In [1]:
```
import sys           # Read system parameters.
import pandas as pd  # Manipulate and analyze data.
import sqlite3    # Manage SQL databases.
import sys                                                  # Read system parameters.
import numpy as np                                          # Work with multi-dimensional arrays.
import pandas as pd                                         # Manipulate and analyze data.
import scipy as sp                                          # Apply advanced mathematical functions.
from scipy import stats
import matplotlib                                           # Create and format charts.
import matplotlib.pyplot as plt
import seaborn as sns                                       # Make charting easier.
import sklearn                                              # Train and evaluate machine learning models.
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import category_encoders as ce                              # Encode data.
import warnings                                             # Suppress warnings.
warnings.filterwarnings('ignore')

# Summarize software libraries used.
print('Libraries used in this project:')
print('- Python {}'.format(sys.version))
print('- NumPy {}'.format(np.__version__))
print('- pandas {}'.format(pd.__version__))
print('- SciPy {}'.format(sp.__version__))
print('- Matplotlib {}'.format(matplotlib.__version__))
print('- Seaborn {}'.format(sns.__version__))
print('- scikit-learn {}'.format(sklearn.__version__))

# Summarize software libraries used.
print('Libraries used in this project:')
print('- Python {}'.format(sys.version))
print('- pandas {}'.format(pd.__version__))
print('- sqlite3 {}'.format(sqlite3.sqlite_version))
```

```
Libraries used in this project:
- Python 3.9.13 (main, Aug 25 2022, 23:51:50) [MSC v.1916 64 bit (AMD64)]
- NumPy 1.21.5
- pandas 1.4.4
- SciPy 1.9.1
- Matplotlib 3.5.2
- Seaborn 0.11.2
- scikit-learn 1.0.2
Libraries used in this project:
- Python 3.9.13 (main, Aug 25 2022, 23:51:50) [MSC v.1916 64 bit (AMD64)]
- pandas 1.4.4
- sqlite3 3.39.3
```

In [2]:
```
covid_data = pd.read_csv(r"C:\Users\Haya\Downloads\covid-data.csv")
```

In [3]:
```
covid_data.head(n=5)
```

Out[3]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoot
0	AFG	Asia	Afghanistan	2020-01-03	NaN	0.0	NaN	NaN	0.0	N
1	AFG	Asia	Afghanistan	2020-01-04	NaN	0.0	NaN	NaN	0.0	N
2	AFG	Asia	Afghanistan	2020-01-05	NaN	0.0	NaN	NaN	0.0	N
3	AFG	Asia	Afghanistan	2020-01-06	NaN	0.0	NaN	NaN	0.0	N
4	AFG	Asia	Afghanistan	2020-01-07	NaN	0.0	NaN	NaN	0.0	N

5 rows × 67 columns

In [4]:
```
covid_data[covid_data.life_expectancy>80]
```

Out[4]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_sm
6400	AND	Europe	Andorra	2020-01-03	NaN	0.0	NaN	NaN	0.0	
6401	AND	Europe	Andorra	2020-01-04	NaN	0.0	NaN	NaN	0.0	
6402	AND	Europe	Andorra	2020-01-05	NaN	0.0	NaN	NaN	0.0	
6403	AND	Europe	Andorra	2020-01-06	NaN	0.0	NaN	NaN	0.0	
6404	AND	Europe	Andorra	2020-01-07	NaN	0.0	NaN	NaN	0.0	
...	...	...	...	...	...	...	...	...	...	
307111	VIR	North America	United States Virgin Islands	2023-07-01	25245.0	0.0	4.857	131.0	0.0	
307112	VIR	North America	United States Virgin Islands	2023-07-02	25270.0	25.0	3.571	132.0	1.0	
307113	VIR	North America	United States Virgin Islands	2023-07-03	25270.0	0.0	3.571	132.0	0.0	
307114	VIR	North America	United States Virgin Islands	2023-07-04	25270.0	0.0	3.571	132.0	0.0	
307115	VIR	North America	United States Virgin Islands	2023-07-05	25270.0	0.0	3.571	132.0	0.0	

69854 rows × 67 columns

In [5]:
```
pd.crosstab(covid_data.life_expectancy,covid_data.location)
```

Out[5]:

location	Afghanistan	Albania	Algeria	American Samoa	Andorra	Angola	Anguilla	Antigua and Barbuda	Argentina	Armenia	...	Vanuatu	Vatican
life_expectancy													
53.28	0	0	0	0	0	0	0	0	0	0	...	0	0
54.24	0	0	0	0	0	0	0	0	0	0	...	0	0
54.33	0	0	0	0	0	0	0	0	0	0	...	0	0
54.69	0	0	0	0	0	0	0	0	0	0	...	0	0
54.70	0	0	0	0	0	0	0	0	0	0	...	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
84.24	0	0	0	0	0	0	0	0	0	0	...	0	0
84.63	0	0	0	0	0	0	0	0	0	0	...	0	0
84.86	0	0	0	0	0	0	0	0	0	0	...	0	0
84.97	0	0	0	0	0	0	0	0	0	0	...	0	0
86.75	0	0	0	0	0	0	0	0	0	0	...	0	0

220 rows × 234 columns

In [6]:
```
covid_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 323661 entries, 0 to 323660
Data columns (total 67 columns):
 #   Column                                Non-Null Count   Dtype  
---  ------                                --------------   -----  
 0   iso_code                              323661 non-null  object 
 1   continent                             308301 non-null  object 
 2   location                              323661 non-null  object 
 3   date                                  323661 non-null  object 
 4   total_cases                           286549 non-null  float64
 5   new_cases                             314782 non-null  float64
 6   new_cases_smoothed                    313523 non-null  float64
 7   total_deaths                          265511 non-null  float64
 8   new_deaths                            314824 non-null  float64
 9   new_deaths_smoothed                   313594 non-null  float64
 10  total_cases_per_million               286549 non-null  float64
 11  new_cases_per_million                 314782 non-null  float64
 12  new_cases_smoothed_per_million        313523 non-null  float64
 13  total_deaths_per_million              265511 non-null  float64
 14  new_deaths_per_million                314824 non-null  float64
 15  new_deaths_smoothed_per_million       313594 non-null  float64
 16  reproduction_rate                     184817 non-null  float64
 17  icu_patients                          36772 non-null   float64
 18  icu_patients_per_million              36772 non-null   float64
 19  hosp_patients                         37593 non-null   float64
 20  hosp_patients_per_million             37593 non-null   float64
 21  weekly_icu_admissions                 9720 non-null    float64
 22  weekly_icu_admissions_per_million     9720 non-null    float64
 23  weekly_hosp_admissions                22403 non-null   float64
 24  weekly_hosp_admissions_per_million    22403 non-null   float64
 25  total_tests                           79387 non-null   float64
 26  new_tests                             75403 non-null   float64
 27  total_tests_per_thousand              79387 non-null   float64
 28  new_tests_per_thousand                75403 non-null   float64
 29  new_tests_smoothed                    103965 non-null  float64
 30  new_tests_smoothed_per_thousand       103965 non-null  float64
 31  positive_rate                         95927 non-null   float64
 32  tests_per_case                        94348 non-null   float64
 33  tests_units                           106788 non-null  object 
 34  total_vaccinations                    76542 non-null   float64
 35  people_vaccinated                     73296 non-null   float64
 36  people_fully_vaccinated               69822 non-null   float64
 37  total_boosters                        45013 non-null   float64
 38  new_vaccinations                      63002 non-null   float64
```

```
 39  new_vaccinations_smoothed                   172825 non-null  float64
 40  total_vaccinations_per_hundred              76542 non-null   float64
 41  people_vaccinated_per_hundred               73296 non-null   float64
 42  people_fully_vaccinated_per_hundred         69822 non-null   float64
 43  total_boosters_per_hundred                  45013 non-null   float64
 44  new_vaccinations_smoothed_per_million       172825 non-null  float64
 45  new_people_vaccinated_smoothed              172624 non-null  float64
 46  new_people_vaccinated_smoothed_per_hundred  172624 non-null  float64
 47  stringency_index                            197651 non-null  float64
 48  population_density                          274674 non-null  float64
 49  median_age                                  255459 non-null  float64
 50  aged_65_older                               246514 non-null  float64
 51  aged_70_older                               252899 non-null  float64
 52  gdp_per_capita                              250354 non-null  float64
 53  extreme_poverty                             161284 non-null  float64
 54  cardiovasc_death_rate                       250868 non-null  float64
 55  diabetes_prevalence                         263639 non-null  float64
 56  female_smokers                              188164 non-null  float64
 57  male_smokers                                185604 non-null  float64
 58  handwashing_facilities                      122882 non-null  float64
 59  hospital_beds_per_thousand                  221444 non-null  float64
 60  life_expectancy                             297699 non-null  float64
 61  human_development_index                     243159 non-null  float64
 62  population                                  323661 non-null  float64
 63  excess_mortality_cumulative_absolute        11245 non-null   float64
 64  excess_mortality_cumulative                 11245 non-null   float64
 65  excess_mortality                            11245 non-null   float64
 66  excess_mortality_cumulative_per_million     11245 non-null   float64
dtypes: float64(62), object(5)
memory usage: 165.4+ MB
```

In [7]:
```
covid_data_3 = covid_data.copy()

covid_data_3['date'] = \
pd.to_datetime(covid_data_3['date'],
               format = '%Y-%m-%d')
```

