UNIVERSITÀ
degli STUDI
di CATANIA

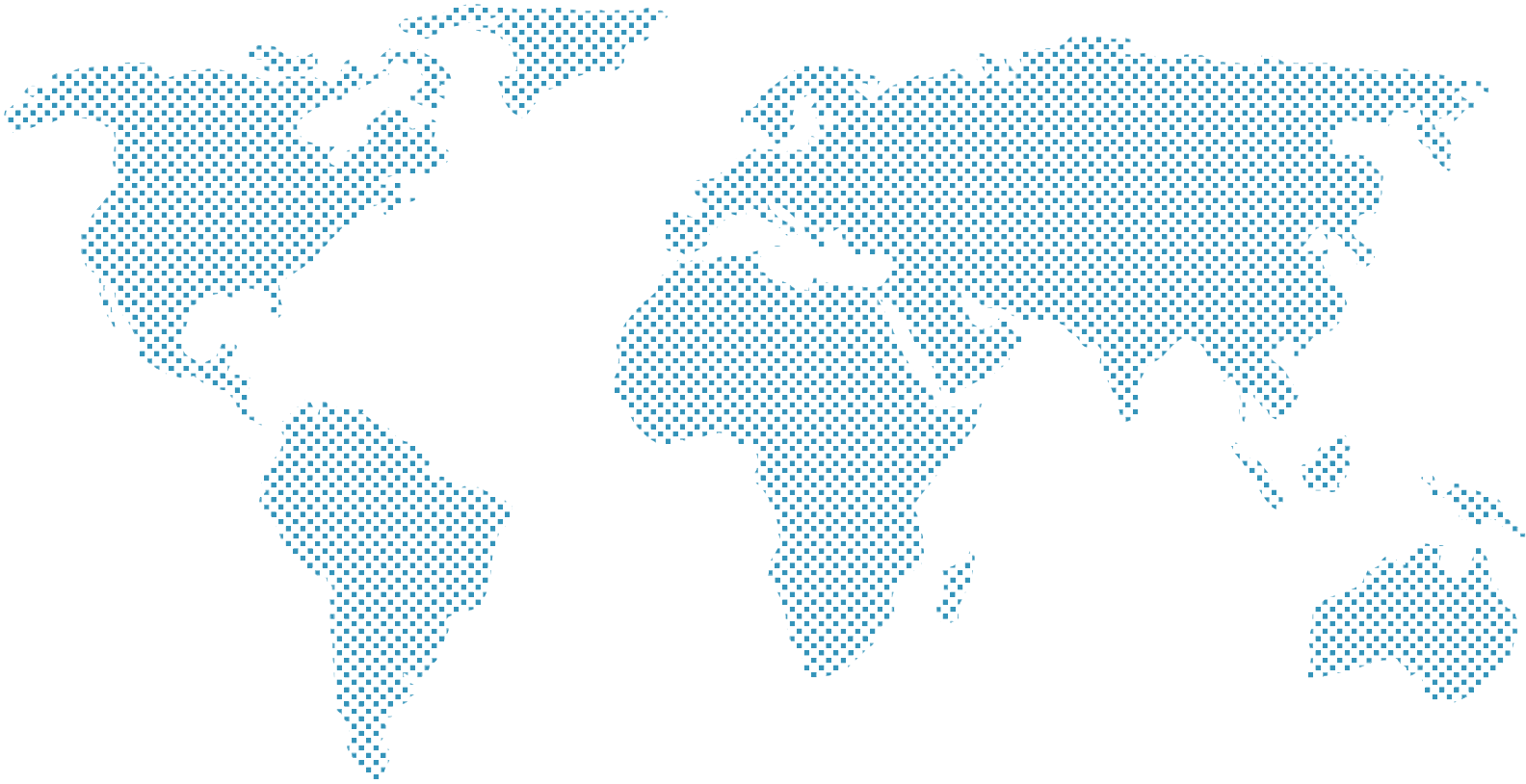
Dipartimento di Matematica e Informatica
Corso di Laurea Triennale in Informatica

Relazione

Metodi Matematici e Statistici
Prof.re Orazio Muscato
A.A. 2019/2020

COVID-19

REGRESSION ANALYSIS AND DATA VISUALIZATION CORONAVIRUS DISEASE IN ITALY



28/03/2020

Studentesse:

Francesca Ragazzi

Matricola: X81000697

Clizia Giorgia Manganaro

Matricola: X81000716

INDICE

PREMESSA

A partire dal mese di novembre 2019 si è sviluppato in Cina un nuovo virus respiratorio, appartenente alla famiglia dei Coronavirus (CoV), denominato successivamente dagli studiosi COVID-19, (COronaVIrus Disease 2019). Tali virus possono causare malattie sia lievi che moderate, dal comune raffreddore a sindromi respiratorie acute-gravi.

La diffusione del virus in Italia ha avuto le sue prime manifestazioni epidemiche il 30 gennaio 2020 quando due turisti provenienti dalla Cina risultarono positivi al Covid-19.

Il 21 febbraio 2020 l'Istituto Superiore di Sanità ha confermato il primo caso autoctono in Italia verificatosi a Codogno, Comune della Lombardia in provincia di Lodi, causando successivamente una rapida diffusione in tutta il territorio nazionale che al momento risulta essere uno dei paesi più colpiti causando danni non solo sociali ma anche economici.

Il 9 marzo 2020 viene emanato il DPCM che decreta misure urgenti di contenimento del contagio sull'intero territorio nazionale allo scopo di contrastare il diffondersi del virus.

L'11 marzo del 2020 l'Organizzazione Mondiale Della Sanità dichiara lo stato di pandemia con più di 118.000 casi in 114 paesi e 4.291 deceduti.

INTRODUZIONE

Il fulcro dell'indagine proposta in questo progetto è l'analisi dei *legami funzionali* tra i dati per una visione completa dei contagi di COVID-19 in Italia; utilizzeremo nel nostro studio, i dati pubblicati e quotidianamente aggiornati dal Dipartimento della Protezione Civile in collaborazione con la Presidenza del Consiglio dei Ministri, sfruttando la tecnica della *regressione lineare.*

Abbiamo scelto di utilizzare come linguaggio di programmazione Python 3.8[1], in quanto permette l'integrazione di numerose librerie, con cui è possibile manipolare dati e ricavare tutte le specifiche che si desiderino.
Nel dettaglio le varie operazioni effettuate sono state svolte con l'ausilio della libreria matplotlib[2]., che consente di incorporare grafici ad applicazioni, i quali hanno lo scopo di fornire immediatamente le caratteristiche essenziali del fenomeno *oggetto* dell'indagine svolta. Python inoltre, è un linguaggio di programmazione ad alto livello, derivato da C, interpretato e multipiattaforma che lo rende versatile nel suo utilizzo.

In un primo caso di studio analizzeremo l'andamento nazionale dei casi a partire dal 25 febbraio 2020, nel quale abbiamo effettuato una visualizzazione complessiva, contenente l'andamento dei contagi in relazione alla data tramite un *line plot* (grafico a linee). Successivamente, grazie ad una matrice mostreremo la correlazione tra tutti i campi del dataset ed infine proseguiremo la nostra indagine analizzando a coppie i campi da noi ritenuti più significativi.

