- Con date.head(rows) pandas ti mostra le prime #rows
- Default sono 5 rows

2.2# Tempo di sbirciare

```
#Visualizziamo alcune info utili
datami.info()
```

- Restituisce alcune informazioni sul dataframe/series.
- Numero di rows, column, tipo dati e altro.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1404 entries, 0 to 1403
Data columns (total 6 columns):
Anno                1404 non-null int64
Mese                1404 non-null int64
NaturaIncidente     1404 non-null object
Incidenti           1404 non-null int64
Feriti              1404 non-null int64
Morti               1404 non-null int64
dtypes: int64(5), object(1)
memory usage: 76.8+ KB
```

1404 rows x 6 columns

2.3# Tempo di sbirciare

```
#Visualizziamo alcune info utili
datami.describe()
```

- Restituisce alcuni indicatori statistici calcoli sul nostro dataframe

- Media, Mediana, Min, Max e altro

	Anno	Mese	Incidenti	Feriti	Morti
count	1404.000000	1404.000000	1404.000000	1404.000000	1404.000000
mean	2007.000000	6.500000	129.482194	174.190171	0.609687
std	3.742991	3.453283	117.957941	177.141507	1.015948
min	2001.000000	1.000000	1.000000	1.000000	0.000000
25%	2004.000000	3.750000	53.000000	64.000000	0.000000
50%	2007.000000	6.500000	110.000000	134.500000	0.000000
75%	2010.000000	9.250000	156.000000	201.000000	1.000000
max	2013.000000	12.000000	701.000000	1044.000000	7.000000

Mediamente 6 morti ogni 1294 incidenti

3# Vogliamo solo il 2005

```
#Rows con Anno == 2005
datami[datami['Anno'] == 2005].head()
```

	Anno	Mese	NaturaIncidente	Incidenti	Feriti	Morti
432	2005	1	Scontro frontale	24	29	1
433	2005	1	Scontro frontale-laterale	399	634	1
434	2005	1	Scontro laterale	106	134	0

- In pratica indichiamo la colonna (date[“Anno”]) e specifichiamo la condizione (== 2005).

- Head ci serve solo per stampare le prime 5 rows.

4# Totali NaturaIncidente

```
#Raggruppiamo per natura incidente
data_summed = datami.groupby("NaturaIncidente").sum()
```

```
#Eliminiamo alcuni dati che non ci interessano
data_summed = data_summed.drop(["Anno","Mese"],1)
```

- all'interno di groupby indichiamo le columns che raggruppiamo
- sum() indica come vogliamo effettuare il raggruppamento.
- Naturalmente si possono specificare altre modalita', mean() per esempio.
- Droppiamo le columns che non ci interessano, 1 indica l'axis di riferimento.

	Incidenti	Feriti	Morti
NaturalIncidente			
Altre cause	4170	4622	7
Fuoriuscita, sbandamento	17933	19367	53
Investimento pedone	21783	25898	294
Scontro frontale	3370	5142	36
Scontro frontale-laterale	62842	93477	207
Scontro laterale	20219	23988	52
Tamponamento	21201	32929	24
Urto con ostacolo	10279	12370	134
Urto con veicolo in fermata o sosta	19996	26770	49

I nostri totali.

Decisamente easy, non per i pedoni.

Qual e' l'incidente con la maggior frequenza ?

Risposta n. 1: Scontro frontale-laterale.

5# Rapporti