In [8]:
```
covid_data_3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 323661 entries, 0 to 323660
Data columns (total 67 columns):
 #   Column                                Non-Null Count   Dtype
---  ------                                --------------   -----
 0   iso_code                              323661 non-null  object
 1   continent                             308301 non-null  object
 2   location                              323661 non-null  object
 3   date                                  323661 non-null  datetime64[ns]
 4   total_cases                           286549 non-null  float64
 5   new_cases                             314782 non-null  float64
 6   new_cases_smoothed                    313523 non-null  float64
 7   total_deaths                          265511 non-null  float64
 8   new_deaths                            314824 non-null  float64
 9   new_deaths_smoothed                   313594 non-null  float64
 10  total_cases_per_million               286549 non-null  float64
 11  new_cases_per_million                 314782 non-null  float64
 12  new_cases_smoothed_per_million        313523 non-null  float64
 13  total_deaths_per_million              265511 non-null  float64
 14  new_deaths_per_million                314824 non-null  float64
 15  new_deaths_smoothed_per_million       313594 non-null  float64
 16  reproduction_rate                     184817 non-null  float64
 17  icu_patients                          36772 non-null   float64
 18  icu_patients_per_million              36772 non-null   float64
 19  hosp_patients                         37593 non-null   float64
 20  hosp_patients_per_million             37593 non-null   float64
 21  weekly_icu_admissions                 9720 non-null    float64
 22  weekly_icu_admissions_per_million     9720 non-null    float64
 23  weekly_hosp_admissions                22403 non-null   float64
 24  weekly_hosp_admissions_per_million    22403 non-null   float64
 25  total_tests                           79387 non-null   float64
 26  new_tests                             75403 non-null   float64
 27  total_tests_per_thousand              79387 non-null   float64
 28  new_tests_per_thousand                75403 non-null   float64
 29  new_tests_smoothed                    103965 non-null  float64
 30  new_tests_smoothed_per_thousand       103965 non-null  float64
 31  positive_rate                         95927 non-null   float64
 32  tests_per_case                        94348 non-null   float64
 33  tests_units                           106788 non-null  object
 34  total_vaccinations                    76542 non-null   float64
 35  people_vaccinated                     73296 non-null   float64
 36  people_fully_vaccinated               69822 non-null   float64
 37  total_boosters                        45013 non-null   float64
 38  new_vaccinations                      63002 non-null   float64
```

```
 39  new_vaccinations_smoothed                   172825 non-null  float64
 40  total_vaccinations_per_hundred              76542 non-null   float64
 41  people_vaccinated_per_hundred               73296 non-null   float64
 42  people_fully_vaccinated_per_hundred         69822 non-null   float64
 43  total_boosters_per_hundred                  45013 non-null   float64
 44  new_vaccinations_smoothed_per_million       172825 non-null  float64
 45  new_people_vaccinated_smoothed              172624 non-null  float64
 46  new_people_vaccinated_smoothed_per_hundred  172624 non-null  float64
 47  stringency_index                            197651 non-null  float64
 48  population_density                          274674 non-null  float64
 49  median_age                                  255459 non-null  float64
 50  aged_65_older                               246514 non-null  float64
 51  aged_70_older                               252899 non-null  float64
 52  gdp_per_capita                              250354 non-null  float64
 53  extreme_poverty                             161284 non-null  float64
 54  cardiovasc_death_rate                       250868 non-null  float64
 55  diabetes_prevalence                         263639 non-null  float64
 56  female_smokers                              188164 non-null  float64
 57  male_smokers                                185604 non-null  float64
 58  handwashing_facilities                      122882 non-null  float64
 59  hospital_beds_per_thousand                  221444 non-null  float64
 60  life_expectancy                             297699 non-null  float64
 61  human_development_index                     243159 non-null  float64
 62  population                                  323661 non-null  float64
 63  excess_mortality_cumulative_absolute        11245 non-null   float64
 64  excess_mortality_cumulative                 11245 non-null   float64
 65  excess_mortality                            11245 non-null   float64
 66  excess_mortality_cumulative_per_million     11245 non-null   float64
dtypes: datetime64[ns](1), float64(62), object(4)
memory usage: 165.4+ MB
```

In [9]:
```
duplicated_data = \
covid_data_3[covid_data_3.duplicated(keep=False)]

print('Number of rows with duplicated data:',duplicated_data.shape[0])
```

```
Number of rows with duplicated data: 0
```

In [10]:
```
covid_data_3.head()
```

Out[10]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoot
0	AFG	Asia	Afghanistan	2020-01-03	NaN	0.0	NaN	NaN	0.0	N
1	AFG	Asia	Afghanistan	2020-01-04	NaN	0.0	NaN	NaN	0.0	N
2	AFG	Asia	Afghanistan	2020-01-05	NaN	0.0	NaN	NaN	0.0	N
3	AFG	Asia	Afghanistan	2020-01-06	NaN	0.0	NaN	NaN	0.0	N
4	AFG	Asia	Afghanistan	2020-01-07	NaN	0.0	NaN	NaN	0.0	N

5 rows × 67 columns

In [11]:
```
text_data = covid_data_3[~covid_data_3["total_cases"].isnull()]
text_data.head(n=3)
```

Out[11]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoo
54	AFG	Asia	Afghanistan	2020-02-26	1.0	1.0	0.143	NaN	0.0	
55	AFG	Asia	Afghanistan	2020-02-27	1.0	0.0	0.143	NaN	0.0	
56	AFG	Asia	Afghanistan	2020-02-28	1.0	0.0	0.143	NaN	0.0	

3 rows × 67 columns

In [12]:
```
covid_data = text_data[~text_data["new_cases"].isnull()]
covid_data.head(n=5)
```

Out[12]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoo
54	AFG	Asia	Afghanistan	2020-02-26	1.0	1.0	0.143	NaN	0.0	
55	AFG	Asia	Afghanistan	2020-02-27	1.0	0.0	0.143	NaN	0.0	
56	AFG	Asia	Afghanistan	2020-02-28	1.0	0.0	0.143	NaN	0.0	
57	AFG	Asia	Afghanistan	2020-02-29	1.0	0.0	0.143	NaN	0.0	
58	AFG	Asia	Afghanistan	2020-03-01	1.0	0.0	0.143	NaN	0.0	

5 rows × 67 columns

In [13]:
```
covid_data_1 = covid_data[~covid_data["new_cases_smoothed"].isnull()]
covid_data_1.head(n=5)
```

Out[13]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoc
54	AFG	Asia	Afghanistan	2020-02-26	1.0	1.0	0.143	NaN	0.0	
55	AFG	Asia	Afghanistan	2020-02-27	1.0	0.0	0.143	NaN	0.0	
56	AFG	Asia	Afghanistan	2020-02-28	1.0	0.0	0.143	NaN	0.0	
57	AFG	Asia	Afghanistan	2020-02-29	1.0	0.0	0.143	NaN	0.0	
58	AFG	Asia	Afghanistan	2020-03-01	1.0	0.0	0.143	NaN	0.0	

5 rows × 67 columns

In [14]:
```
covid_data_2 = covid_data_1[~covid_data_1["total_deaths"].isnull()]
covid_data_2.head(n=30)
```

Out[14]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smo
81	AFG	Asia	Afghanistan	2020-03-24	40.0	16.0	3.429	1.0	1.0	
82	AFG	Asia	Afghanistan	2020-03-25	42.0	2.0	2.857	1.0	0.0	
83	AFG	Asia	Afghanistan	2020-03-26	74.0	32.0	7.429	1.0	0.0	
84	AFG	Asia	Afghanistan	2020-03-27	74.0	0.0	7.429	1.0	0.0	
85	AFG	Asia	Afghanistan	2020-03-28	80.0	6.0	8.286	2.0	1.0	
86	AFG	Asia	Afghanistan	2020-03-29	91.0	11.0	9.571	2.0	0.0	
87	AFG	Asia	Afghanistan	2020-03-30	106.0	15.0	11.714	3.0	1.0	
88	AFG	Asia	Afghanistan	2020-03-31	114.0	8.0	10.571	4.0	1.0	
89	AFG	Asia	Afghanistan	2020-04-01	166.0	52.0	17.714	4.0	0.0	
90	AFG	Asia	Afghanistan	2020-04-02	192.0	26.0	16.857	4.0	0.0	
91	AFG	Asia	Afghanistan	2020-04-03	194.0	2.0	17.143	4.0	0.0	
92	AFG	Asia	Afghanistan	2020-04-04	254.0	60.0	24.857	5.0	1.0	
93	AFG	Asia	Afghanistan	2020-04-05	274.0	20.0	26.143	5.0	0.0	
94	AFG	Asia	Afghanistan	2020-04-06	299.0	25.0	27.571	7.0	2.0	
95	AFG	Asia	Afghanistan	2020-04-07	337.0	38.0	31.857	7.0	0.0	
96	AFG	Asia	Afghanistan	2020-04-08	367.0	30.0	28.714	11.0	4.0	
97	AFG	Asia	Afghanistan	2020-04-09	423.0	56.0	33.000	14.0	3.0	
98	AFG	Asia	Afghanistan	2020-04-10	444.0	21.0	35.714	15.0	1.0	
99	AFG	Asia	Afghanistan	2020-04-11	484.0	40.0	32.857	15.0	0.0	
100	AFG	Asia	Afghanistan	2020-04-12	521.0	37.0	35.286	15.0	0.0	
101	AFG	Asia	Afghanistan	2020-04-13	555.0	34.0	36.571	18.0	3.0	
102	AFG	Asia	Afghanistan	2020-04-14	607.0	52.0	38.571	19.0	1.0	
103	AFG	Asia	Afghanistan	2020-04-15	665.0	58.0	42.571	22.0	3.0	
104	AFG	Asia	Afghanistan	2020-04-16	721.0	56.0	42.571	25.0	3.0	

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smo
105	AFG	Asia	Afghanistan	2020-04-17	794.0	73.0	50.000	29.0	4.0	
106	AFG	Asia	Afghanistan	2020-04-18	845.0	51.0	51.571	30.0	1.0	
107	AFG	Asia	Afghanistan	2020-04-19	908.0	63.0	55.286	30.0	0.0	
108	AFG	Asia	Afghanistan	2020-04-20	933.0	25.0	54.000	30.0	0.0	
109	AFG	Asia	Afghanistan	2020-04-21	996.0	63.0	55.571	33.0	3.0	
110	AFG	Asia	Afghanistan	2020-04-22	1026.0	30.0	51.571	36.0	3.0	

30 rows × 67 columns

In [15]:
```
covid_data_3 = covid_data_1[~covid_data_1["total_deaths"].isnull()]
covid_data_3.head(n=30)
```

Out[15]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smo
81	AFG	Asia	Afghanistan	2020-03-24	40.0	16.0	3.429	1.0	1.0	
82	AFG	Asia	Afghanistan	2020-03-25	42.0	2.0	2.857	1.0	0.0	
83	AFG	Asia	Afghanistan	2020-03-26	74.0	32.0	7.429	1.0	0.0	
84	AFG	Asia	Afghanistan	2020-03-27	74.0	0.0	7.429	1.0	0.0	
85	AFG	Asia	Afghanistan	2020-03-28	80.0	6.0	8.286	2.0	1.0	
86	AFG	Asia	Afghanistan	2020-03-29	91.0	11.0	9.571	2.0	0.0	
87	AFG	Asia	Afghanistan	2020-03-30	106.0	15.0	11.714	3.0	1.0	
88	AFG	Asia	Afghanistan	2020-03-31	114.0	8.0	10.571	4.0	1.0	
89	AFG	Asia	Afghanistan	2020-04-01	166.0	52.0	17.714	4.0	0.0	
90	AFG	Asia	Afghanistan	2020-04-02	192.0	26.0	16.857	4.0	0.0	
91	AFG	Asia	Afghanistan	2020-04-03	194.0	2.0	17.143	4.0	0.0	
92	AFG	Asia	Afghanistan	2020-04-04	254.0	60.0	24.857	5.0	1.0	
93	AFG	Asia	Afghanistan	2020-04-05	274.0	20.0	26.143	5.0	0.0	
94	AFG	Asia	Afghanistan	2020-04-06	299.0	25.0	27.571	7.0	2.0	
95	AFG	Asia	Afghanistan	2020-04-07	337.0	38.0	31.857	7.0	0.0	
96	AFG	Asia	Afghanistan	2020-04-08	367.0	30.0	28.714	11.0	4.0	
97	AFG	Asia	Afghanistan	2020-04-09	423.0	56.0	33.000	14.0	3.0	
98	AFG	Asia	Afghanistan	2020-04-10	444.0	21.0	35.714	15.0	1.0	
99	AFG	Asia	Afghanistan	2020-04-11	484.0	40.0	32.857	15.0	0.0	
100	AFG	Asia	Afghanistan	2020-04-12	521.0	37.0	35.286	15.0	0.0	
101	AFG	Asia	Afghanistan	2020-04-13	555.0	34.0	36.571	18.0	3.0	
102	AFG	Asia	Afghanistan	2020-04-14	607.0	52.0	38.571	19.0	1.0	
103	AFG	Asia	Afghanistan	2020-04-15	665.0	58.0	42.571	22.0	3.0	
104	AFG	Asia	Afghanistan	2020-04-16	721.0	56.0	42.571	25.0	3.0	