In un secondo caso di studio verrà prodotto un diagramma a barre, laddove verrà mostrato il numero totale di casi positivi per regione, riportati in ordine decrescente, utilizzando ulteriori dati del dataset sopracitato.

In conclusione, alla luce dei risultati ottenuti, tenteremo di dimostrare come i dati siano influenzabili tra loro.

1. DESCRIZIONE DATASET

L'analisi è stata condotta utilizzando i dati COVID-19 Italia reperibili sul repository GitHub del profilo della Presidenza del Consiglio dei Ministri - Dipartimento della Protezione Civile[3].
Per informare i cittadini e mettere a disposizione i dati raccolti, il Dipartimento della Protezione Civile mette a disposizione le informazioni aggiornate quotidianamente alle 18:30, in particolare per la nostra analisi di indagine andremo ad utilizzare le directory dati-andamento-nazionale e dati-regioni presenti in repository.

Le directory dati-andamento-nazionale sono formate dai seguenti campi:

- Data;
- Stato;
- ricoverati_con_sintomi;
- terapia_intensiva;
- totale_ospedalizzati;
- isolamento_domiciliare;
- totale_attualmente_positivi;
- nuovi_attualmente_positivi;
- dimessi_guariti;
- deceduti;
- totale_casi;
- tamponi.

Alla directory dati-regioni oltre i campi sopra riportati vengono aggiunti:

- denominazione_regione;
- lat;
- long.

Andremo, in un primo momento, ad approfondire la correlazione tra le serie dei campi tramite il coefficiente di correlazione lineare (o di Pearson) e andremo a calcolare la covarianza tra:

- tamponi e totale_casi;
- totale_attualmente_positivi e totale_ospedalizzati;
- totale_ospedalizzati e terapia_intensiva;
- totale_casi e dimessi_guariti;
- totale_casi e deceduti.

Verrà visualizzata la retta di regressione lineare per i campi sopracitati.

In un secondo momento, tramite un grafico a barre *(bar plot)* e la directory dati-regioni, verrà visualizzato il numero di casi per regione, i quali sono stati precedentemente ordinati in modo decrescente per numeri di casi, tramite i campi del dataset denominazione_regione

In questo tipo di grafico, ogni modalità della variabile x_i viene rappresentata da un rettangolo avente base costante, la cui altezza (o lunghezza) è proporzionale alla frequenza.

2. BASI TEORICHE[4]

2.1. CORRELAZIONE TRA DUE SERIE DI DATI

Assumiamo che {xi} e {yi} siano due serie di set di dati aventi caratteri quantitativi di numerosità n. Vogliamo verificare se esiste una relazione tra questi dati, per fare ciò confrontiamo le variazioni delle coppie di set di dati rispetto ai valori medi

$$xi - \bar{x} \qquad yi - \bar{y}.$$

Per esserci una dipendenza tra $\{x_i\}$ $\{y_i\}$ dobbiamo supporre che $xi - \bar{x}$ e $yi - \bar{y}$ devono essere dello stesso segno ed inoltre, se i loro prodotti $(xi - \bar{x})(yi - \bar{y})$ sono concordi tanto più i dati hanno una forte dipendenza.

A questo punto possiamo definire la covarianza come l'indice per misurare la dipendenza fra due set di dati

$$c_{xy} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

La covarianza possiede le seguenti proprietà:

- $c_{xy} \epsilon \mathbb{R}$;
- $c_{xy} > 0$ i due set di dati $\{x_i\}\{y_i\}$ sono correlati positivamente;
- $c_{xy} < 0$ i due set di dati $\{x_i\}\{y_i\}$ sono correlati negativamente;
- $c_{xy} = 0$ i due set di dati $\{x_i\}\{y_i\}$ sono statisticamente incorrelati;
- se $\{x_i\}\{y_i\}$ sono fortemente correlati c_{xy} è grande in valore assoluto.

2.2. ANALISI DI REGRESSIONE

L' analisi di regressione è una tecnica della statistica descrittiva che permette di verificare l'esistenza di una relazione tra due variabili, che possa essere descritta da una funzione del tipo:

$$y = f(x)$$

Lo studio della regressione consiste, dunque, nella determinazione di una funzione matematica che esprima la relazione tra due dati.

2.3. METODO DEI MINIMI QUADRATI

Per determinare la funzione che leghi due variabili x_i y_i, utilizziamo il metodo dei minimi quadrati:

$$g(f) = \sum_{i=1}^{n}[f(x_i) - y_i]^2$$

Tale funzione deve minimizzare la somma dei quadrati delle distanze tra i dati osservati nel grafico di dispersione e i valori teorici sulla retta di regressione.

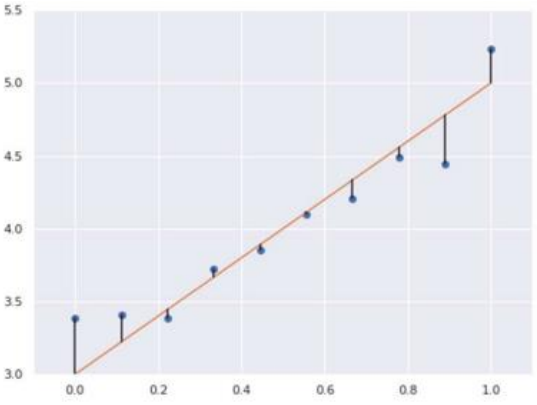


Figura- 1 Grafico di dispersione

2.4. REGRESSIONE LINEARE

Nel dettaglio il metodo utilizzato per questo tipo di indagine è quello della regressione lineare dove la funzione f è una retta data da $f(x) = mx + q$.

Per calcolare m e q nella retta di regressione utilizzo il metodo dei minimi quadrati:

$$g(m,q) = \sum_{i=1}^{n} [mx_i + q - y_i]^2$$

poiché è una funzione di due variabili, affinché la distanza sia un minimo relativo dobbiamo calcolarci le derivate parziali rispetto a m e q ed uguagliarle a 0 ottenendo infine

$$m = \frac{\frac{1}{n}\sum_i x_i y_i - \overline{xy}}{\frac{\sum_i x_i^2}{n} - (\bar{x})^2}$$

e sostituendo con la formula della varianza per la variabile x e della covarianza

$$s_x^2 = \frac{\sum_i x_i^2}{n} - (\bar{x})^2 \qquad c_{xy} = \frac{1}{n}\sum_{i=1}^{n}(x_i y_i - \bar{x}\bar{y})$$

otterremo $m = \frac{c_{xy}}{s_x^2}$ da cui ricavo $q = \bar{y} - \frac{c_{xy}}{s_x^2}\bar{x}$

2.5. COEFFICIENTE DI CORRELAZIONE LINEARE

Il coefficiente di correlazione lineare (o di Pearson) permette di valutare il grado di concordanza tra due caratteri quantitativi. Viene definito come il rapporto tra la covarianza e la varianza per la variabile x e y.

$$r_{xy} = \frac{c_{xy}}{s_x s_y}$$

Il coefficiente di correlazione possiede le seguenti proprietà:

1. $r_{xy} \in [-1,1]$
2. $se\ r_{xy} = \pm 1$ i dati sono perfettamente allineati con la retta di regressione
3. $se\ r_{xy} > 0$ la retta è ascendente
4. $se\ r_{xy} < 0$ la retta è discendente
5. $se\ r_{xy} = 0$ non vi è nessuna relazione lineare tra x e y

Inoltre, nella pratica, se $|r_{xy}| < 0.9$, i dati si allontanano dall'andamento rettilineo

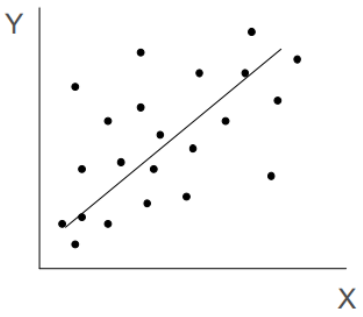


Figura -2
RETTA ASCENDENTE
Se r_{xy}>0. I valori sono correlati positivamente.

