```
from google.colab import drive


drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True)
```

```
import pandas as pd


content = """Date
Location_ISO_Code
Location
New_Cases
New_Deaths
New_Recovered
New_Active_Cases
Total_Cases
Total_Deaths
Total_Recovered
Total_Active_Cases
Location_Level
City_or_Regency
Province
Country
Continent
Island
Time_Zone
Special_Status
Total_Regencies
Total_Cities
Total_Districts
Total_Urban_Villages
Total_Rural_Villages
Area_(km2)
Population
```

```
Population
Population_Density
Longitude
Latitude
New_Cases_per_Million
Total_Cases_per_Million
New_Deaths_per_Million
Total_Deaths_per_Million
Case_Fatality_Rate
Case_Recovered_Rate
Growth_Factor_of_New_Cases
Growth_Factor_of_New_Deaths"""
columns_list = content.split("\n")


df = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/covid_19_indonesia_time_series_all.csv", header=0, names=columns_list, index
df.head()
```

	Date	Location_ISO_Code	Location	New_Cases	New_Deaths	New_Recovered	New_Act
0	3/1/2020	ID-JK	DKI Jakarta	2	0	0	
1	3/2/2020	ID-JK	DKI Jakarta	2	0	0	
2	3/2/2020	IDN	Indonesia	2	0	0	
3	3/2/2020	ID-RI	Riau	1	0	0	
4	3/3/2020	ID-JK	DKI Jakarta	2	0	0	

```
df = df.set_index('Location')
df.head()
```

Location	Date	Location_ISO_Code	New_Cases	New_Deaths	New_Recovered	New_Activ
DKI Jakarta	3/1/2020	ID-JK	2	0	0	
DKI Jakarta	3/2/2020	ID-JK	2	0	0	
Indonesia	3/2/2020	IDN	2	0	0	
Riau	3/2/2020	ID-RI	1	0	0	
DKI Jakarta	3/3/2020	ID-JK	2	0	0	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 20816 entries, DKI Jakarta to Sumatera Utara
Data columns (total 36 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   Date                        20816 non-null  object 
 1   Location_ISO_Code           20816 non-null  object 
 2   New_Cases                   20816 non-null  int64  
 3   New_Deaths                  20816 non-null  int64  
 4   New_Recovered               20816 non-null  int64  
 5   New_Active_Cases            20816 non-null  int64  
 6   Total_Cases                 20816 non-null  int64  
 7   Total_Deaths                20816 non-null  int64  
 8   Total_Recovered             20816 non-null  int64  
 9   Total_Active_Cases          20816 non-null  int64  
 10  Location_Level              20816 non-null  object 
 11  City_or_Regency             0 non-null      float64
 12  Province                    20202 non-null  object 
 13  Country                     20816 non-null  object 
 14  Continent                   20816 non-null  object 
 15  Island                      20202 non-null  object 
 16  Time_Zone                   20202 non-null  object 
```

```
 17  Special_Status              2988 non-null   object
 18  Total_Regencies             20816 non-null  int64
 19  Total_Cities                20228 non-null  float64
 20  Total_Districts             20816 non-null  int64
 21  Total_Urban_Villages        20226 non-null  float64
 22  Total_Rural_Villages        20201 non-null  float64
 23  Area_(km2)                  20816 non-null  int64
 24  Population                  20816 non-null  int64
 25  Population_Density          20816 non-null  float64
 26  Longitude                   20816 non-null  float64
 27  Latitude                    20816 non-null  float64
 28  New_Cases_per_Million       20816 non-null  float64
 29  Total_Cases_per_Million     20816 non-null  float64
 30  New_Deaths_per_Million      20816 non-null  float64
 31  Total_Deaths_per_Million    20816 non-null  float64
 32  Case_Fatality_Rate          20816 non-null  float64
 33  Case_Recovered_Rate         20816 non-null  object
 34  Growth_Factor_of_New_Cases  20816 non-null  object
 35  Growth_Factor_of_New_Deaths 19709 non-null  float64
dtypes: float64(13), int64(12), object(11)
memory usage: 5.9+ MB
```

Get a view feature