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smo
105	AFG	Asia	Afghanistan	2020-04-17	794.0	73.0	50.000	29.0	4.0	
106	AFG	Asia	Afghanistan	2020-04-18	845.0	51.0	51.571	30.0	1.0	
107	AFG	Asia	Afghanistan	2020-04-19	908.0	63.0	55.286	30.0	0.0	
108	AFG	Asia	Afghanistan	2020-04-20	933.0	25.0	54.000	30.0	0.0	
109	AFG	Asia	Afghanistan	2020-04-21	996.0	63.0	55.571	33.0	3.0	
110	AFG	Asia	Afghanistan	2020-04-22	1026.0	30.0	51.571	36.0	3.0	

30 rows × 67 columns

In [16]:
```
description= covid_data_3['continent'].value_counts()
print(description)
```

```
Africa           64850
Europe           59722
Asia             51202
North America    44825
South America    15568
Oceania          14239
Name: continent, dtype: int64
```

In [17]:
```
len(np.unique(covid_data_3.extreme_poverty))
```

Out[17]:
```
77
```

In [18]:
```
corr_matrix =covid_data_3.corr().abs()
corr_matrix
```

Out[18]:

	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothed	tota
total_cases	1.000000	0.400467	0.476020	0.935027	0.324419	0.360864	
new_cases	0.400467	1.000000	0.838377	0.449408	0.498040	0.425669	
new_cases_smoothed	0.476020	0.838377	1.000000	0.527789	0.473284	0.508987	
total_deaths	0.935027	0.449408	0.527789	1.000000	0.490452	0.543867	
new_deaths	0.324419	0.498040	0.473284	0.490452	1.000000	0.908371	
...	...	...	...	...	...	...	
population	0.641180	0.384070	0.449113	0.732750	0.598404	0.658236	
excess_mortality_cumulative_absolute	0.765445	0.131955	0.378090	0.935685	0.259736	0.461382	
excess_mortality_cumulative	0.055929	0.031952	0.025078	0.240368	0.062511	0.186284	
excess_mortality	0.043893	0.000567	0.067055	0.000896	0.178133	0.272576	
excess_mortality_cumulative_per_million	0.168656	0.008140	0.031790	0.268256	0.004047	0.046283	

62 rows × 62 columns

In [19]:
```
covid_data_3.describe(datetime_is_numeric = True, include = 'all')
```

Out[19]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths
count	265386	250406	265386	265386	2.653860e+05	2.653860e+05	2.653860e+05	2.653860e+05	265356.000000
unique	238	6	238	NaN	NaN	NaN	NaN	NaN	NaN
top	OWID_HIC	Africa	High income	NaN	NaN	NaN	NaN	NaN	NaN
freq	1275	64850	1275	NaN	NaN	NaN	NaN	NaN	NaN
mean	NaN	NaN	NaN	2021-12-05 10:42:30.301824256	6.564924e+06	1.225864e+04	1.225697e+04	8.262097e+04	109.488536
min	NaN	NaN	NaN	2020-01-08 00:00:00	1.000000e+00	0.000000e+00	0.000000e+00	1.000000e+00	0.000000
25%	NaN	NaN	NaN	2021-02-17 00:00:00	1.181900e+04	0.000000e+00	5.714000e+00	1.240000e+02	0.000000
50%	NaN	NaN	NaN	2021-12-17 00:00:00	8.789100e+04	2.200000e+01	7.671400e+01	1.221000e+03	0.000000
75%	NaN	NaN	NaN	2022-09-28 00:00:00	8.259762e+05	6.070000e+02	9.229642e+02	1.101675e+04	7.000000
max	NaN	NaN	NaN	2023-07-05 00:00:00	7.677261e+08	8.401732e+06	6.402720e+06	6.948751e+06	27941.000000
std	NaN	NaN	NaN	NaN	3.909047e+07	1.247394e+05	1.066603e+05	4.251826e+05	693.213056

11 rows × 67 columns

In [20]:
```
covid_data_3.drop(['iso_code'],axis=1).mode()
```

Out[20]:

	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothed	total_
0	Africa	Europe	2023-06-17	17786.0	0.0	0.0	1.0	0.0	0.0	
1	NaN	European Union	2023-06-18	NaN	NaN	NaN	NaN	NaN	NaN	
2	NaN	Germany	2023-06-19	NaN	NaN	NaN	NaN	NaN	NaN	
3	NaN	High income	2023-06-20	NaN	NaN	NaN	NaN	NaN	NaN	
4	NaN	World	2023-06-21	NaN	NaN	NaN	NaN	NaN	NaN	
5	NaN	NaN	2023-06-22	NaN	NaN	NaN	NaN	NaN	NaN	
6	NaN	NaN	2023-06-23	NaN	NaN	NaN	NaN	NaN	NaN	
7	NaN	NaN	2023-06-24	NaN	NaN	NaN	NaN	NaN	NaN	
8	NaN	NaN	2023-06-25	NaN	NaN	NaN	NaN	NaN	NaN	
9	NaN	NaN	2023-06-26	NaN	NaN	NaN	NaN	NaN	NaN	
10	NaN	NaN	2023-06-27	NaN	NaN	NaN	NaN	NaN	NaN	
11	NaN	NaN	2023-06-28	NaN	NaN	NaN	NaN	NaN	NaN	
12	NaN	NaN	2023-06-29	NaN	NaN	NaN	NaN	NaN	NaN	

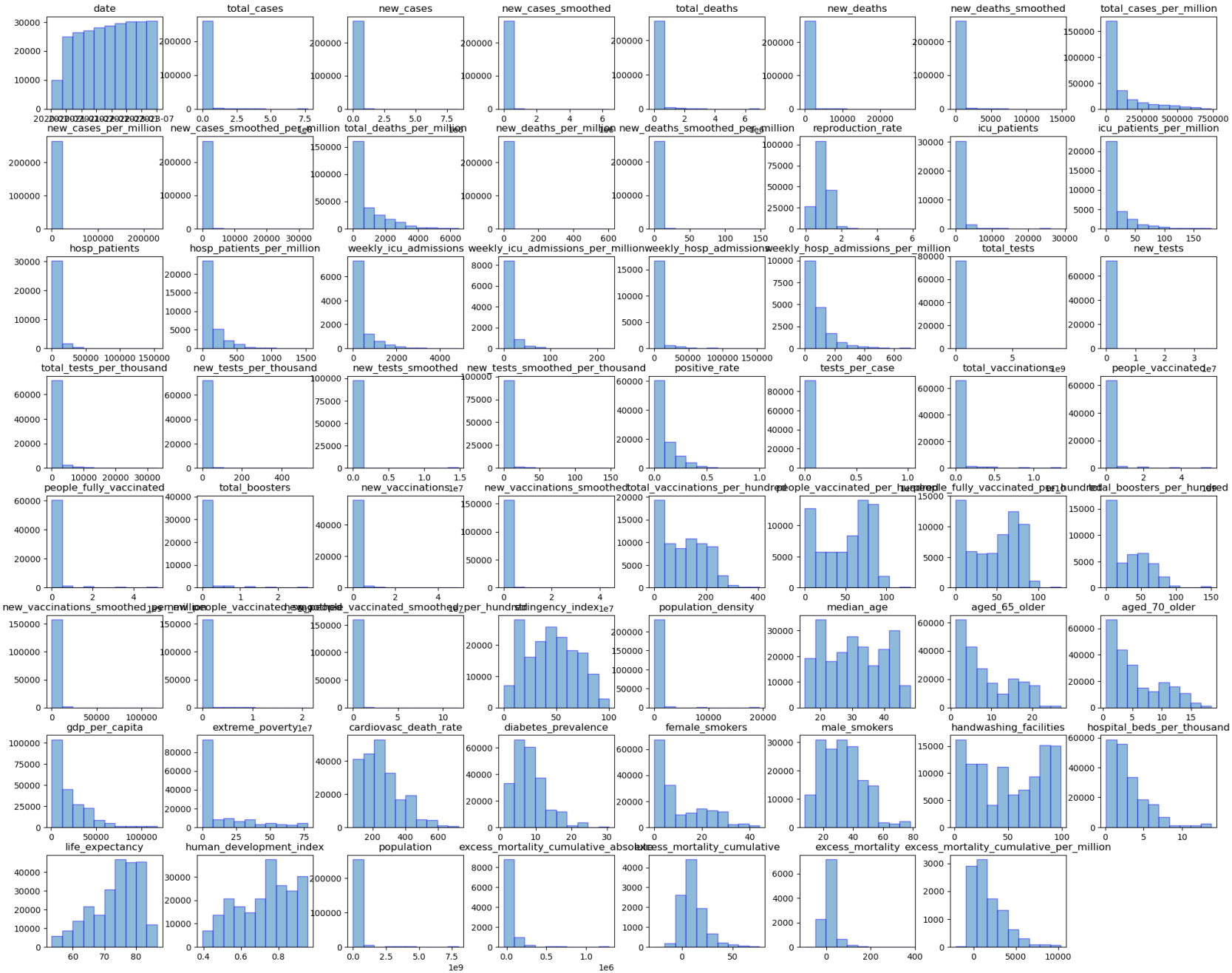
13 rows × 66 columns

In [21]:
```
covid_data_3.skew()
```

Out[21]:
```
total_cases                                11.669432
new_cases                                  32.484583
new_cases_smoothed                         29.150026
total_deaths                               10.011780
new_deaths                                 13.687627
                                             ...
population                                  8.006901
excess_mortality_cumulative_absolute        5.435966
excess_mortality_cumulative                 1.518195
excess_mortality                            3.159321
excess_mortality_cumulative_per_million     1.419986
Length: 62, dtype: float64
```

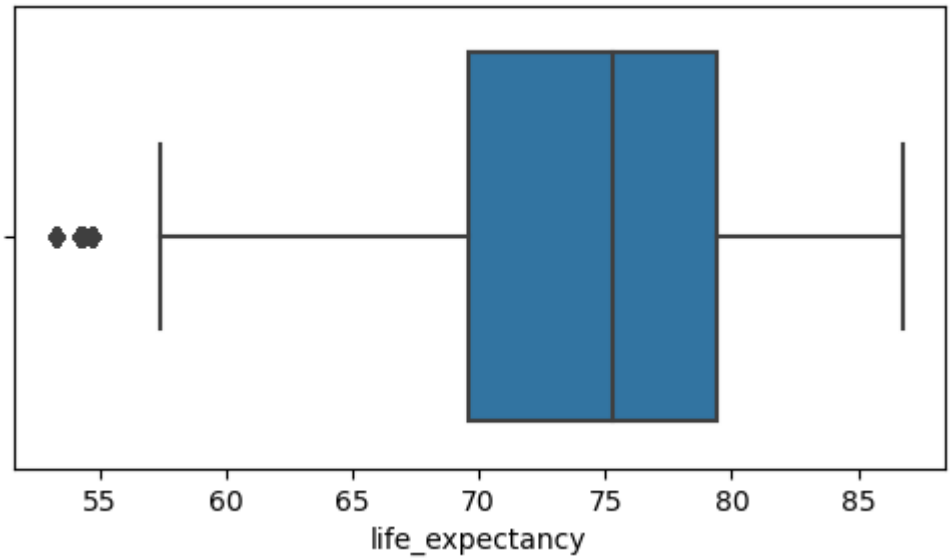
In [ ]:

In [22]:
```
covid_data_3.hist(figsize=(25,20), edgecolor = 'blue', alpha = 0.5, grid = False );
```

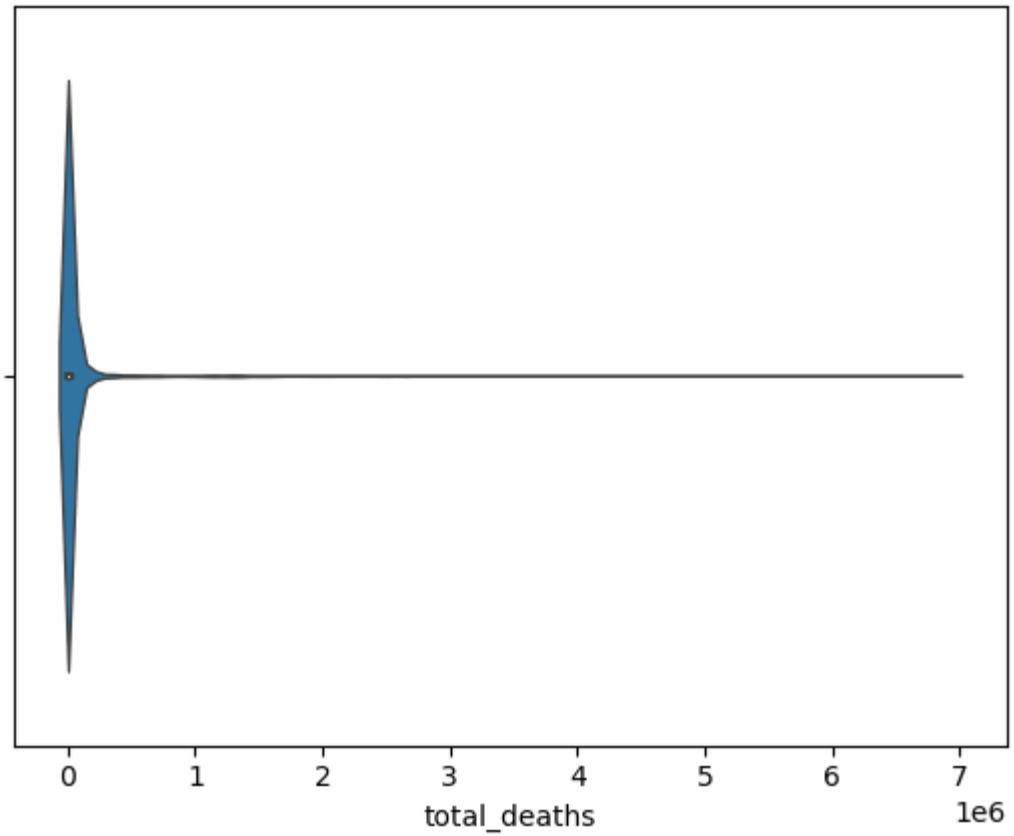


In [23]:
```
plt.figure(figsize = (6,3))
sns.boxplot(x = (covid_data_3['life_expectancy']))
```

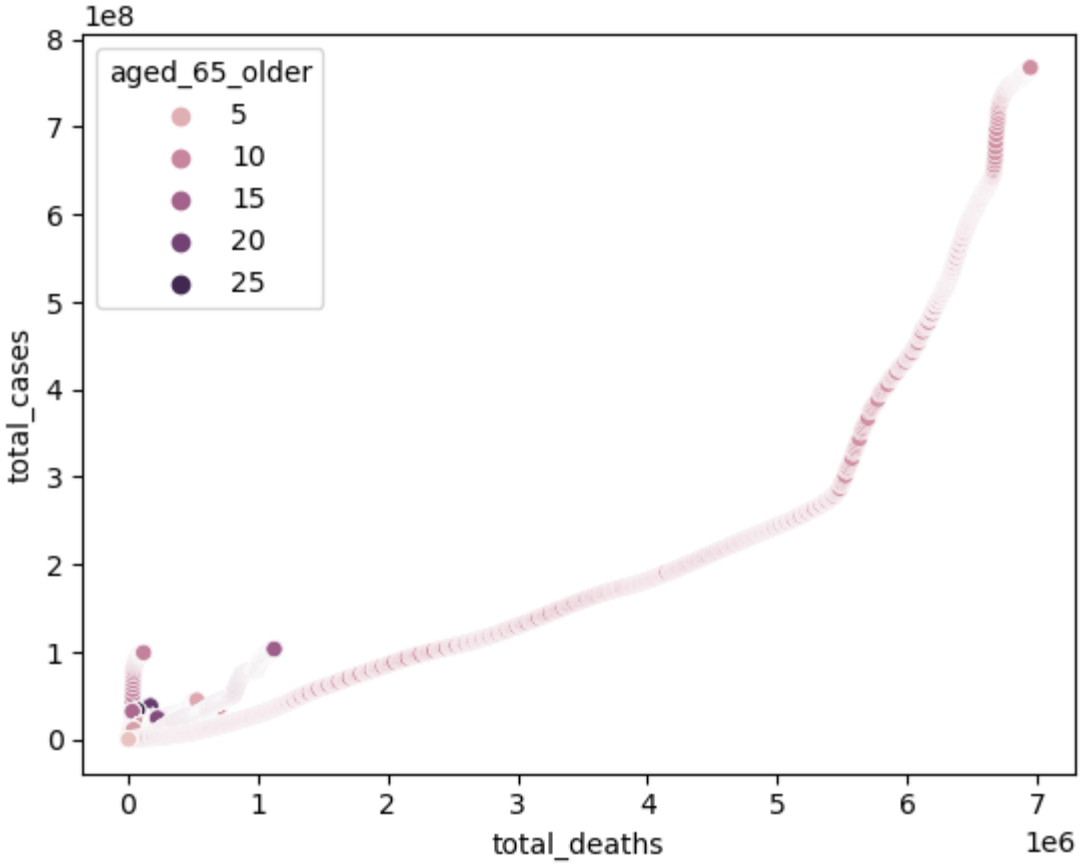
Out[23]: <AxesSubplot:xlabel='life_expectancy'>



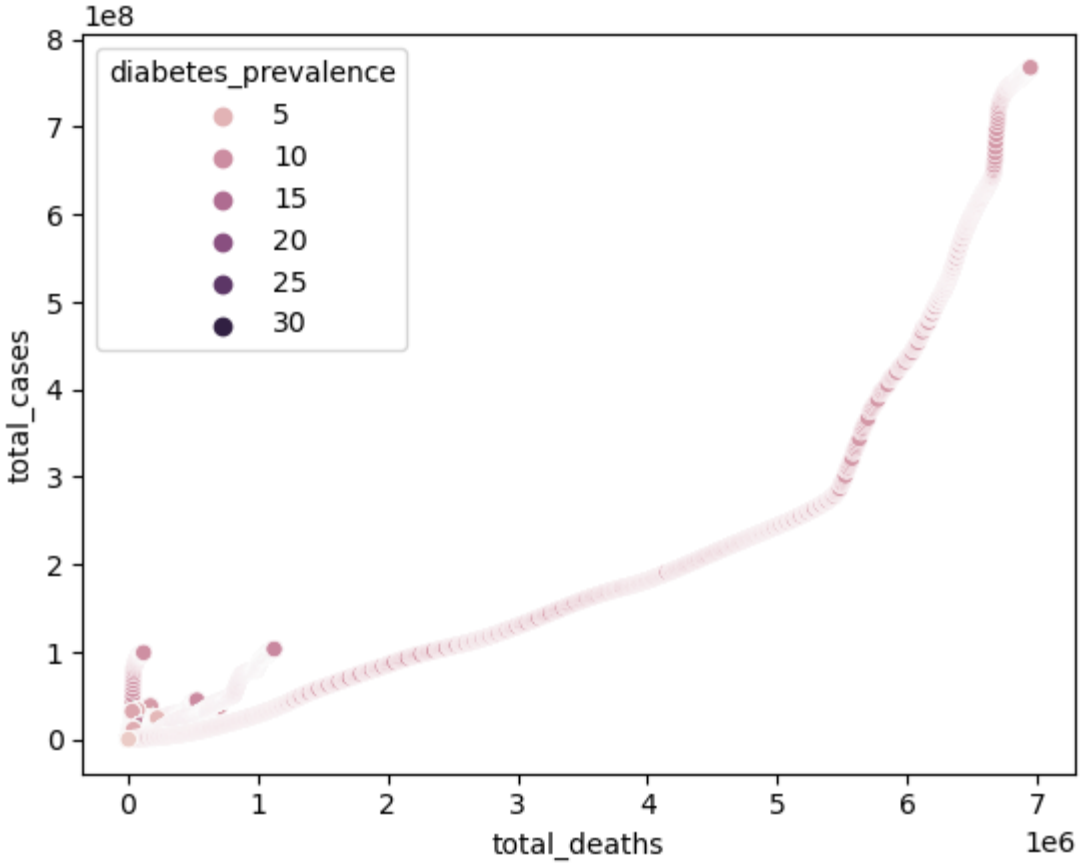
In [24]:
```
sns.violinplot(x =covid_data_3['total_deaths'],linewidth =0.9);
```



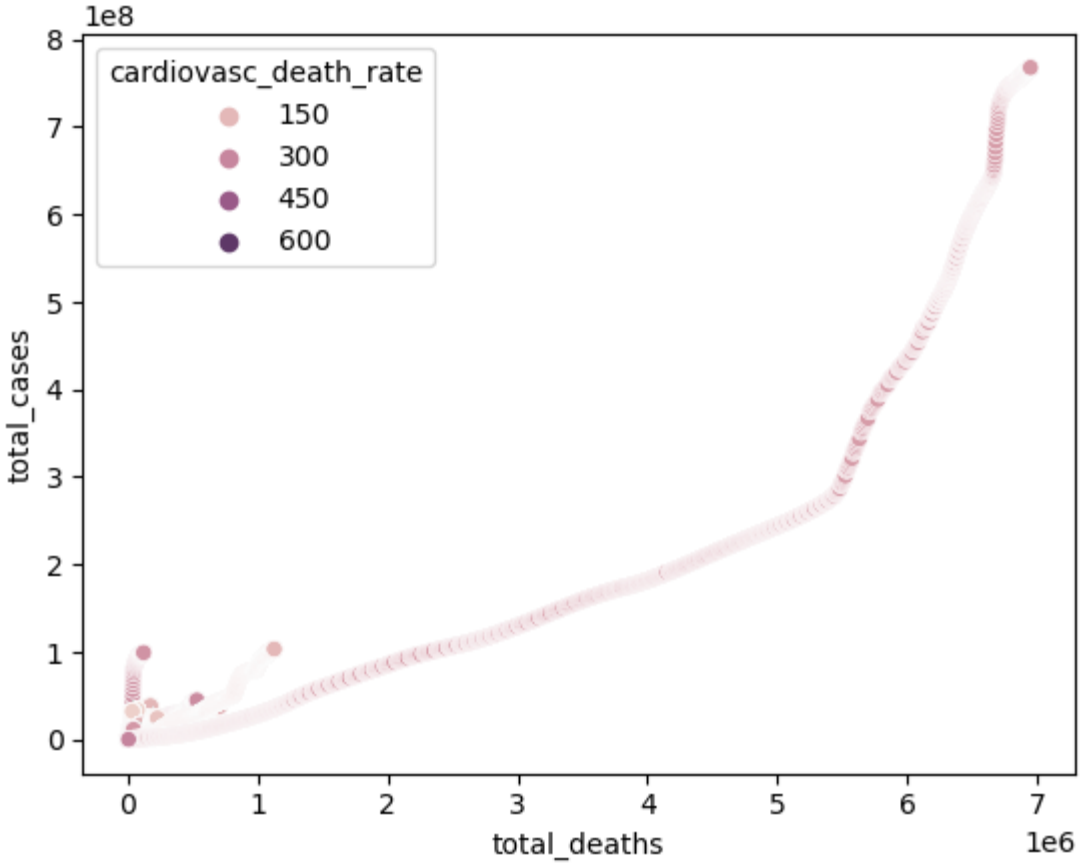
In [25]:
```
sns.scatterplot(data= covid_data_3 ,x= 'total_deaths',y='total_cases', hue = 'aged_65_older' );
```