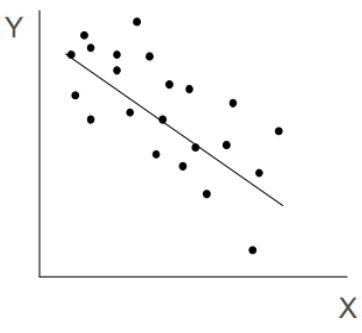


Figura-3
RETTA DISCENDENTE
Se r_{xy}<0. I valori sono correlati negativamente.

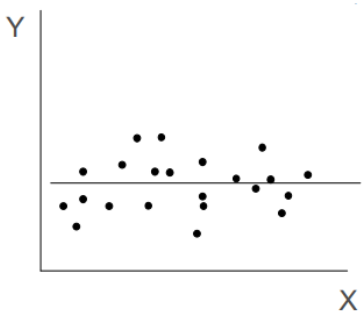


Figura-4
RETTA PARALLELA ALL'ASSE X
Se r_{xy}=0 i valori non sono correlati.

3. FUNZIONAMENTO DEL SOFTWARE

Come già accennato, il software è stato interamente sviluppato in Python 3.8[1]., sarà quindi necessario aver installato la versione 3.8 nel proprio dispositivo.

Inoltre, sarà necessario installare le seguenti librerie per il corretto funzionamento del software.

```
In [5]: #importiamo le librerie che andremo ad utilizzare
        from sklearn.linear_model import LinearRegression
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        %matplotlib inline
        import seaborn as sns
```

Vengono prelevati i dati dal repository.

```
In [6]: #fetch dataset.csv
        url = "https://raw.githubusercontent.com/pcm-dpc/COVID-19/master/dati-andamento-nazionale/dpc-covid19-ita-andamento-nazionale.cs
        df = pd.read_csv(url)
        df
```

3.1. LINE PLOT

Al fine di analizzare la relazione 'totale_casi' e 'data' viene rappresentato in un grafico l'andamento a partire dal 24 febbraio 2020 sino a data odierna. La data verrà aggiornata automaticamente con i casi giornalieri alle ore 18:30. Viene rappresentata nella linea rossa i dati legati a numero casi e data.

```
In [24]: list_data = [str(s)[:10] for s in df.data]
         df.data = list_data
         df_casi = df.loc[:,['data','totale_casi']]
         plt.figure(figsize=(20, 4), dpi=100)
         plt.title("totale casi e giorni")
         plt.xticks(rotation=45)
         fmri = sns.load_dataset("fmri")
         ax = sns.lineplot(x="data", y="totale_casi", data=df_casi,color="red" )
         ax.grid(b=True, which='major', color='#666666', linestyle='-')
```

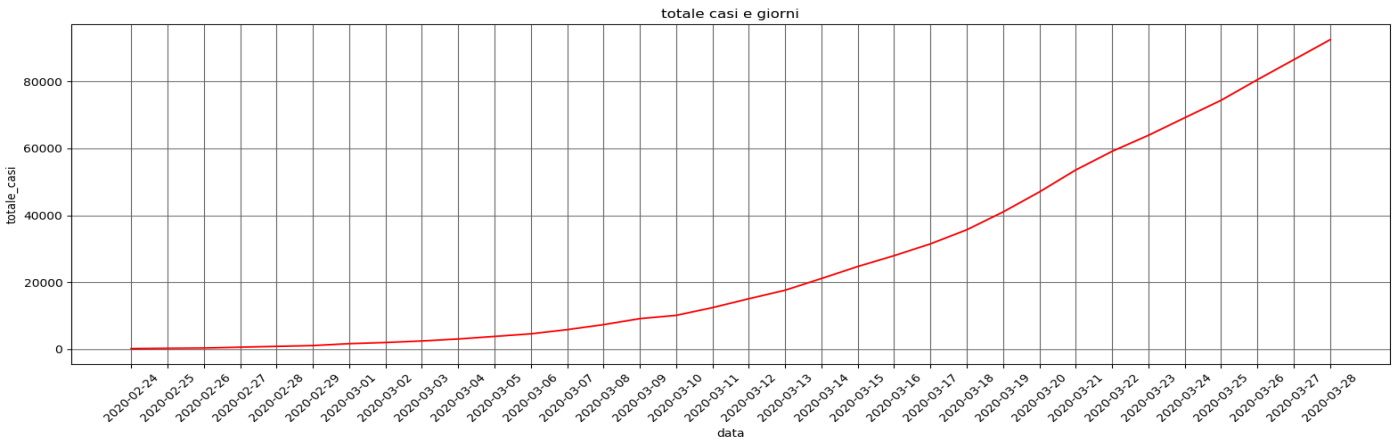


Figura- 5 *Lineplot*

3.1.1. CURVE FITTING

Verifichiamo se il numero di contagi, ad oggi si possa approssimare ad un andamento esponenziale utilizzando la funzione *.curve_fit()*[12] che utilizza il metodo dei minimi quadrati per adattare una funzione (che nel nostro caso sarà $y = ab^x$) ai dati. Visualizziamo nella linea rossa il numero di totale_casi, nella linea blu la retta di regressione ed infine, nella linea verde la curva esponenziale fittata (Figura 5.1).