```
#Calcoliamo qualche coefficiente
data_summed['M/I (x 1000)'] =
        data_summed["Morti"] / data_summed["Incidenti"] * 1000
data_summed['F/I (x 100)'] =
        data_summed["Feriti"] / data_summed["Incidenti"] * 100
data_summed['Frequenza relativa'] =
        data_summed["Incidenti"] / data_summed["Incidenti"].sum() * 100
```

- definiamo una nuova column come un singolo elemento di un dizionario.
- Nel caso sopra indichiamo come valori delle nuove columns il risultato di alcune operazioni
- Calcoliamo la somma totale di una column semplicemente con data[“column”].sum()

	Incidenti	Feriti	Morti	M/I (x 1000)	F/I (x 100)	Frequenza relativa
NaturaIncidente						
Altre cause	4170	4622	7	1.678657	110.839329	2.293818
Fuoriuscita, sbandamento	17933	19367	53	2.955445	107.996431	9.864516
Investimento pedone	21783	25898	294	13.496764	118.890878	11.982310
Scontro frontale	3370	5142	36	10.682493	152.581602	1.853757
Scontro frontale-laterale	62842	93477	207	3.293975	148.749244	34.567888
Scontro laterale	20219	23988	52	2.571838	118.640882	11.121990
Tamponamento	21201	32929	24	1.132022	155.318145	11.662165
Urto con ostacolo	10279	12370	134	13.036288	120.342446	5.654233
Urto con veicolo in fermata o sosta	19996	26770	49	2.450490	133.876775	10.999323

Le nostre superdivisioni.

Boom baby !

Qual e’ la tipologia di incidente con la mortalita’ piu’ alta ?

Risposta n. 2: Investimento pedone.

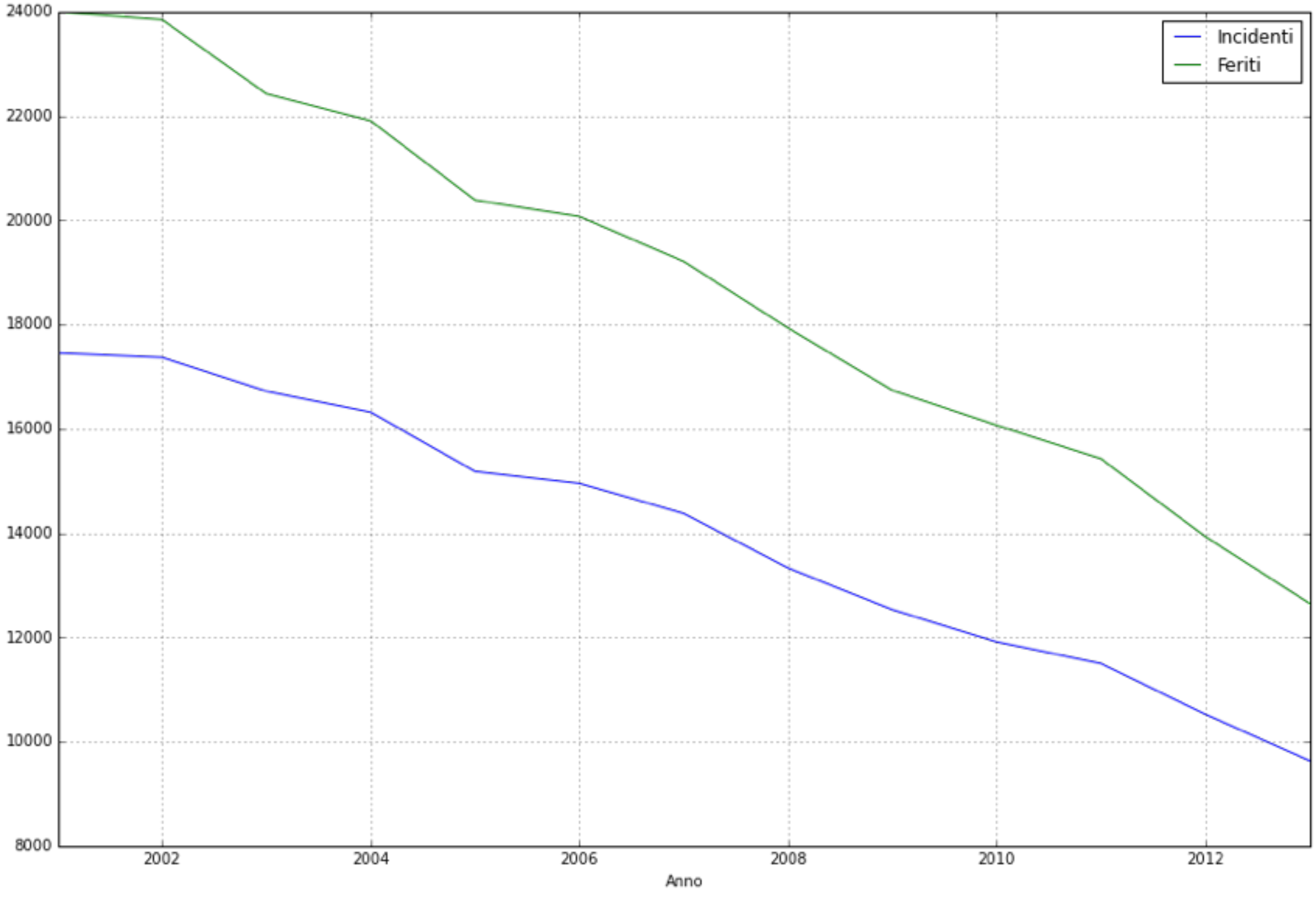
6# Andamenti storici