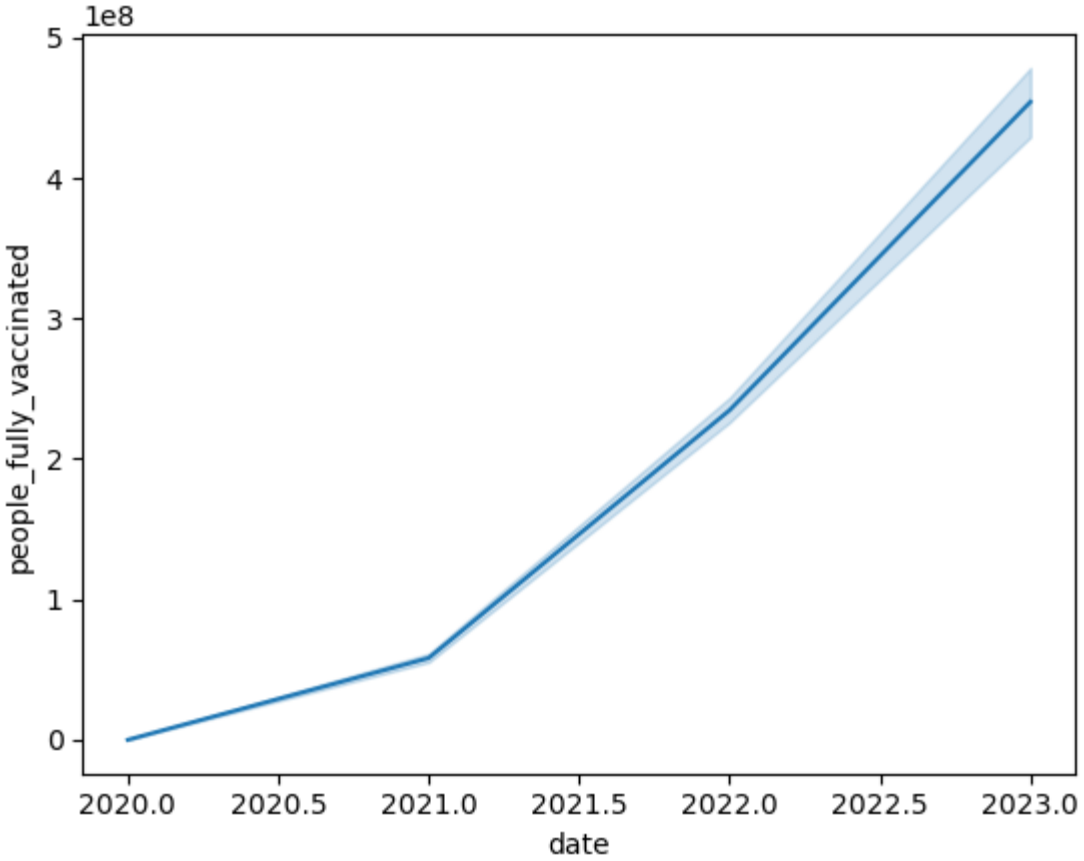
In [26]:
```
sns.scatterplot(data= covid_data_3 ,x= 'total_deaths',y='total_cases', hue = 'diabetes_prevalence' );
```



In [27]:
```
sns.scatterplot(data= covid_data_3 ,x= 'total_deaths',y='total_cases', hue = 'cardiovasc_death_rate' );
```

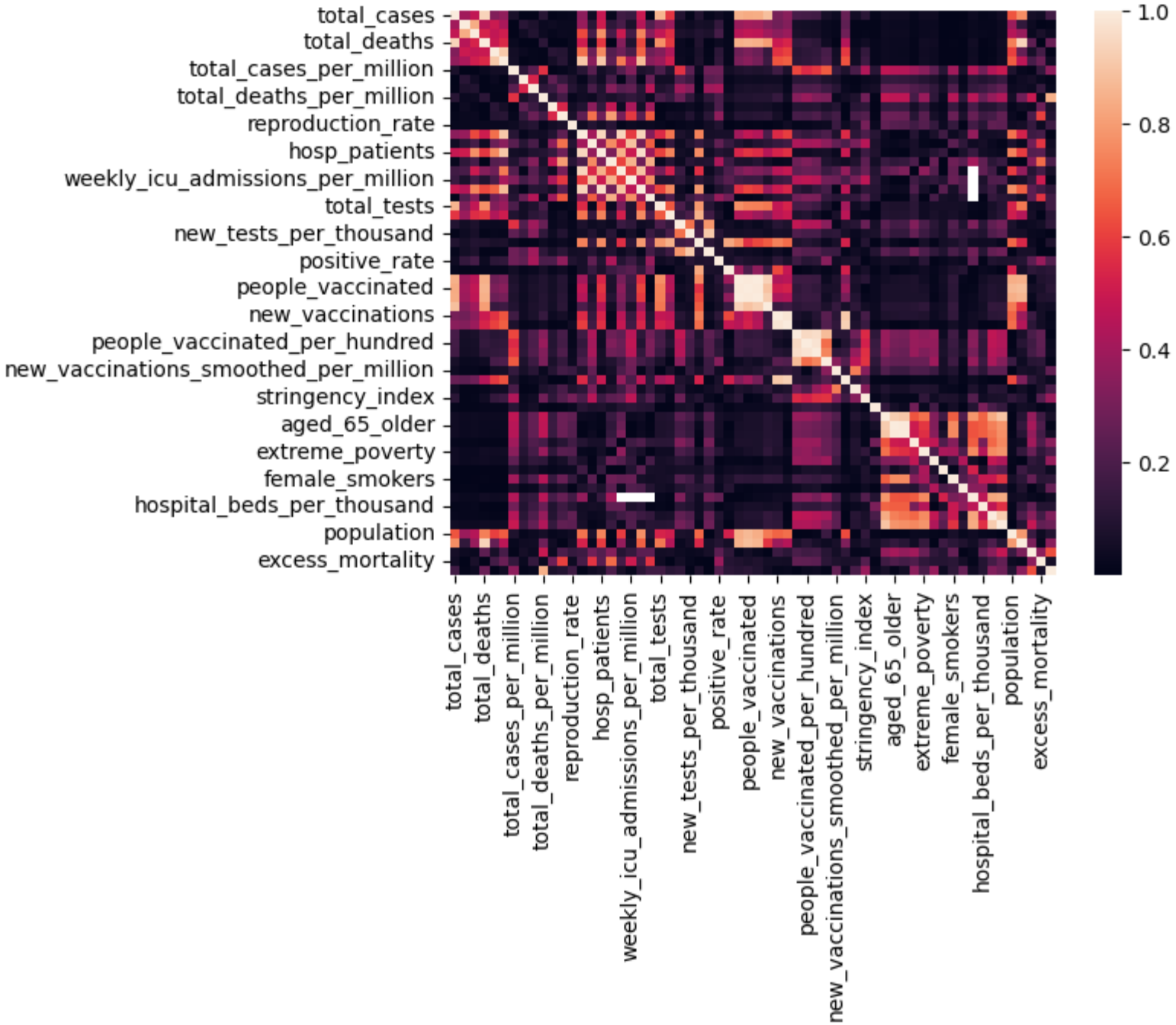


In [28]:
```
year = covid_data_3['date'].dt.year
sns.lineplot(data = covid_data_3, x= year ,y='people_fully_vaccinated', estimator = np.mean);
plt.figure(figsize = (7,3));
```




```
<Figure size 700x300 with 0 Axes>
```

In [29]:
```
sns.heatmap(corr_matrix);
```


total_cases
total_deaths
total_cases_per_million
total_deaths_per_million
reproduction_rate
hosp_patients
weekly_icu_admissions_per_million
total_tests
new_tests_per_thousand
positive_rate
people_vaccinated
new_vaccinations
people_vaccinated_per_hundred
new_vaccinations_smoothed_per_million
stringency_index
aged_65_older
extreme_poverty
female_smokers
hospital_beds_per_thousand
population
excess_mortality
1.0
0.8
0.6
0.4
0.2

In [30]:
```
covid_data_3.isnull().sum()
```

Out[30]:
```
iso_code                                    0
continent                               14980
location                                    0
date                                        0
total_cases                                 0
                                          ...
population                                  0
excess_mortality_cumulative_absolute       255091
excess_mortality_cumulative                255091
excess_mortality                           255091
excess_mortality_cumulative_per_million    255091
Length: 67, dtype: int64
```

In [31]:
```
percent_missing = covid_data_3.isnull().mean()

percent_missing
```

Out[31]:
```
iso_code                                   0.000000
continent                                  0.056446
location                                   0.000000
date                                       0.000000
total_cases                                0.000000
                                             ...
population                                 0.000000
excess_mortality_cumulative_absolute       0.961207
excess_mortality_cumulative                0.961207
excess_mortality                           0.961207
excess_mortality_cumulative_per_million    0.961207
Length: 67, dtype: float64
```

In [32]:
```
def missing_value_pct_df(data):
    """Create a DataFrame to summarize missing values."""

    percent_missing = data.isnull().mean()
    missing_value_df = \
    pd.DataFrame(percent_missing).reset_index()

    missing_value_df = \
    missing_value_df.rename(columns = { 'index':'column_name', 0 : 'percent_missing'})

    missing_value_df['percent_missing']=  missing_value_df['percent_missing'].apply(lambda x:round(x*100,2))

    missing_value_df =  missing_value_df.sort_values(by = ['percent_missing'], ascending = False)

    return missing_value_df
```

In [33]:
```
missing_value_df =  missing_value_pct_df(covid_data_3)
missing_value_df
```

Out[33]:

	column_name	percent_missing
21	weekly_icu_admissions	96.36
22	weekly_icu_admissions_per_million	96.36
63	excess_mortality_cumulative_absolute	96.12
66	excess_mortality_cumulative_per_million	96.12
64	excess_mortality_cumulative	96.12
...	...	...
11	new_cases_per_million	0.00
10	total_cases_per_million	0.00
7	total_deaths	0.00
6	new_cases_smoothed	0.00
0	iso_code	0.00