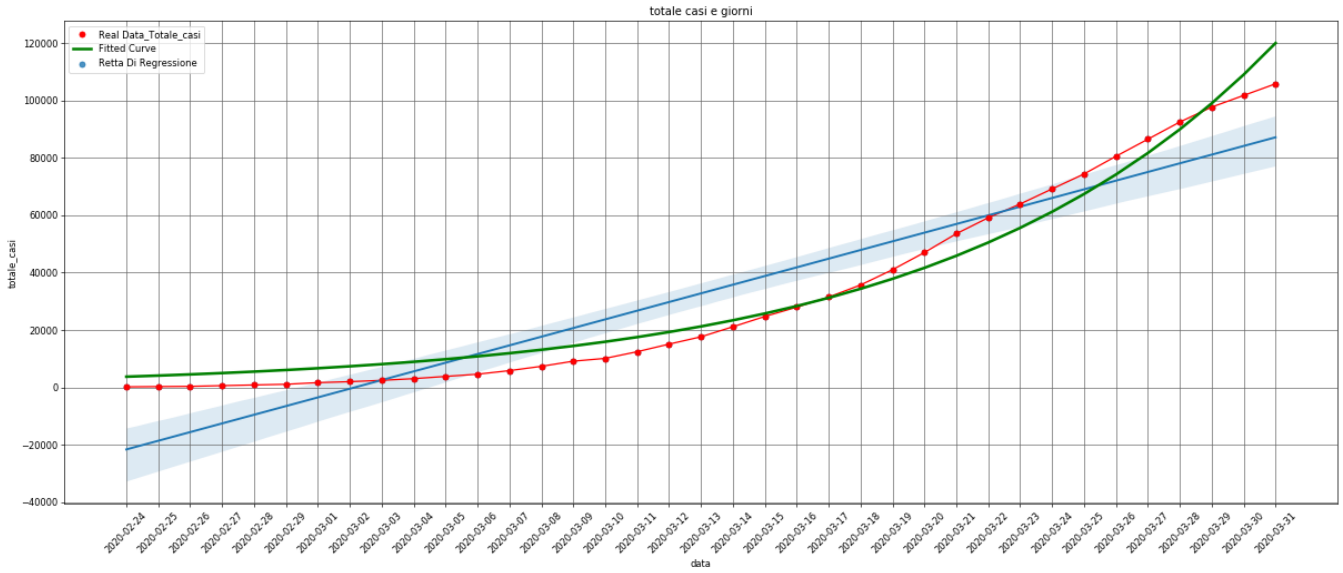


Figura- 5.1 *Andamento esponenziale con Retta di regressione*

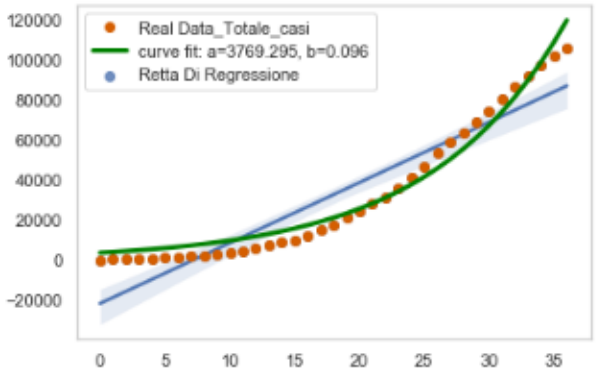
```
In [19]: from scipy.optimize import curve_fit
a= np.arange(df_casi.shape[0]).astype(dtype=float).reshape(1,-1)
b= df_casi.totale_casi.to_numpy(dtype=float).reshape(1, -1)
reg = LinearRegression()
reg.fit(a, b)
sns.regplot(a, b, label="Retta Di Regressione")
x = np.arange(df_casi.shape[0]).astype(dtype=float)
y = df_casi.totale_casi.to_numpy(dtype=float)
plt.plot(x, y, 'ro', label="Real Data_Totale_casi")

def func(x, a, b):
    return a*np.exp(b*x)

l=(tuple(popt))
my_list=list(l)
print(' valori a e b: ',my_list)
popt, pcov = curve_fit(func, x, y)

plt.plot(x, func(x, *popt), label='curve fit: a=%5.3f, b=%5.3f'% tuple(popt),color='green',lw=3.0)
plt.legend(loc='upper left')
plt.grid()
plt.show()
```

```
valori a e b:  [3769.2951318971986, 0.0961228963632327]
```



3.1.2. DIAGRAMMA LOG-LOG E SEMI-LOG

Ponendo nell''asse x i giorni riportati numericamente, per verificare se l'andamento è esattamente esponenziale eseguiamo un plot con X=log(x) e Y=log(y) (Figura 5.2) tramite la funzione *.loglog()*[13] appartenente alla libreria matplotlib.

Successivamente andiamo ad mostrare in un altro plot la funzione *.semilogy()*[14] ponendo X=x e Y=log(y) (Figura 5.3).

```
In [42]: x= np.array(range(1,len(df_casi)+1), dtype='float')
         y = df_casi.totale_casi.to_numpy(dtype=int)
         plt.loglog(x,y, label='diagramma log-log')
         plt.legend()
         plt.show()
```

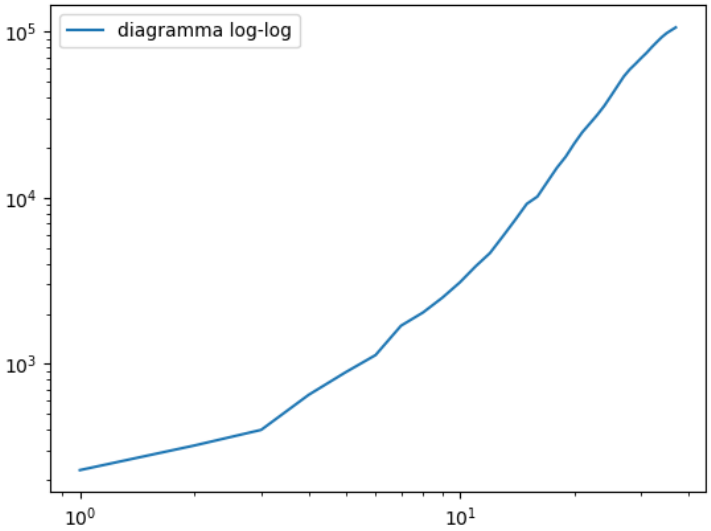
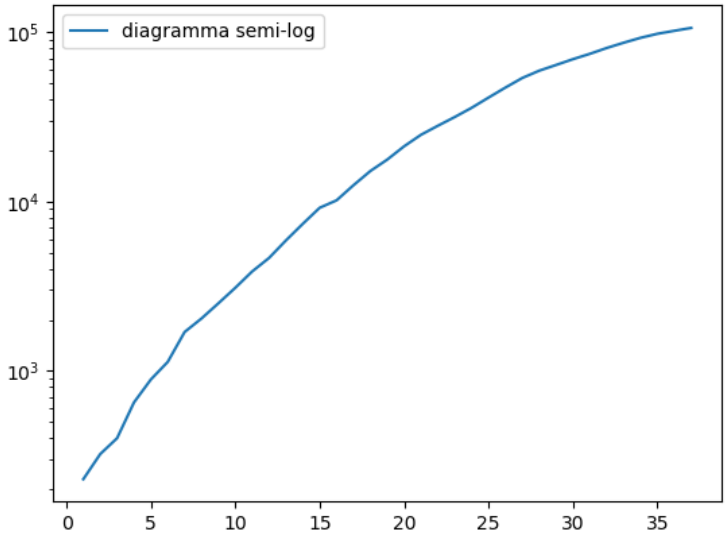


Figura- 5.2 *Diagramma log-log*

```
In [44]: x= np.array(range(1,len(df_casi)+1), dtype='float')
         y = df_casi.totale_casi.to_numpy(dtype=int)
         plt.semilogy(x,y, label='diagramma semi-log')
         plt.legend()
         plt.show()
```