```
df = df[['Date', 'Location_ISO_Code', 'New_Cases', 'New_Deaths', 'Total_Cases', 'Total_Deaths', 'Total_Recovered', 'New_Active_Cases'
df.head()
```

```
                Date  Location_ISO_Code  New_Cases  New_Deaths  Total_Cases  Total_Deaths
Location
```

```
# convert Date column to date type
df["Date"] = pd.to_datetime(df["Date"])
```

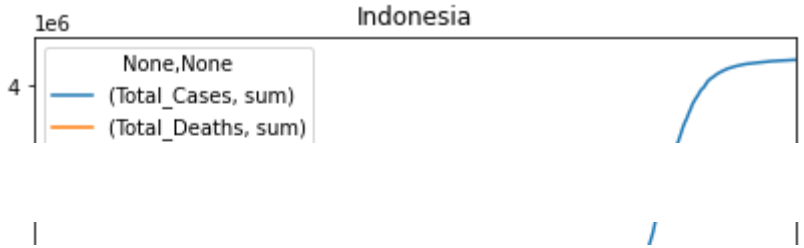
Visualization

```
import matplotlib.pyplot as plt
%matplotlib inline

#IDN
ConfirmedCases_date_IDN= df[df['Location_ISO_Code']=='IDN'].groupby(['Date']).agg({'Total_Cases':['sum']})
fatalities_date_IDN = df[df['Location_ISO_Code']=='IDN'].groupby(['Date']).agg({'Total_Deaths':['sum']})
total_date_IDN= ConfirmedCases_date_IDN.join(fatalities_date_IDN)


plt.figure(figsize=(15,10))
plt.subplot(2, 2, 1)
total_date_IDN.plot(ax=plt.gca(), title='Indonesia')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f03f58e3210>
```



Preprocessing

```
data1 = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/covid_19_indonesia_time_series_all.csv", header=0, names=columns_list, in
data1 = data1.set_index('Location')
data1.head()
```

Location	Date	Location_ISO_Code	New_Cases	New_Deaths	New_Recovered	New_Activ
DKI Jakarta	3/1/2020	ID-JK	2	0	0	
DKI Jakarta	3/2/2020	ID-JK	2	0	0	
Indonesia	3/2/2020	IDN	2	0	0	
Riau	3/2/2020	ID-RI	1	0	0	
DKI Jakarta	3/3/2020	ID-JK	2	0	0	

```
import numpy as np

# preprocessing replace the nan data
data1= data1.replace([np.inf, -np.inf], np.nan)
```

```
# preprocessig fill the nan data
data1 = data1.fillna(0)
data1
```

Location	Date	Location_ISO_Code	New_Cases	New_Deaths	New_Recovered	New_Activ
DKI Jakarta	3/1/2020	ID-JK	2	0	0	
DKI Jakarta	3/2/2020	ID-JK	2	0	0	
Indonesia	3/2/2020	IDN	2	0	0	
Riau	3/2/2020	ID-RI	1	0	0	
DKI Jakarta	3/3/2020	ID-JK	2	0	0	
...	...	...	...	...	...	
Sulawesi Tengah	11/5/2021	ID-ST	11	0	0	
Sulawesi Utara	11/5/2021	ID-SA	3	1	1	
Sumatera Barat	11/5/2021	ID-SB	1	0	3	
Sumatera Selatan	11/5/2021	ID-SS	1	0	0	
Sumatera Utara	11/5/2021	ID-SU	7	1	3	

20816 rows × 36 columns