```
#Andamento annuale
andamento = datami.groupby("Anno").sum()
```

```
andamento = andamento.drop("Mese", 1)
```

```
andamento[["Incidenti","Feriti"]].plot(figsize=(15, 10))
```

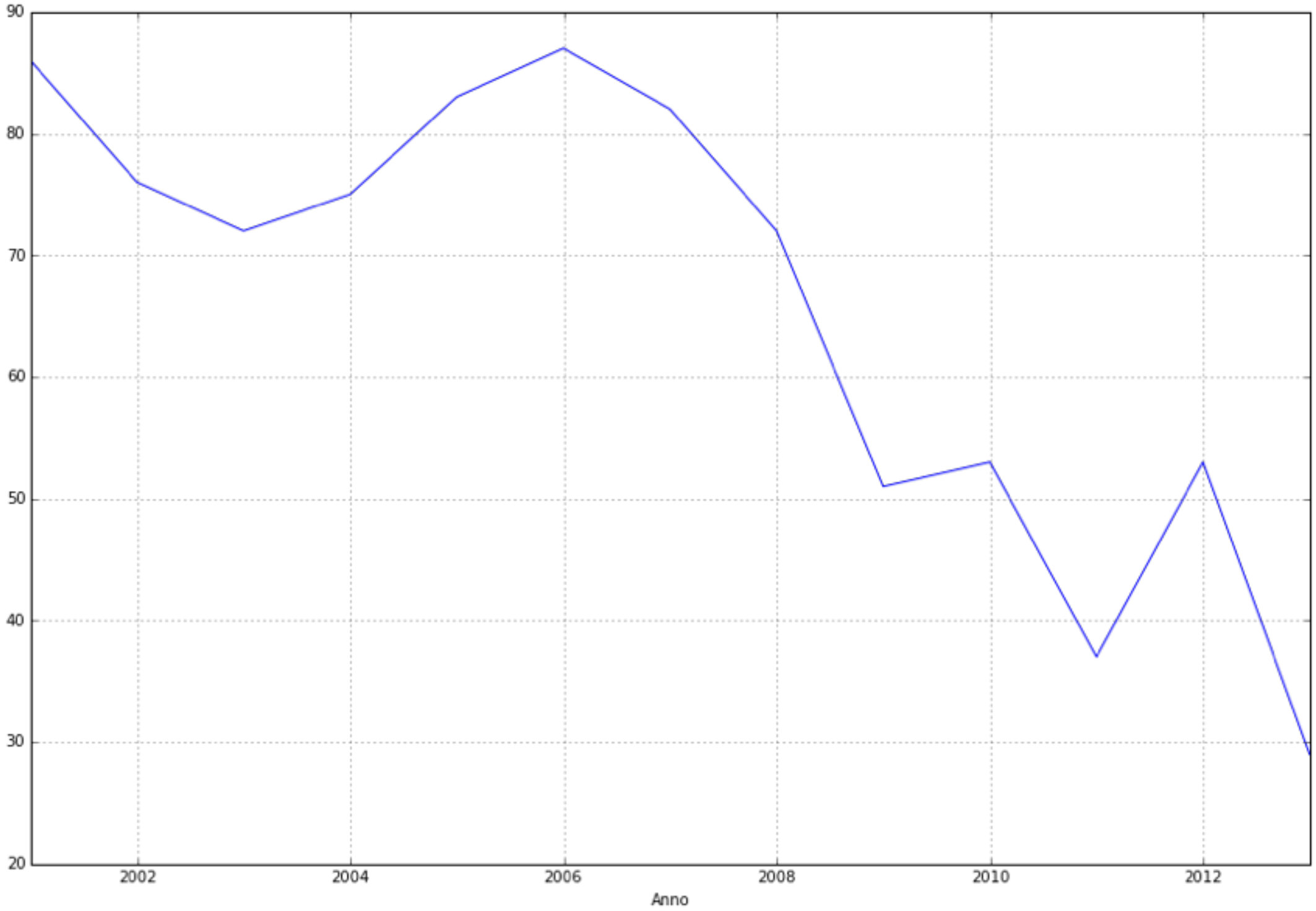
- Questa volta “grouppiamo” per “Anno”
- Eliminano la column “Mese”
- “Plottiamo” solo le columns “Incidenti” e “Feriti”


Incidenti
Feriti
24000
22000
20000
18000
16000
14000
12000
10000
8000
2002
2004
2006
2008
2010
2012
Anno

6.1# Andamenti storici