3.2.MATRICE DI CORREL*Figura- 5.3 Diagramma semi-log*

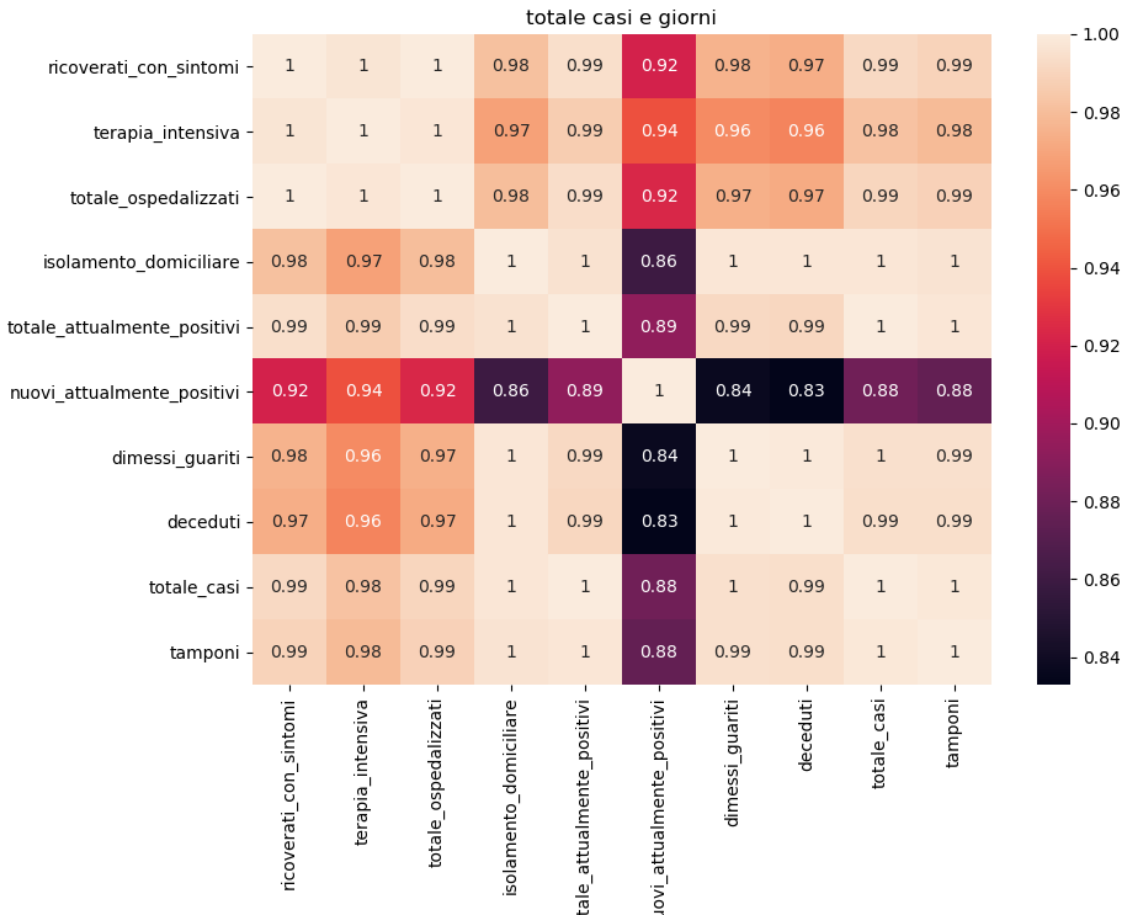
Tramite la funzione *.heatmap()*[5]. è possibile rappresentare i dati in una forma bidimensionale tramite una matrice. Alla funzione *.heatmap()* viene passato *df.corr()*[6]., in quanto abbiamo scelto di riportare i coefficienti di correlazione (o Pearson) tra due campi del dataset analizzato, all'interno di ogni cella.

L'obiettivo della funzione *.heatmap()* è fornire un riepilogo visivo delle informazioni, in quanto i valori dei dati sono rappresentati come colori nel grafico.

Sappiamo che il coefficiente di Pearson è compreso nell'intervallo [-1,1].

Noteremo che il valore minimo del coefficiente di correlazione tra due campi sarà 0.86, come visualizzato in figura nella leggenda sulla destra.

```
In [15]: sns.heatmap( df.corr(), annot=True)
         plt.show()
```



3.3. REGRESSIONI LINEARI

Analizzeremo qui di seguito le regressioni lineari tra alcuni campi del dataset.

3.3.1. Tamponi e totale casi

Per calcolare la regressione lineare come prima cosa vengono calcolati il coefficiente angolare m (tramite la funzione *.coef()*[7].) e l'intercetta q (tramite la funzione *.intercept()*[8]).

Viene, inoltre, calcolato il coefficiente di Pearson tramite la funzione *.corr()*[9].

```
In [5]: df_1=df.loc[:,['tamponi', 'totale_casi']]
        cf1 = df_1.corr()
        X=df.loc[:,['tamponi']]
        Y=df.loc[:,['totale_casi']]
        reg = LinearRegression()
        reg.fit(X, Y)
        print()
        print("intercetta q:       ",reg.intercept_)
        print("coefficiente angolare m: ", reg.coef_, "\n")
        print("coefficiente di correlazione: ",  cf1)
```

```
intercetta q:        [-3107.50958716]
coefficiente angolare m:  [[0.23304691]]

coefficiente di correlazione:
                tamponi  totale_casi
tamponi        1.000000     0.998387
totale_casi    0.998387     1.000000
```

Figura-6 *Heatmap*

Mostriamo il grafico inerente alla distribuzione dei dati e la retta di regressione mettendo nell'asse x il numero di tamponi e nell'asse y il numero totale di casi. Utilizziamo la libreria *seaborn*[10]. basata su *matplotlib* per visualizzare il grafico.

```
In [6]: sns.regplot('tamponi','totale_casi',df)
        plt.grid()
        plt.show()
```