```
# Convert sting to numeric LabelEncoder
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
def FunLabelEncoder(df):
    for c in df.columns:
        if df.dtypes[c] == object:
            le.fit(df[c].astype(str))
            df[c] = le.transform(df[c].astype(str))
    return df


data1 = FunLabelEncoder(data1)
data1.info()
# df1.iloc[235:300,:]
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 20816 entries, DKI Jakarta to Sumatera Utara
Data columns (total 36 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   Date                         20816 non-null  int64  
 1   Location_ISO_Code            20816 non-null  int64  
 2   New_Cases                    20816 non-null  int64  
 3   New_Deaths                   20816 non-null  int64  
 4   New_Recovered                20816 non-null  int64  
 5   New_Active_Cases             20816 non-null  int64  
 6   Total_Cases                  20816 non-null  int64  
 7   Total_Deaths                 20816 non-null  int64  
 8   Total_Recovered              20816 non-null  int64  
 9   Total_Active_Cases           20816 non-null  int64  
 10  Location_Level               20816 non-null  int64  
 11  City_or_Regency              20816 non-null  float64
 12  Province                     20816 non-null  int64  
 13  Country                      20816 non-null  int64  
 14  Continent                    20816 non-null  int64  
 15  Island                       20816 non-null  int64  
 16  Time_Zone                    20816 non-null  int64  
 17  Special_Status               20816 non-null  int64  
 18  Total_Regencies              20816 non-null  int64  
```

```
 19  Total_Cities                20816 non-null  float64
 20  Total_Districts             20816 non-null  int64
 21  Total_Urban_Villages        20816 non-null  float64
 22  Total_Rural_Villages        20816 non-null  float64
 23  Area_(km2)                  20816 non-null  int64
 24  Population                  20816 non-null  int64
 25  Population_Density          20816 non-null  float64
 26  Longitude                   20816 non-null  float64
 27  Latitude                    20816 non-null  float64
 28  New_Cases_per_Million       20816 non-null  float64
 29  Total_Cases_per_Million     20816 non-null  float64
 30  New_Deaths_per_Million      20816 non-null  float64
 31  Total_Deaths_per_Million    20816 non-null  float64
 32  Case_Fatality_Rate          20816 non-null  float64
 33  Case_Recovered_Rate         20816 non-null  int64
 34  Growth_Factor_of_New_Cases  20816 non-null  int64
 35  Growth_Factor_of_New_Deaths 20816 non-null  float64
dtypes: float64(13), int64(23)
memory usage: 5.9+ MB
```

Splitting Data

```
from sklearn.model_selection import train_test_split

Y = data1['New_Cases']
X = data1.drop(columns=['New_Cases'])

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=9)


print('X train shape: ', X_train.shape)
print('Y train shape: ', Y_train.shape)
print('X test shape: ', X_test.shape)
print('Y test shape: ', Y_test.shape)
```

```
X train shape:  (16652, 35)
Y train shape:  (16652,)
X test shape:  (4164, 35)
Y test shape:  (4164,)
```

Modellig

```
from sklearn.tree import DecisionTreeClassifier

# We define the model
dtcla = DecisionTreeClassifier(random_state=None)

# We train model
dtcla.fit(X_train, Y_train)

# We predict target values
Y_predict = dtcla.predict(X_test)


#Test
X_test
```

Location	Date	Location_ISO_Code	New_Deaths	New_Recovered	New_Active_Cases	Tota
Indonesia	177	34	268	7261	703	
Sulawesi Utara	611	25	0	18	3	
Riau	251	24	0	1	0	
DKI Jakarta	511	9	19	589	-95	
Bali	423	1	0	1	32	
...	...	...	...	...	...	
DKI Jakarta	34	9	3	69	68	
Sulawesi Selatan	132	28	4	96	212	

Predict and Evaluation

```
from sklearn.model_selection import train_test_split

Y1 = data1['New_Deaths']
X1 = data1.drop(columns=['New_Deaths'])

X1_train, X1_test, Y1_train, Y1_test = train_test_split(X1, Y1, test_size=0.2, random_state=9)


print('X1 train shape: ', X1_train.shape)
print('Y1 train shape: ', Y1_train.shape)
```