67 rows × 2 columns

In [34]:
```
threshold = 85

cols_to_drop = list(missing_value_df[missing_value_df['percent_missing']>threshold]['column_name'])

print("Number of features to drop :", len(missing_value_df[missing_value_df['percent_missing']>threshold]))

print( f'features with missing values greater than {threshold }%:',cols_to_drop)
```

```
Number of features to drop : 12
features with missing values greater than 85%: ['weekly_icu_admissions', 'weekly_icu_admissions_per_million', 'exces
s_mortality_cumulative_absolute', 'excess_mortality_cumulative_per_million', 'excess_mortality_cumulative', 'excess_
mortality', 'weekly_hosp_admissions_per_million', 'weekly_hosp_admissions', 'icu_patients_per_million', 'icu_patient
s', 'hosp_patients_per_million', 'hosp_patients']
```

In [35]:
```
covid_data_cleaned = covid_data_3.drop(cols_to_drop,axis=1 )
covid_data_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 265386 entries, 81 to 323660
Data columns (total 55 columns):
 #   Column                                    Non-Null Count   Dtype
---  ------                                    --------------   -----
 0   iso_code                                  265386 non-null  object
 1   continent                                 250406 non-null  object
 2   location                                  265386 non-null  object
 3   date                                      265386 non-null  datetime64[ns]
 4   total_cases                               265386 non-null  float64
 5   new_cases                                 265386 non-null  float64
 6   new_cases_smoothed                        265386 non-null  float64
 7   total_deaths                              265386 non-null  float64
 8   new_deaths                                265356 non-null  float64
 9   new_deaths_smoothed                       265356 non-null  float64
 10  total_cases_per_million                   265386 non-null  float64
 11  new_cases_per_million                     265386 non-null  float64
 12  new_cases_smoothed_per_million            265386 non-null  float64
 13  total_deaths_per_million                  265386 non-null  float64
 14  new_deaths_per_million                    265356 non-null  float64
 15  new_deaths_smoothed_per_million           265356 non-null  float64
 16  reproduction_rate                         180005 non-null  float64
 17  total_tests                               76201 non-null   float64
 18  new_tests                                 72474 non-null   float64
 19  total_tests_per_thousand                  76201 non-null   float64
 20  new_tests_per_thousand                    72474 non-null   float64
 21  new_tests_smoothed                        98842 non-null   float64
 22  new_tests_smoothed_per_thousand           98842 non-null   float64
 23  positive_rate                             92951 non-null   float64
 24  tests_per_case                            91834 non-null   float64
 25  tests_units                               101159 non-null  object
 26  total_vaccinations                        70901 non-null   float64
 27  people_vaccinated                         67980 non-null   float64
 28  people_fully_vaccinated                   64548 non-null   float64
 29  total_boosters                            41798 non-null   float64
 30  new_vaccinations                          58047 non-null   float64
 31  new_vaccinations_smoothed                 159475 non-null  float64
 32  total_vaccinations_per_hundred            70901 non-null   float64
 33  people_vaccinated_per_hundred             67980 non-null   float64
 34  people_fully_vaccinated_per_hundred       64548 non-null   float64
 35  total_boosters_per_hundred                41798 non-null   float64
 36  new_vaccinations_smoothed_per_million     159475 non-null  float64
 37  new_people_vaccinated_smoothed            159473 non-null  float64
 38  new_people_vaccinated_smoothed_per_hundred  159473 non-null  float64
```

```
 39  stringency_index                         170567 non-null  float64
 40  population_density                       235647 non-null  float64
 41  median_age                               222817 non-null  float64
 42  aged_65_older                            215732 non-null  float64
 43  aged_70_older                            220425 non-null  float64
 44  gdp_per_capita                           216739 non-null  float64
 45  extreme_poverty                          144184 non-null  float64
 46  cardiovasc_death_rate                    218476 non-null  float64
 47  diabetes_prevalence                      227305 non-null  float64
 48  female_smokers                           167749 non-null  float64
 49  male_smokers                             165338 non-null  float64
 50  handwashing_facilities                   107428 non-null  float64
 51  hospital_beds_per_thousand               194750 non-null  float64
 52  life_expectancy                          247246 non-null  float64
 53  human_development_index                  212752 non-null  float64
 54  population                               265386 non-null  float64
dtypes: datetime64[ns](1), float64(50), object(4)
memory usage: 121.4+ MB
```

In [36]:
```
missing_value_df =  missing_value_pct_df(covid_data_cleaned)

missing_columns = list(missing_value_df[missing_value_df['percent_missing']>0]['column_name'])

print ("Number of features missing from the DataFrame:", len(missing_columns))
```

```
Number of features missing from the DataFrame: 43
```

In [37]:
```
dtypes = ("int","float")

numerical_columns = list(covid_data_cleaned.select_dtypes(dtypes).columns)

print("Numerical features with missing values", list(set(numerical_columns).intersection(missing_columns)))
```

```
Numerical features with missing values ['new_deaths_smoothed', 'people_fully_vaccinated_per_hundred', 'total_vaccina
tions_per_hundred', 'diabetes_prevalence', 'male_smokers', 'aged_65_older', 'new_vaccinations_smoothed_per_million',
'life_expectancy', 'gdp_per_capita', 'positive_rate', 'hospital_beds_per_thousand', 'new_tests_per_thousand', 'total
_boosters_per_hundred', 'new_tests_smoothed_per_thousand', 'new_deaths_smoothed_per_million', 'cardiovasc_death_rate
', 'handwashing_facilities', 'human_development_index', 'tests_per_case', 'total_tests_per_thousand', 'median_age',
'new_tests', 'total_vaccinations', 'female_smokers', 'population_density', 'extreme_poverty', 'people_fully_vaccinat
ed', 'total_tests', 'new_deaths_per_million', 'new_people_vaccinated_smoothed_per_hundred', 'total_boosters', 'new_t
ests_smoothed', 'aged_70_older', 'stringency_index', 'people_vaccinated_per_hundred', 'new_deaths', 'new_vaccination
s_smoothed', 'people_vaccinated', 'new_people_vaccinated_smoothed', 'reproduction_rate', 'new_vaccinations']
```

In [ ]:

In [38]:
```
print("mean of life expectancy:", round(covid_data_cleaned['life_expectancy'].mean(),2))
covid_data_cleaned["life_expectancy"].fillna(round(covid_data_cleaned['life_expectancy'].mean(),2),inplace=True)
```

```
mean of life expectancy: 73.7
```

In [39]:
```
print("mean of cardiovasc_death_rate:", round(covid_data_cleaned['cardiovasc_death_rate'].mean(),2))
covid_data_cleaned["cardiovasc_death_rate"].fillna(round(covid_data_cleaned['cardiovasc_death_rate'].mean(),2),inpla
```

```
mean of cardiovasc_death_rate: 256.98
```

In [40]:
```
print("mean of diabetes_prevalence:", round(covid_data_cleaned['diabetes_prevalence'].mean(),2))
covid_data_cleaned["diabetes_prevalence"].fillna(round(covid_data_cleaned['diabetes_prevalence'].mean(),2),inplace=Tr
```

```
mean of diabetes_prevalence: 8.2
```

In [41]:
```
covid_data_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 265386 entries, 81 to 323660
Data columns (total 55 columns):
 #   Column                                    Non-Null Count   Dtype
---  ------                                    --------------   -----
 0   iso_code                                  265386 non-null  object
 1   continent                                 250406 non-null  object
 2   location                                  265386 non-null  object
 3   date                                      265386 non-null  datetime64[ns]
 4   total_cases                               265386 non-null  float64
 5   new_cases                                 265386 non-null  float64
 6   new_cases_smoothed                        265386 non-null  float64
 7   total_deaths                              265386 non-null  float64
 8   new_deaths                                265356 non-null  float64
 9   new_deaths_smoothed                       265356 non-null  float64
 10  total_cases_per_million                   265386 non-null  float64
 11  new_cases_per_million                     265386 non-null  float64
 12  new_cases_smoothed_per_million            265386 non-null  float64
 13  total_deaths_per_million                  265386 non-null  float64
 14  new_deaths_per_million                    265356 non-null  float64
 15  new_deaths_smoothed_per_million           265356 non-null  float64
 16  reproduction_rate                         180005 non-null  float64
 17  total_tests                               76201 non-null   float64
 18  new_tests                                 72474 non-null   float64
 19  total_tests_per_thousand                  76201 non-null   float64
 20  new_tests_per_thousand                    72474 non-null   float64
 21  new_tests_smoothed                        98842 non-null   float64
 22  new_tests_smoothed_per_thousand           98842 non-null   float64
 23  positive_rate                             92951 non-null   float64
 24  tests_per_case                            91834 non-null   float64
 25  tests_units                               101159 non-null  object
 26  total_vaccinations                        70901 non-null   float64
 27  people_vaccinated                         67980 non-null   float64
 28  people_fully_vaccinated                   64548 non-null   float64
 29  total_boosters                            41798 non-null   float64
 30  new_vaccinations                          58047 non-null   float64
 31  new_vaccinations_smoothed                 159475 non-null  float64
 32  total_vaccinations_per_hundred            70901 non-null   float64
 33  people_vaccinated_per_hundred             67980 non-null   float64
 34  people_fully_vaccinated_per_hundred       64548 non-null   float64
 35  total_boosters_per_hundred                41798 non-null   float64
 36  new_vaccinations_smoothed_per_million     159475 non-null  float64
 37  new_people_vaccinated_smoothed            159473 non-null  float64
 38  new_people_vaccinated_smoothed_per_hundred  159473 non-null  float64
```