```
andamento["Morti"].plot(figsize=(15, 10))
```

- Questa volta “plottiamo” solo la column “Morti”


90
80
70
60
50
40
30
20
2002
2004
2006
2008
2010
2012
Anno

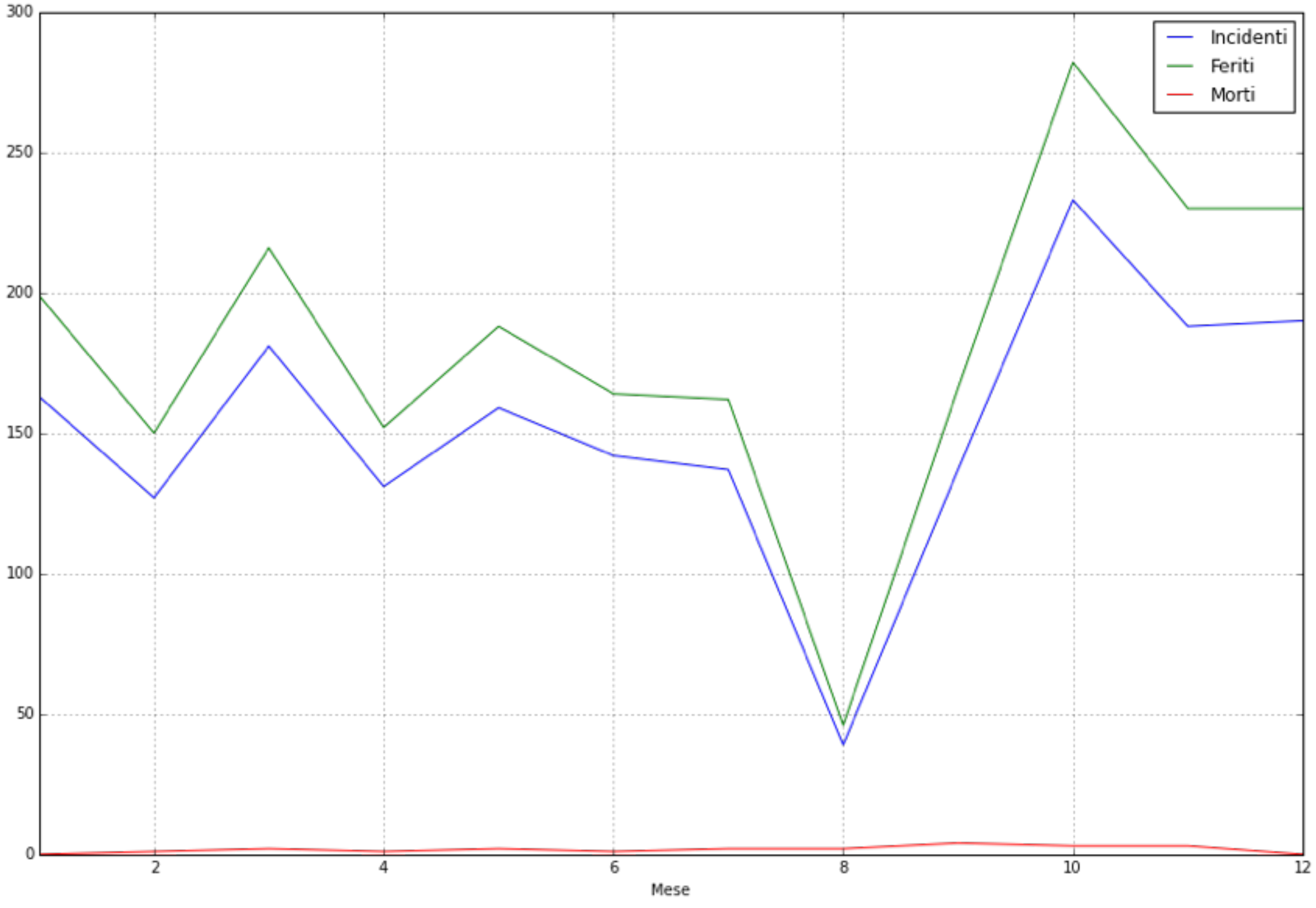
Il numero di incidenti, feriti e morti e’ diminuito durante gli anni ?

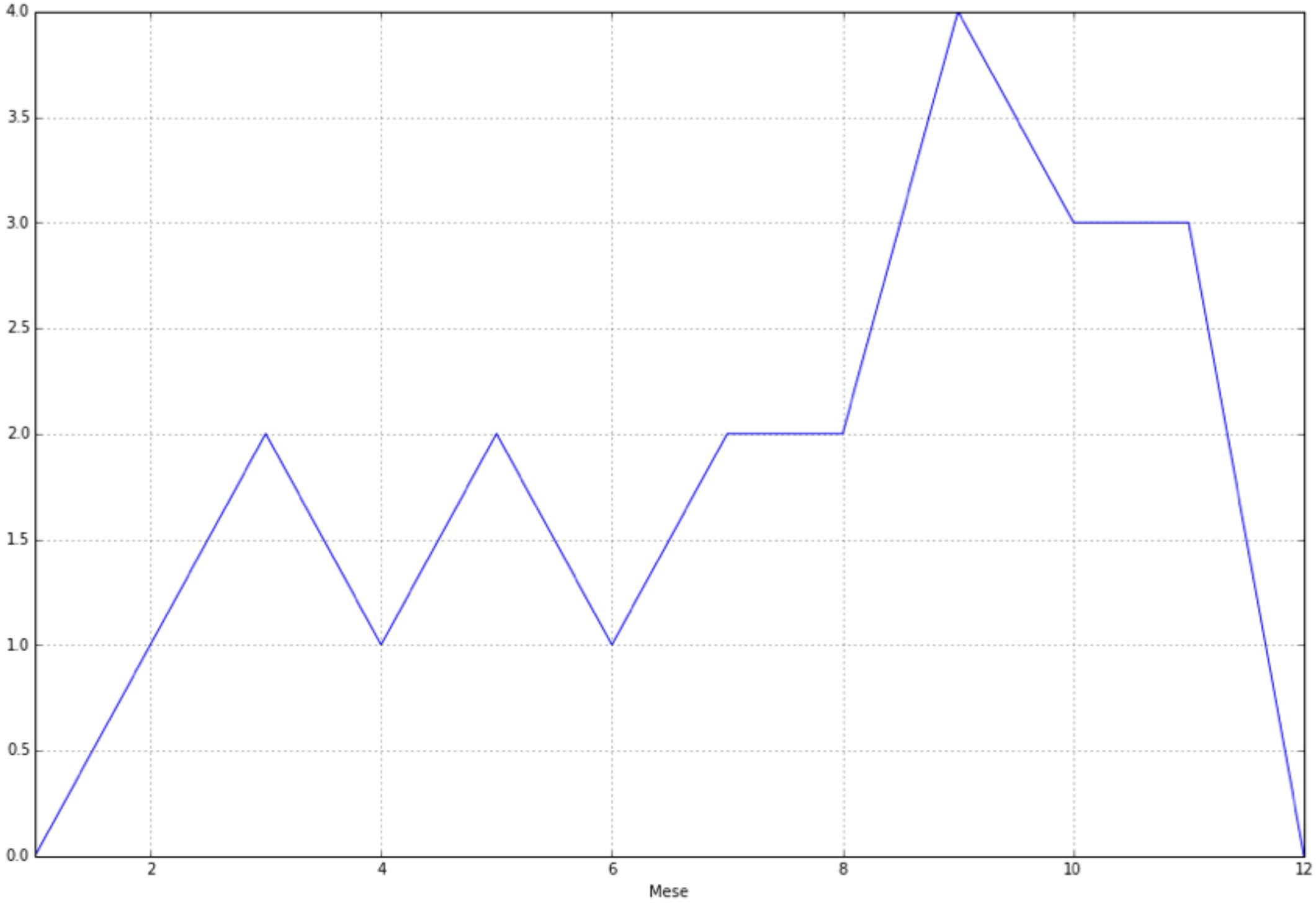
Risposta n. 3: Si, sono diminuiti.

7# Andamento mensile