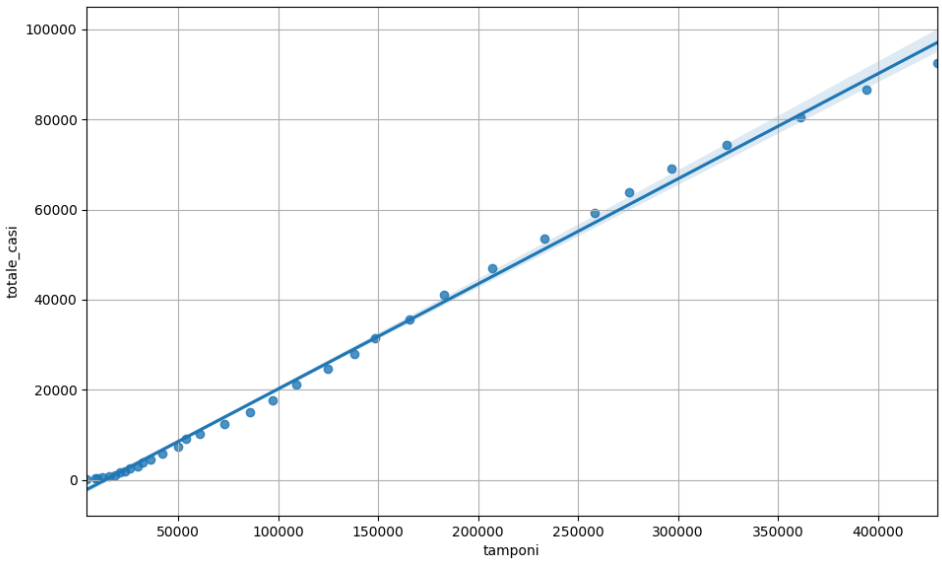


Figura -7 *Retta di Regressione*

3.3.2. Totale attualmente positivi e totale ospedalizzati

Analizziamo adesso i casi di totale attualmente positivi e totale ospedalizzati misurando coefficiente di correlazione, coefficiente angolare e intercetta con le istruzioni sopra descritte.

```
In [7]: df_2=df.loc[:,['totale_attualmente_positivi', 'totale_ospedalizzati']]
        cf2= df_2.corr()
        X=df.loc[:,['totale_attualmente_positivi']]
        Y=df.loc[:,['totale_ospedalizzati']]
        reg = LinearRegression()
        reg.fit(X, Y)
        print("intercetta q:",reg.intercept_)
        print("coefficiente angolare m: ",reg.coef_)
        print("\n")
        print("coefficiente di correlazione")
        print(cf2,"\n")
```

```
intercetta q: [1056.42897504]
coefficiente angolare m:  [[0.45360954]]


coefficiente di correlazione
                           totale_attualmente_positivi  totale_ospedalizzati
totale_attualmente_positivi                     1.000000              0.994038
totale_ospedalizzati                            0.994038              1.000000
```

Mostriamo i dati nel grafico insieme alla retta di regressione per i casi totale attualmente positivi nell'asse x ed il totale ospedalizzati nell'asse y.

```
In [8]: sns.regplot('totale_attualmente_positivi','totale_ospedalizzati',df)
        plt.grid()
        plt.show()
```

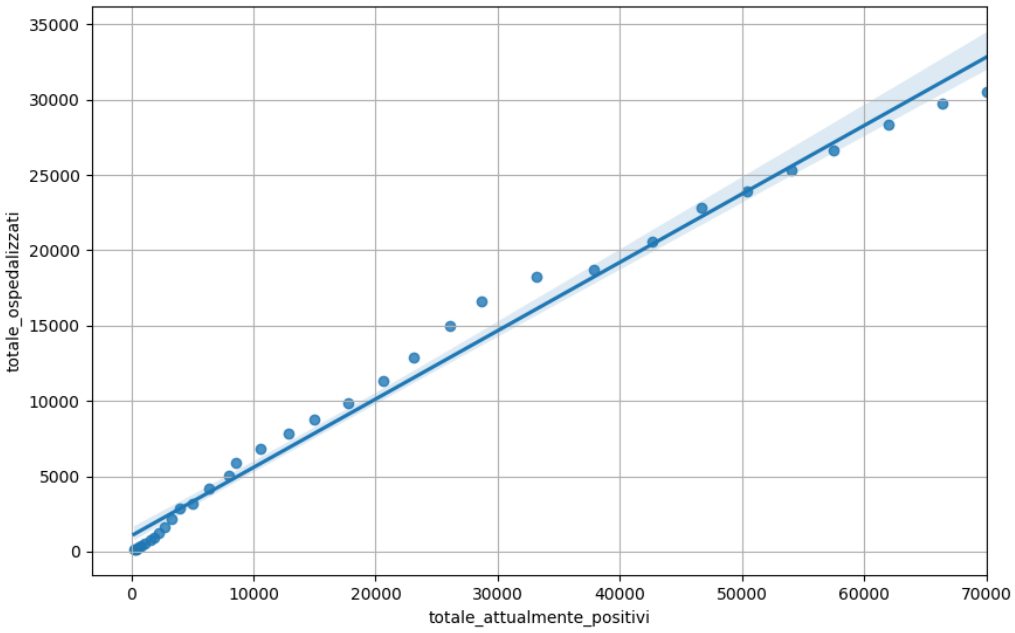


Figura -8 *Retta di Regressione*

3.3.3. Totale ospedalizzati e terapia intensiva

Analizziamo i casi di totale ospedalizzati e terapia intensiva misurando, anche in questo caso, il coefficiente di correlazione, intercetta e coefficiente angolare.

```
In [17]: df_3=df.loc[:,['totale_ospedalizzati','terapia_intensiva']]
         cf3= df_3.corr()
         X=df.loc[:,['totale_ospedalizzati']]
         Y=df.loc[:,['terapia_intensiva']]
         reg = LinearRegression()
         reg.fit(X, Y)
         print("intercetta q:",reg.intercept_)
         print("coefficiente angolare m: ",reg.coef_)
         print("\n")
         print("coefficiente di correlazione")
         print(cf3,"\n")
```

```
intercetta q: [-693.99605992]
coefficiente angolare m:  [[7.74849133]]


coefficiente di correlazione
                    terapia_intensiva  totale_ospedalizzati
terapia_intensiva            1.000000              0.997909
totale_ospedalizzati         0.997909              1.000000
```

Visualizziamo la retta di regressione e le distribuzioni dei dati mettendo nell'asse x i totali ospedalizzati e nell'asse y i casi di terapia intensiva.

```
In [10]: sns.regplot('totale_ospedalizzati','terapia_intensiva',df)
         plt.grid()
         plt.show()
```