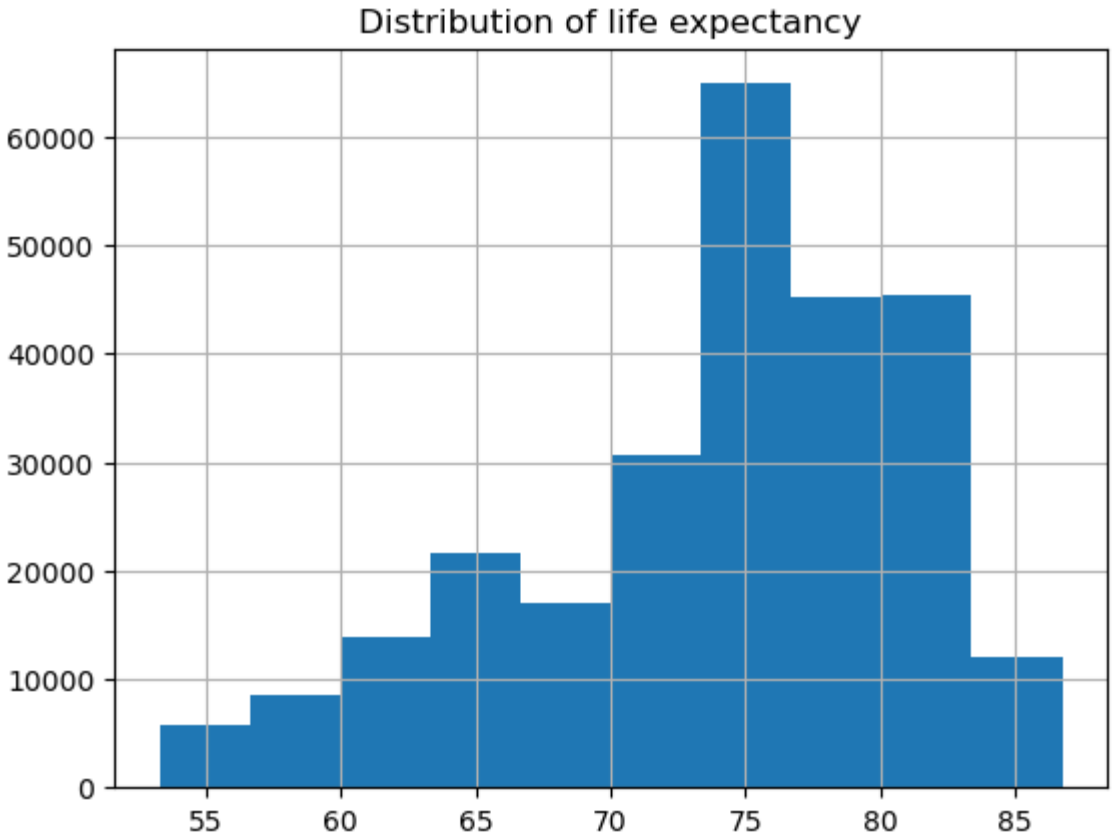
```
 39  stringency_index                    170567 non-null  float64
 40  population_density                  235647 non-null  float64
 41  median_age                          222817 non-null  float64
 42  aged_65_older                       215732 non-null  float64
 43  aged_70_older                       220425 non-null  float64
 44  gdp_per_capita                      216739 non-null  float64
 45  extreme_poverty                     144184 non-null  float64
 46  cardiovasc_death_rate               265386 non-null  float64
 47  diabetes_prevalence                 265386 non-null  float64
 48  female_smokers                      167749 non-null  float64
 49  male_smokers                        165338 non-null  float64
 50  handwashing_facilities              107428 non-null  float64
 51  hospital_beds_per_thousand          194750 non-null  float64
 52  life_expectancy                     265386 non-null  float64
 53  human_development_index             212752 non-null  float64
 54  population                          265386 non-null  float64
dtypes: datetime64[ns](1), float64(50), object(4)
memory usage: 121.4+ MB
```

In [42]:
```
categorical_columns = list(covid_data_cleaned.select_dtypes('object').columns)
print ("Categorical features with missing values:", list(set(categorical_columns).intersection(missing_columns)))
```

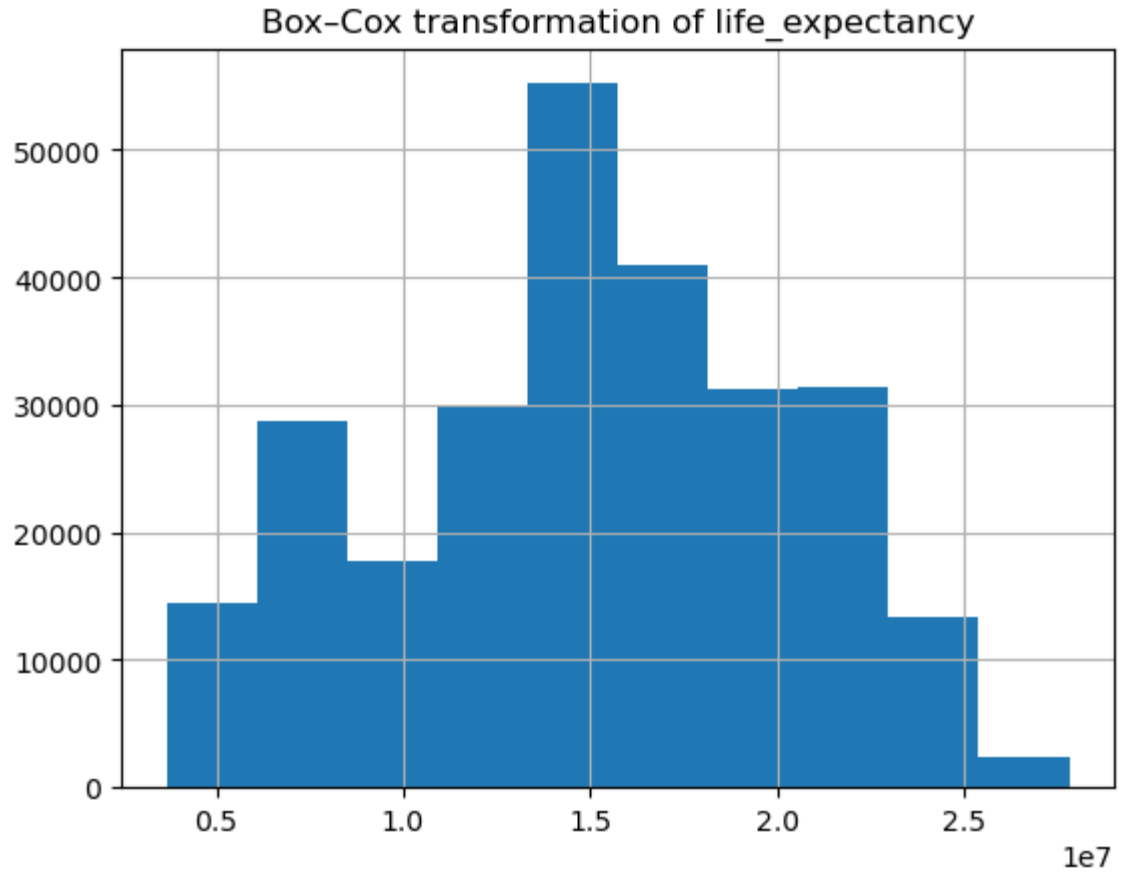
```
Categorical features with missing values: ['continent', 'tests_units']
```

In [43]:
```
covid_data_cleaned['tests_units'].fillna('Unknown',inplace =True)
covid_data_cleaned['continent'].fillna('Unknown',inplace =True)
```

In [44]:
```
covid_data_cleaned['life_expectancy'].hist()
plt.title("Distribution of life expectancy");
```



In [45]:
```
from scipy import stats
pd.Series(stats.boxcox(covid_data_cleaned['life_expectancy'])[0]).hist()
plt.title("Box-Cox transformation of life_expectancy");
```



In [46]:
```
(covid_data_cleaned).life_expectancy.describe()
```

Out[46]:
```
count    265386.000000
mean         73.701625
std           7.266999
min          53.280000
25%          69.910000
50%          74.620000
75%          78.930000
max          86.750000
Name: life_expectancy, dtype: float64
```

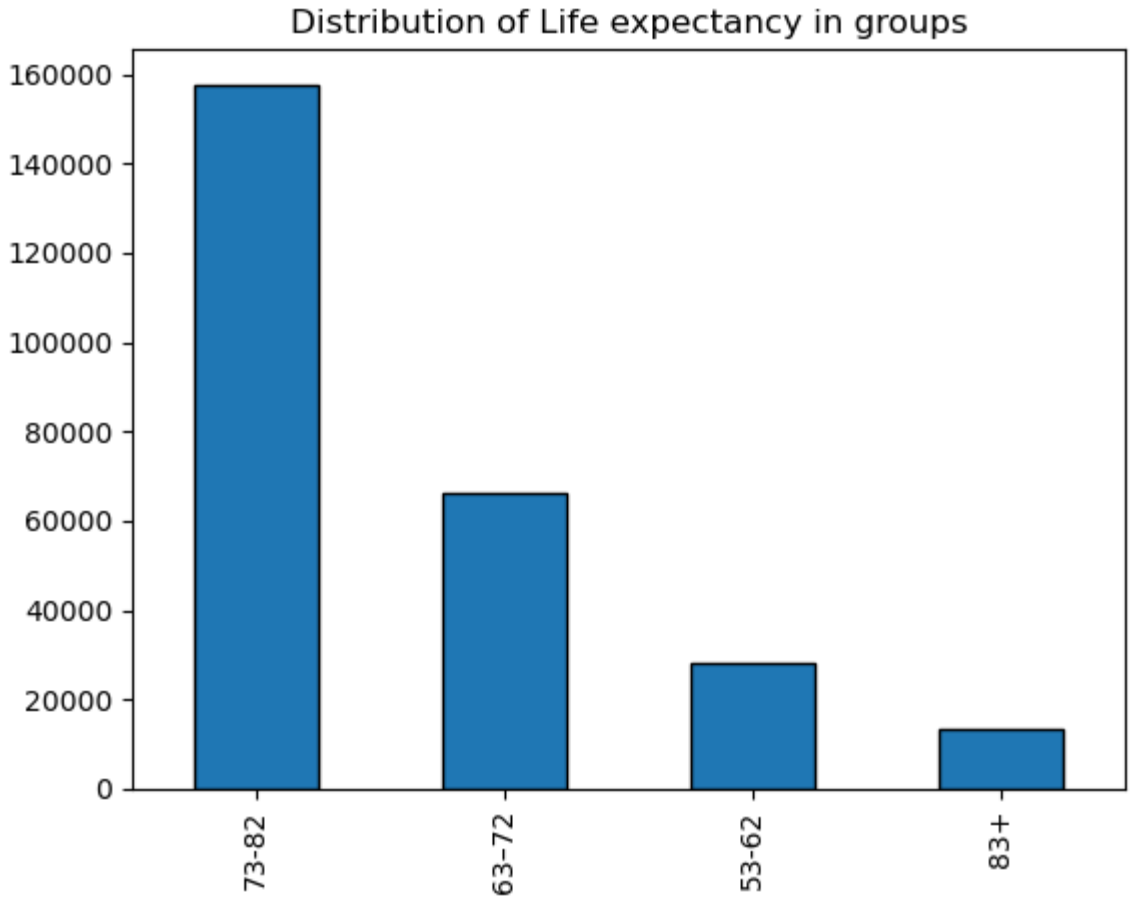
In [47]:
```
bins = [53, 63, 73, 83,93]
labels = ['53-62', '63-72', '73-82',
          '83+']
covid_data_cleaned["Age Group-life exp"]= pd.cut(covid_data_cleaned["life_expectancy"],bins=bins,labels=labels,right=

covid_data_cleaned
```

Out[47]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_s
81	AFG	Asia	Afghanistan	2020-03-24	40.0	16.0	3.429	1.0	1.0	
82	AFG	Asia	Afghanistan	2020-03-25	42.0	2.0	2.857	1.0	0.0	
83	AFG	Asia	Afghanistan	2020-03-26	74.0	32.0	7.429	1.0	0.0	
84	AFG	Asia	Afghanistan	2020-03-27	74.0	0.0	7.429	1.0	0.0	
85	AFG	Asia	Afghanistan	2020-03-28	80.0	6.0	8.286	2.0	1.0	
...	...	...	...	...	...	...	...	...	...	...
323656	ZWE	Africa	Zimbabwe	2023-07-01	265524.0	0.0	15.857	5707.0	0.0	
323657	ZWE	Africa	Zimbabwe	2023-07-02	265524.0	0.0	15.857	5707.0	0.0	
323658	ZWE	Africa	Zimbabwe	2023-07-03	265604.0	80.0	11.429	5709.0	2.0	
323659	ZWE	Africa	Zimbabwe	2023-07-04	265604.0	0.0	11.429	5709.0	0.0	
323660	ZWE	Africa	Zimbabwe	2023-07-05	265604.0	0.0	11.429	5709.0	0.0	