```
morti_inv_2003 =
    datami[(datami['Anno'] == 2003)
           & (datami['NaturaIncidente'] == "Investimento pedone")]
morti_inv_2003 = morti_inv_2003.set_index("Mese")
morti_inv_2003 = morti_inv_2003.drop("Anno", 1)
morti_inv_2003.plot(figsize=(15, 10))
morti_inv_2003["Morti"].plot(figsize=(15, 10))
```

- Creiamo un nuovo dataframe partendo da datami
- Condizioni “Anno” = 2003 e “NaturaIncidente” = “Investimento pedone”
- Settiamo la column “Mese” come index
- Eliminiamo la column “Anno”

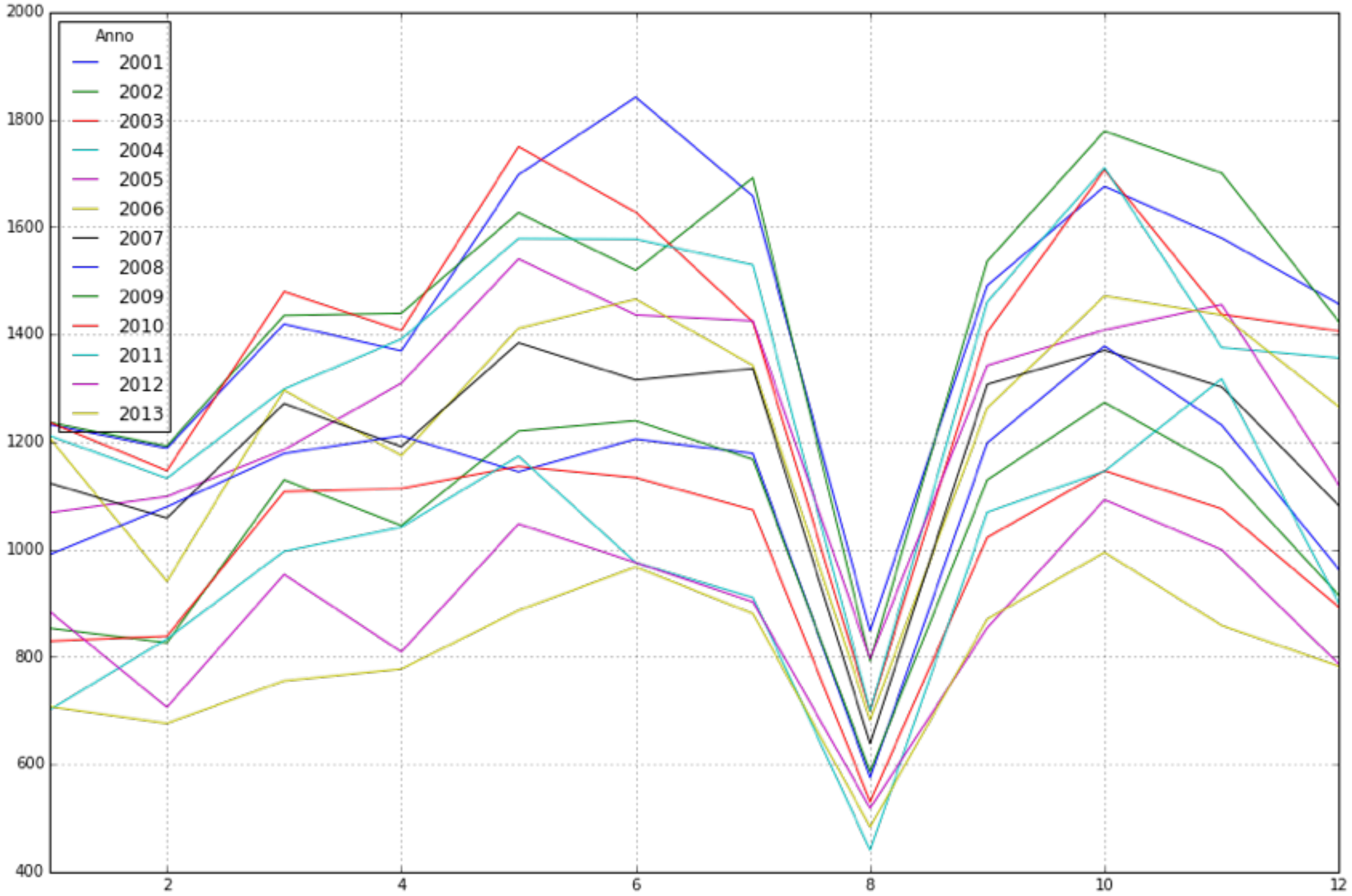

Incidenti
Feriti
Morti
300
250
200
150
100
50
0
2
4
6
8
10
12
Mese


4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
2
4
6
8
10
12
Mese

7.1# Andamento mensile storico