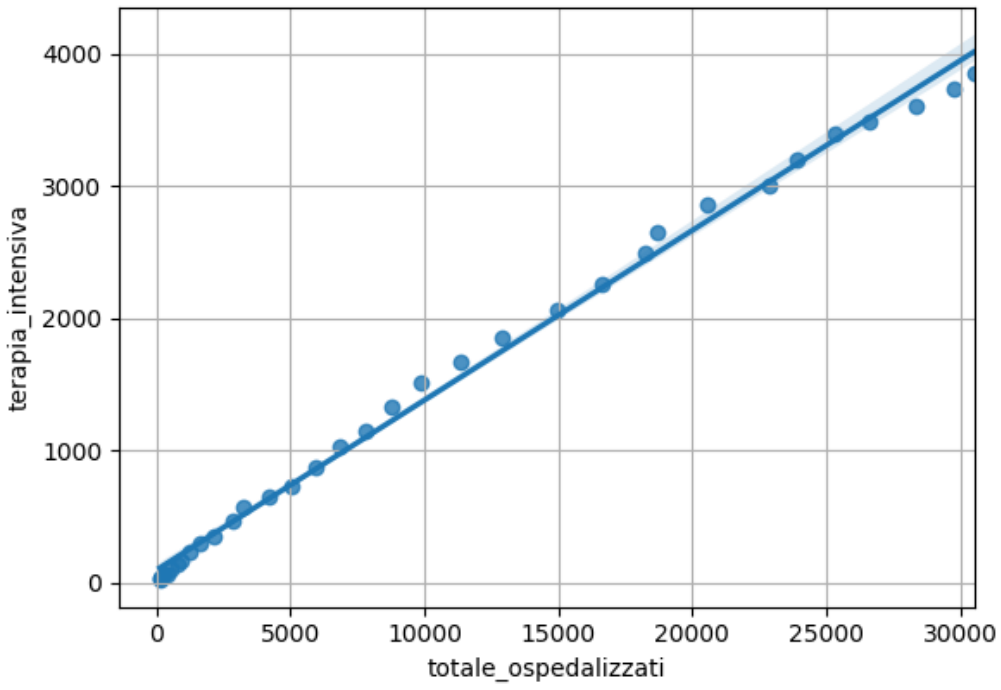


Figura -9 *Retta di Regressione*

3.3.4. Totale casi e dimessi guariti

Studiamo adesso i casi totali in relazione con i dimessi guariti misurando, anche in questo caso il coefficiente angolare, di correlazione e l'intercetta.

```
In [22]: df_4=df.loc[:,['totale_casi', 'dimessi_guariti']]
         cf4= df_4.corr()
         X=df.loc[:,['totale_casi']]
         Y=df.loc[:,['dimessi_guariti']]
         reg = LinearRegression()
         reg.fit(X, Y)
         print("intercetta q:",reg.intercept_)
         print("coefficiente angolare m: ",reg.coef_)
         print()
         print("coefficente di correlazione")
         print(cf4,"\n")
```

```
intercetta q: [-333.09833062]
coefficiente angolare m:  [[0.12691436]]

coefficente di correlazione
                 totale_casi  dimessi_guariti
totale_casi         1.000000         0.995121
dimessi_guariti     0.995121         1.000000
```

Visualizziamo la distribuzione dei punti (casi) e la retta di regressione mettendo nell'asse x i totali casi e nell'asse y i dimessi guariti.

```
In [12]: sns.regplot('totale_casi','dimessi_guariti',df)
         plt.grid()
         plt.show()
```

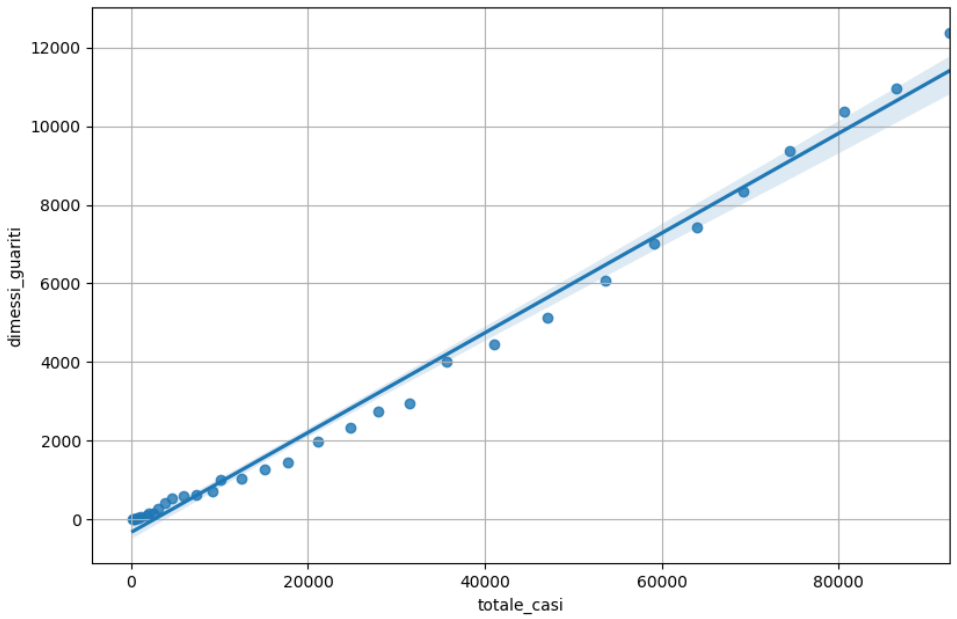


Figura -10 *Retta di Regressione*

3.3.5. Totale casi e deceduti

Concludiamo la nostra analisi di regressione mettendo in relazione i totali casi e il numero di deceduti utilizzando le medesime istruzioni.

```
In [25]: df_5=df.loc[:,['totale_casi', 'deceduti']]
         cf5= df_5.corr()
         X=df.loc[:,['totale_casi']]
         Y=df.loc[:,['deceduti']]
         reg = LinearRegression()
         reg.fit(X, Y)
         print("intercetta q:",reg.intercept_)
         print("coefficiente angolare m: ",reg.coef_)
         print()
         print("coefficiente di correlazione")
         print(cf5,"\n")
```

```
intercetta q: [-354.6695617]
coefficiente angolare m:  [[0.10342561]]

coefficiente di correlazione
             totale_casi  deceduti
totale_casi      1.00000   0.99453
deceduti         0.99453   1.00000
```

Mostriamo nel grafico la retta di regressione e la distribuzione dei casi mettendo nell'asse x i casi totali e nell'asse y i casi deceduti.

```
In [14]: sns.regplot('totale_casi','deceduti',df)
         plt.grid()
         plt.show()
```

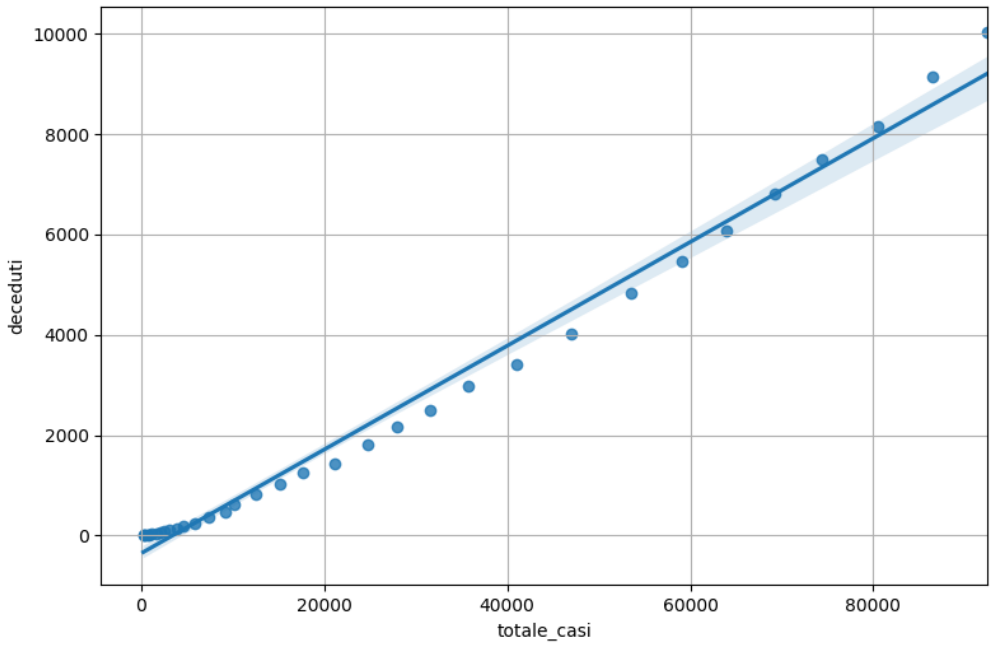


Figura-11 *Retta di Regressione*

3.4. BAR PLOT

Infine, andremo a mostrare l'andamento dei casi regione per regione attraverso un diagramma a barre. Preleviamo i dati dalla directory dati-regione del dataset[3].

```
In [15]: #analizziamo il dataset della regione civile alla directory dati-regioni
         url1= "https://raw.githubusercontent.com/pcm-dpc/COVID-19/master/dati-regioni/dpc-covid19-ita-regioni-2
         data_frame_regione = pd.read_csv(url1)
         data_frame_regione
```