```
print('X1 test shape: ', X1_test.shape)
print('Y1 test shape: ', Y1_test.shape)
```

```
X1 train shape:  (16652, 35)
Y1 train shape:  (16652,)
X1 test shape:  (4164, 35)
Y1 test shape:  (4164,)
```

```
from sklearn.tree import DecisionTreeClassifier

# We define the model
dtcla = DecisionTreeClassifier(random_state=None)

# We train model
dtcla.fit(X1_train, Y1_train)

# We predict target values
Y1_predict = dtcla.predict(X1_test)


#Test
data1=X1_test
data1
```

Location	Date	Location_ISO_Code	New_Cases	New_Recovered	New_Active_Cases	Total
Indonesia	177	34	8232	7261	703	1
Sulawesi Utara	611	25	21	18	3	
Riau	251	24	1	1	0	
DKI Jakarta	511	9	513	589	-95	
Bali	423	1	33	1	32	
...	...	...	...	...	...	
DKI Jakarta	34	9	140	69	68	

```
#Create a  DataFrame
submission = pd.DataFrame({'New_Cases':Y_predict,'New_Deaths':Y1_predict})


#Visualize the first 100 rows
submission.head(100)
```

	New_Cases	New_Deaths
0	6412	267
1	16	0
2	1	0
3	551	19
4	33	0

```
#Convert DataFrame to a csv file that can be uploaded
#This is saved in the same directory as your notebook
filename = 'submission.csv'

submission.to_csv(filename,index=False)

print('Saved file: ' + filename)
```

```
                precision    recall  f1-score   support

             0       0.20      0.93      0.33       374
             1       0.02      0.08      0.03       141
             2       0.01      0.03      0.02       116
             3       0.00      0.01      0.01        85
             4       0.00      0.00      0.00        63
             5       0.01      0.02      0.01        60
             6       0.00      0.00      0.00        78
             7       0.00      0.00      0.00        46
             8       0.00      0.00      0.00        59
             9       0.00      0.00      0.00        48
            10       0.00      0.00      0.00        44
            11       0.00      0.00      0.00        54
            12       0.00      0.00      0.00        44
            13       0.00      0.00      0.00        43
            14       0.00      0.00      0.00        44
            15       0.00      0.00      0.00        43
            16       0.00      0.00      0.00        32
            17       0.00      0.00      0.00        36
```

18	0.00	0.00	0.00	26
19	0.00	0.00	0.00	28
20	0.00	0.00	0.00	38
21	0.00	0.00	0.00	37
22	0.00	0.00	0.00	31
23	0.00	0.00	0.00	35
24	0.00	0.00	0.00	24
25	0.00	0.00	0.00	22
26	0.00	0.00	0.00	25
27	0.00	0.00	0.00	28
28	0.00	0.00	0.00	29
29	0.00	0.00	0.00	16
30	0.00	0.00	0.00	23
31	0.00	0.00	0.00	19
32	0.00	0.00	0.00	19
33	0.00	0.00	0.00	35
34	0.00	0.00	0.00	20
35	0.00	0.00	0.00	16
36	0.00	0.00	0.00	15
37	0.00	0.00	0.00	19
38	0.00	0.00	0.00	12
39	0.00	0.00	0.00	24
40	0.00	0.00	0.00	18
41	0.00	0.00	0.00	14
42	0.00	0.00	0.00	12
43	0.00	0.00	0.00	14
44	0.00	0.00	0.00	14
45	0.00	0.00	0.00	22
46	0.00	0.00	0.00	13
47	0.00	0.00	0.00	15
48	0.00	0.00	0.00	16
49	0.00	0.00	0.00	13
50	0.00	0.00	0.00	16
51	0.00	0.00	0.00	9
52	0.00	0.00	0.00	14
53	0.00	0.00	0.00	5
54	0.00	0.00	0.00	17