```
#Andamento mensile negli anni
columns = datami.groupby(["Anno","Mese"]).sum()
c_mensile = pd.DataFrame(index=[1,2,3,4,5,6,7,8,9,10,11,12],
                         columns=columns.index.levels[0])
for i in columns.index.levels[0]:
 c_mensile[i] = columns.loc[i]["Incidenti"]
c_mensile.plot(figsize=(15, 10))
```

- Creiamo un dataframe “grouppando” anno e mese sommando i dati
- Creiamo un nuovo dataframe con index uguale ai 12 mesi e come columns i singoli anni presi direttamente dal dataframe precedente
- Il for non ci serve ad altro che per copiare i dati dal primo dataframe al secondo
- loc ti serve quando lavori sugli index
- Probabilmente esiste un metodo piu’ elegante per farlo.


Anno
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2000
1800
1600
1400
1200
1000
800
600
400
2
4
6
8
10
12

Esiste una qualche stagionalita’ nel numero di incidenti ?

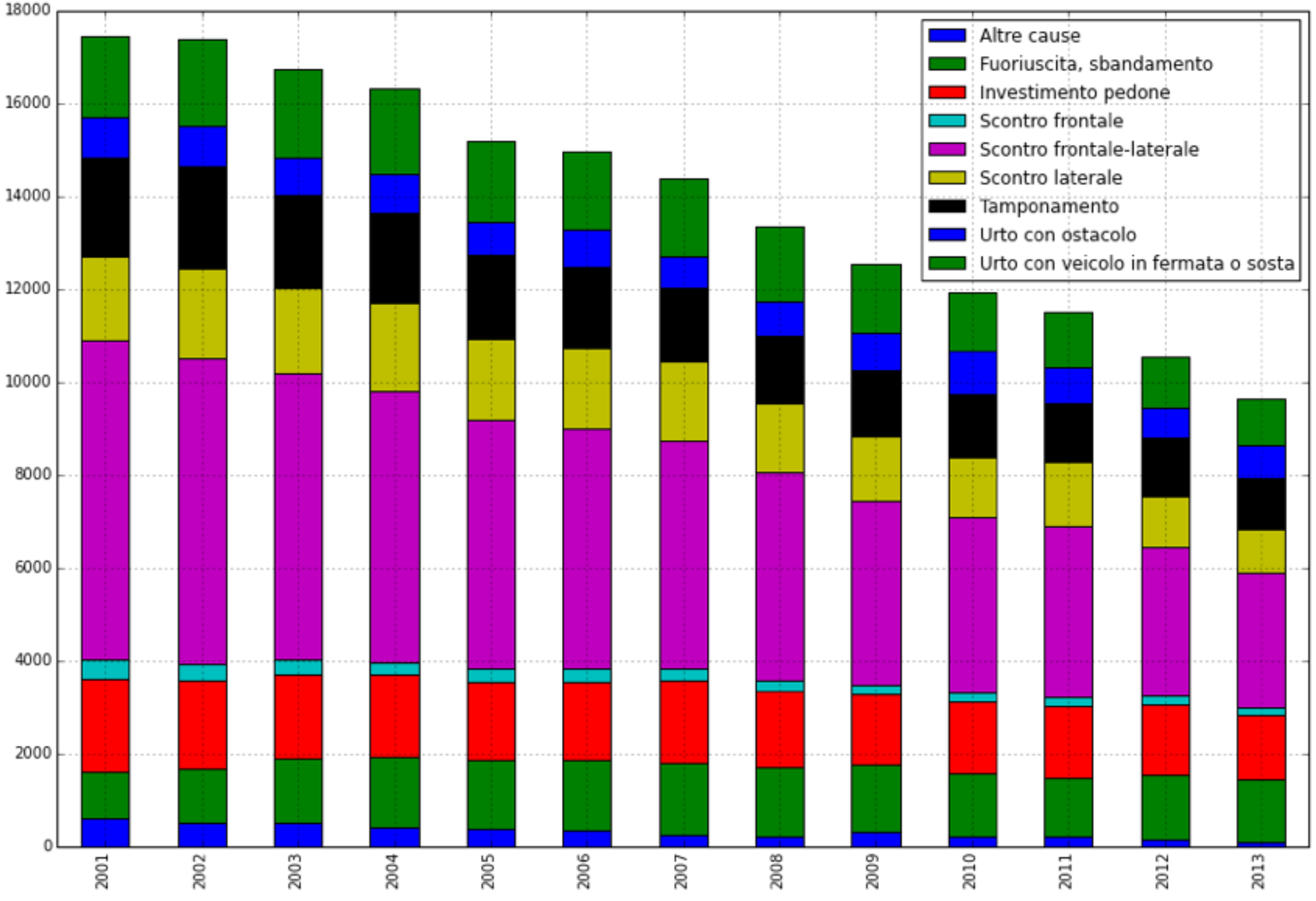
Risposta n. 4: Direi di si.

8# Composizione del numero di incidenti