Visualizzeremo il grafico attraverso la funzione *.barplot()*[11] appartenente alla libreria *seaborn*[10] (la quale ci consente di mostrare le stime dei punti e gli intervalli di confidenza come barre rettangolari). Nel nostro caso specifico la funzione *.barplot()* ci permette di rappresentare graficamente il numero di contagi per regione, ovvero ogni modalità della variabile xi viene rappresentata da un rettangolo avente base costante, la cui altezza (o lunghezza) è proporzionale alla frequenza.

```
In [26]: sns.set(style="whitegrid")
         f, ax = plt.subplots(figsize=(27,9))
         tot_casi = data_frame_regione.sort_values('totale_casi',ascending=False)
         sns.set_color_codes("colorblind")
         sns.barplot( x="totale_casi", y="denominazione_regione", data=tot_casi,label="Total", color="b")
         plt.grid()
         plt.show()
```

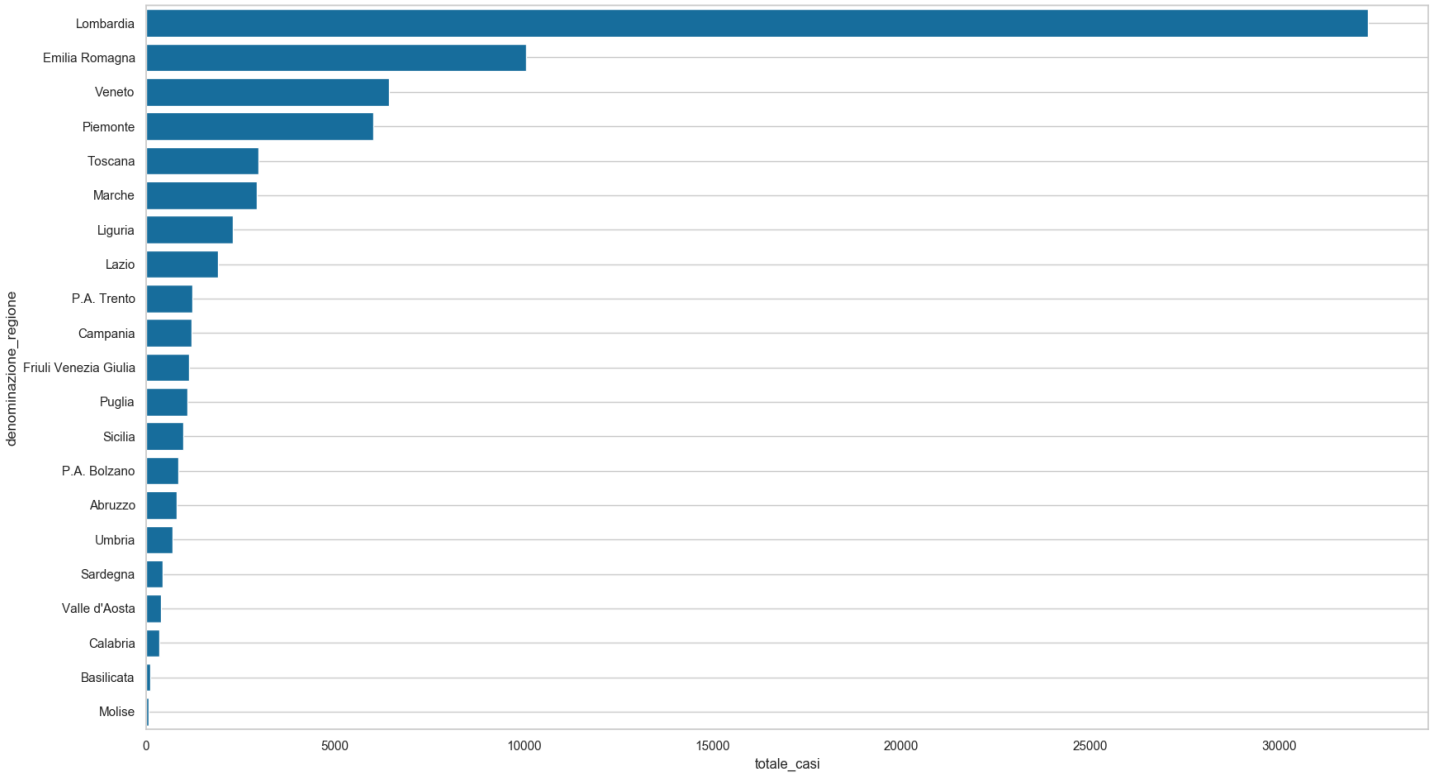


Figura -12 *Bar plot*

4. RISULTATI E CONCLUSIONI

L'obiettivo di questa indagine era quello di verificare le correlazioni tra i dati elaborati dai quali si evince:

- l'aumento dei contagi non è esattamente esponenziale.
- la matrice di correlazione ha evidenziato come il coefficiente di Pearson assuma valori abbastanza alti, in quanto il valore minimo che viene riscontrato è di 0,83. Quindi, da quanto detto in precedenza, avremo una forte correlazione tra i vari campi del dataset (Figura-6);

- le coppie studiate tramite la tecnica della regressione lineare, hanno enfatizzato quanto riportato dalla matrice di correlazione, infatti, tutte le rette possiedono un valore $r_{xy} > 0$ che le rende ascendenti (Figura 7-8-9-10-11);

- infine, il bar plot ha rilevato che le regioni più colpite dal COVID-19 sono la Lombardia, Emilia-Romagna e Veneto in data odierna (Figura-12).

In conclusione, si potrebbero effettuare ulteriori studi futuri per migliorare le analisi statistiche dei dati. In particolare, si potrebbero attenzionare i casi di terapia intensiva con l'obiettivo di monitorare il sistema sanitario soprattutto in regioni come la Lombardia maggiormente colpite dal contagio.

Inoltre, tramite algoritmi di Machine Learning con la tecnica della *prediction*, è possibile preventivamente calcolare il numero dei contagi giornalieri al fine di adottare adeguate misure di contenimento, prevenzione e disposizione di sufficienti servizi sanitari volti ad affrontare l'emergenza COVID-19.

5. REFERENZE

- [1] https://www.python.org/
- [2] https://matplotlib.org/
- [3] https://github.com/pcm-dpc/covid-19
- [4] Metodi Matematici e Statistici, Prof. Orazio Muscato
- [5] https://seaborn.pydata.org/generated/seaborn.heatmap.html
- [6] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html
- [7] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
- [8] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
- [9] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html
- [10] https://seaborn.pydata.org/
- [11] https://seaborn.pydata.org/generated/seaborn.barplot.html
- [12] https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html
- [13] https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.pyplot.loglog.html
- [14] https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.pyplot.semilogy.html