```
df = datami.groupby(["Anno","NaturaIncidente"]).sum()
df = df.drop(["Mese"],1)
df = df.reset_index()
andamento = pd.DataFrame(
    {anno: df[df["Anno"] == anno]["Incidenti"].values
     for anno in df["Anno"].unique()},
    index=df["NaturaIncidente"].unique())
andamento.T.plot(kind="bar",stacked=True,figsize=(15, 10))
andamento
```

- Partiamo sempre dal dataframe iniziale “grouppando” anno e NaturaIncidente
- Droppiamo la column “Mese”, resettiamo l’index, in pratica ricreiamo un nuove indice numerico
- Creiamo un nuovo dataframe, utilizzando la dict comprehension e come index i valori “unique” della column df[“NaturaIncidente”]
- “T” fa diventare gli indexes columns e le columns indexes, in questo modo posso avere gli anni nel grafico sull’asse delle x
- Probabilmente esiste un metodo piu’ elegante per farlo.

	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
Altre cause	596	494	512	397	389	348	259	204	296	213	216	159	87
Fuoriuscita, sbandamento	1022	1159	1371	1541	1456	1515	1527	1509	1466	1367	1267	1388	1345
Investimento pedone	1989	1906	1827	1747	1704	1684	1770	1623	1521	1547	1533	1522	1410
Scontro frontale	424	359	312	286	293	270	279	230	205	187	192	170	163
Scontro frontale-laterale	6855	6577	6165	5840	5352	5169	4898	4473	3963	3764	3692	3202	2892
Scontro laterale	1804	1933	1838	1893	1724	1755	1702	1496	1383	1289	1365	1112	925
Tamponamento	2137	2194	1983	1938	1801	1731	1584	1456	1410	1363	1262	1232	1110
Urto con ostacolo	860	886	818	845	726	797	691	750	826	942	779	657	702
Urto con veicolo in fermata o sosta	1765	1859	1893	1829	1736	1683	1666	1588	1460	1240	1193	1083	1001


18000
16000
14000
12000
10000
8000
6000
4000
2000
0
Altre cause
Fuoriuscita, sbandamento
Investimento pedone
Scontro frontale
Scontro frontale-laterale
Scontro laterale
Tamponamento
Urto con ostacolo
Urto con veicolo in fermata o sosta
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013

Esiste una qualche stagionalita’ nel numero di incidenti ?

Risposta n. 5: Il numero di incidenti e’ generalmente diminuito, tranne per “Fuoriuscita, Sbandamento”

```
Periodo,Pioggia(mm)
01/01/2001,0.1
01/02/2001,3.8
01/03/2001,11.4
01/04/2001,15.6
01/05/2001,5.2
01/06/2001,15.8
```

Che la pioggia sia con noi.

Si ringrazia l’ARPA lombarda.

```
#Dati sul clima
clima = pd.read_csv("data/meteo_milano.csv", sep=",",
                    index_col="Periodo", parse_dates=True)
clima_data = clima.resample("M", how="mean")

#Correlation POWAA !
data_2013 = datami[datami["Anno"] == 2013]
data_2013 = data_2013.drop("Anno",1)
clima_2013 = clima_data.loc["2013"]

for i in data_2013["NaturaIncidente"].unique():
    temp = data_2013[data_2013["NaturaIncidente"] == i]
    temp = temp.drop("NaturaIncidente",1)
    temp = temp.set_index(clima_2013.index) ###
    clima_2013["Incidenti " + i] = temp['Incidenti']
    clima_2013["Feriti " + i] = temp['Feriti']
    clima_2013["Morti " + i] = temp['Morti']
```

```
corr_2013 = clima_2013.corr()
corr_2013["Pioggia(mm)"]
    [(corr_2013["Pioggia(mm)"] >= 0.25)
     | (corr_2013["Pioggia(mm)"] <= -0.25)]
```

9# Sembra complicato

Ma non lo e’.

- Partiamo sempre dal dataframe iniziale “grouppando” anno e NaturaIncidente

- Droppiamo la column “Mese”, resettiamo l’index, in pratica ricreiamo un nuove indice numerico

- Creiamo un nuovo dataframe, utilizzando la dict comprehension e come index i valori “unique” della column df[“NaturaIncidente”]

- “T” fa diventare gli index -> columns e le columns -> index, in questo modo posso avere gli anni nel grafico sull’asse delle x

- Probabilmente esiste un metodo piu’ elegante per farlo.

	Pioggia(mm)	Incidenti Scontro frontale	Feriti Scontro frontale	Morti Scontro frontale	Incidenti Scontro frontale-laterale	Feriti Scontro frontale-laterale	Morti Scontro frontale-laterale	Incidenti Scontro laterale
Periodo								
2013-01-31	2.419355	13	20	0	210	331	1	62
2013-02-28	1.821429	12	19	0	193	266	1	54
2013-03-31	6.264516	14	21	0	247	376	0	80
2013-04-30	5.686667	15	26	0	243	360	2	72
2013-05-31	6.000000	16	23	0	272	401	1	82
2013-06-30	1.986667	15	22	0	297	423	0	96
2013-07-31	0.503226	11	14	0	278	388	1	100

clima_2013 dataframe

Per chi se lo chiede questa non e’ tutta la tabella.

```
Pioggia(mm)                                1.000000
Incidenti Scontro frontale                 0.388274
Feriti Scontro frontale                    0.451986
Feriti Scontro frontale-laterale           0.312275
Incidenti Tamponamento                     0.361103
Feriti Tamponamento                        0.374726
Incidenti Urto con ostacolo                0.613364
Feriti Urto con ostacolo                   0.566065
Morti Urto con ostacolo                    0.259459
Incidenti Fuoriuscita, sbandamento        -0.258573
Feriti Fuoriuscita, sbandamento           -0.261792
Morti Fuoriuscita, sbandamento            -0.401988
Incidenti Altre cause                     -0.322604
Feriti Altre cause                        -0.425022
Name: Pioggia(mm), dtype: float64
```

I nostri coeff. di correlazione.

Quando piove la gente si diverte ad andare contro le cose.

Esiste qualche correlazione tra la piovosita’ e il numero di incidenti, feriti, morti nel 2013 ?

Risposta n. 6: Si, positiva e negativa, per alcune tipologie di incidenti.

Domande ?

Guidate con prudenza e occhio ai panda !