265386 rows × 56 columns

In [48]:
```
covid_age = covid_data_cleaned["Age Group-life exp"].value_counts()
covid_age.plot(kind='bar',edgecolor='black')
plt.title("Distribution of Life expectancy in groups");
```



In [49]:
```
list(covid_data_cleaned)
```

Out[49]:
```
['iso_code',
 'continent',
 'location',
 'date',
 'total_cases',
 'new_cases',
 'new_cases_smoothed',
 'total_deaths',
 'new_deaths',
 'new_deaths_smoothed',
 'total_cases_per_million',
 'new_cases_per_million',
 'new_cases_smoothed_per_million',
 'total_deaths_per_million',
 'new_deaths_per_million',
 'new_deaths_smoothed_per_million',
 'reproduction_rate',
 'total_tests',
 'new_tests',
 'total_tests_per_thousand',
 'new_tests_per_thousand',
 'new_tests_smoothed',
 'new_tests_smoothed_per_thousand',
 'positive_rate',
 'tests_per_case',
 'tests_units',
 'total_vaccinations',
 'people_vaccinated',
 'people_fully_vaccinated',
 'total_boosters',
 'new_vaccinations',
 'new_vaccinations_smoothed',
 'total_vaccinations_per_hundred',
 'people_vaccinated_per_hundred',
 'people_fully_vaccinated_per_hundred',
 'total_boosters_per_hundred',
 'new_vaccinations_smoothed_per_million',
 'new_people_vaccinated_smoothed',
 'new_people_vaccinated_smoothed_per_hundred',
 'stringency_index',
 'population_density',
 'median_age',
 'aged_65_older',
 'aged_70_older',
```

```
'gdp_per_capita',
'extreme_poverty',
'cardiovasc_death_rate',
'diabetes_prevalence',
'female_smokers',
'male_smokers',
'handwashing_facilities',
'hospital_beds_per_thousand',
'life_expectancy',
'human_development_index',
'population',
'Age Group-life exp']
```

In [50]:
```
covid_data_cleaned.dropna()
```

Out[50]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_s
23785	BGD	Asia	Bangladesh	2022-01-15	1612489.0	3447.0	2897.143	28136.0	7.0	
23786	BGD	Asia	Bangladesh	2022-01-16	1617711.0	5222.0	3430.143	28144.0	8.0	
23787	BGD	Asia	Bangladesh	2022-01-17	1624387.0	6676.0	4065.143	28154.0	10.0	
23788	BGD	Asia	Bangladesh	2022-01-18	1632794.0	8407.0	4915.000	28164.0	10.0	
23789	BGD	Asia	Bangladesh	2022-01-19	1642294.0	9500.0	5855.571	28176.0	12.0	
...	...	...	...	...	...	...	...	...	...	
323167	ZWE	Africa	Zimbabwe	2022-02-27	235803.0	336.0	368.429	5393.0	1.0	
323175	ZWE	Africa	Zimbabwe	2022-03-07	239209.0	190.0	462.000	5399.0	2.0	
323179	ZWE	Africa	Zimbabwe	2022-03-11	241548.0	499.0	491.714	5408.0	1.0	
323180	ZWE	Africa	Zimbabwe	2022-03-12	242069.0	521.0	475.714	5412.0	4.0	
323248	ZWE	Africa	Zimbabwe	2022-05-19	250007.0	259.0	175.571	5486.0	2.0	

1020 rows × 56 columns

In [51]:
```
categorical_columns = list(covid_data_cleaned.select_dtypes(['object']).columns)

print("Number of categorical features:", len(categorical_columns))
print("Names of categorical features:", (categorical_columns))
```

```
Number of categorical features: 4
Names of categorical features: ['iso_code', 'continent', 'location', 'tests_units']
```

In [52]:
```
cols= ['iso_code', 'continent', 'location', 'tests_units']
encoder= ce.OneHotEncoder(cols=cols,
                          return_df=True,
                          use_cat_names=True)
```

In [53]:
```
covid_data_encoded= encoder.fit_transform(covid_data_cleaned)


# Preview the data.
covid_data_encoded.head()
```

Out[53]:

	iso_code_AFG	iso_code_OWID_AFR	iso_code_ALB	iso_code_DZA	iso_code_ASM	iso_code_AND	iso_code_AGO	iso_code_AIA	iso_code_
81	1	0	0	0	0	0	0	0	
82	1	0	0	0	0	0	0	0	
83	1	0	0	0	0	0	0	0	
84	1	0	0	0	0	0	0	0	
85	1	0	0	0	0	0	0	0	

5 rows × 540 columns

In [54]:

```
import sys                                                  # Read system parameters.
import numpy as np                                          # Work with multi-dimensional arrays.
import pandas as pd                                         # Manipulate and analyze data.
import matplotlib                                           # Create and format charts.
import matplotlib.pyplot as plt
import category_encoders as ce                              # Encode data.
import sklearn                                              # Train and evaluate machine learning models.
from sklearn.model_selection import train_test_split, \
                                    learning_curve, \
                                    RandomizedSearchCV, \
                                    GridSearchCV
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, \
                            confusion_matrix, \
                            f1_score, \
                            recall_score, \
                            precision_score, \
                            plot_roc_curve, \
                            plot_precision_recall_curve, \
                            plot_confusion_matrix
from sklearn.dummy import DummyClassifier
import xgboost                                              # Build gradient boosting models.
from xgboost import XGBClassifier
import imblearn                                             # Deal with imbalanced data.
from imblearn.over_sampling import SMOTE                    # Perform oversampling.
from collections import Counter                             # Count objects in containers.
import pickle                                               # Save Python objects as binary files.
import warnings                                             # Suppress warnings.
warnings.filterwarnings('ignore')

# Ensure results are reproducible.
np.random.seed(1)

# Summarize software libraries used.
print('Libraries used in this project:')
print('- Python {}'.format(sys.version))
print('- NumPy {}'.format(np.__version__))
```

```
print('- pandas {}'.format(pd.__version__))
print('- Matplotlib {}'.format(matplotlib.__version__))
print('- Category Encoders {}'.format(ce.__version__))
print('- scikit-learn {}'.format(sklearn.__version__))
print('- XGBoost {}'.format(xgboost.__version__))
print('- imbalanced-learn {}\n'.format(imblearn.__version__))
```

```
Libraries used in this project:
- Python 3.9.13 (main, Aug 25 2022, 23:51:50) [MSC v.1916 64 bit (AMD64)]
- NumPy 1.21.5
- pandas 1.4.4
- Matplotlib 3.5.2
- Category Encoders 2.6.1
- scikit-learn 1.0.2
- XGBoost 1.7.6
- imbalanced-learn 0.10.1
```

In [55]:
```
covid_data_2 = \
covid_data_encoded.filter(regex = 'total_cases|new_cases|total_deaths|new_deaths')
covid_data_2.head(n = 3)
```

Out[55]:

	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothed	total_cases_per_million	new_cases_per_m
81	40.0	16.0	3.429	1.0	1.0	0.143	0.973	
82	42.0	2.0	2.857	1.0	0.0	0.143	1.021	
83	74.0	32.0	7.429	1.0	0.0	0.143	1.799	

In [56]:
```
scaler = StandardScaler()

scaler.fit(covid_data_2)
covid_data_scaled = scaler.transform(covid_data_2)
print("new standard deviation:",covid_data_2.std())
print("new mean:", round(covid_data_2.mean()))
```

```
new standard deviation: total_cases                       3.909047e+07
new_cases                             1.247394e+05
new_cases_smoothed                    1.066603e+05
total_deaths                          4.251826e+05
new_deaths                            6.932131e+02
new_deaths_smoothed                   6.301621e+02
total_cases_per_million               1.450273e+05
new_cases_per_million                 1.082273e+03
new_cases_smoothed_per_million        5.775614e+02
total_deaths_per_million              1.071608e+03
new_deaths_per_million                5.742879e+00
new_deaths_smoothed_per_million       3.085512e+00
dtype: float64
new mean: total_cases                          6564924.0
new_cases                                12259.0
new_cases_smoothed                       12257.0
total_deaths                             82621.0
new_deaths                                 109.0
new_deaths_smoothed                        109.0
total_cases_per_million                  97961.0
new_cases_per_million                      174.0
new_cases_smoothed_per_million             175.0
total_deaths_per_million                   832.0
new_deaths_per_million                       1.0
new_deaths_smoothed_per_million              1.0
dtype: float64
```

In [57]:
```
pip install yellowbrick
```

```
Requirement already satisfied: yellowbrick in c:\users\haya\anaconda3\lib\site-packages (1.5)
Requirement already satisfied: matplotlib!=3.0.0,>=2.0.2 in c:\users\haya\anaconda3\lib\site-packages (from yellowbr
ick) (3.5.2)
Requirement already satisfied: numpy>=1.16.0 in c:\users\haya\anaconda3\lib\site-packages (from yellowbrick) (1.21.
5)
Requirement already satisfied: cycler>=0.10.0 in c:\users\haya\anaconda3\lib\site-packages (from yellowbrick) (0.11.
0)
Requirement already satisfied: scikit-learn>=1.0.0 in c:\users\haya\anaconda3\lib\site-packages (from yellowbrick)
(1.0.2)
Requirement already satisfied: scipy>=1.0.0 in c:\users\haya\anaconda3\lib\site-packages (from yellowbrick) (1.9.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\haya\anaconda3\lib\site-packages (from matplotlib!=3.0.
0,>=2.0.2->yellowbrick) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\haya\anaconda3\lib\site-packages (from matplotlib!=3.0.
0,>=2.0.2->yellowbrick) (1.4.2)
Requirement already satisfied: pillow>=6.2.0 in c:\users\haya\anaconda3\lib\site-packages (from matplotlib!=3.0.0,>=
2.0.2->yellowbrick) (9.2.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\haya\anaconda3\lib\site-packages (from matplotlib!=
3.0.0,>=2.0.2->yellowbrick) (2.8.2)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\haya\anaconda3\lib\site-packages (from matplotlib!=3.0.
0,>=2.0.2->yellowbrick) (3.0.9)
Requirement already satisfied: packaging>=20.0 in c:\users\haya\anaconda3\lib\site-packages (from matplotlib!=3.0.0,
>=2.0.2->yellowbrick) (21.3)
Requirement already satisfied: joblib>=0.11 in c:\users\haya\anaconda3\lib\site-packages (from scikit-learn>=1.0.0->
yellowbrick) (1.3.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\haya\anaconda3\lib\site-packages (from scikit-learn>
=1.0.0->yellowbrick) (2.2.0)
Requirement already satisfied: six>=1.5 in c:\users\haya\anaconda3\lib\site-packages (from python-dateutil>=2.7->mat
plotlib!=3.0.0,>=2.0.2->yellowbrick) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [58]:
```
import sklearn                                          # Train and evaluate machine learning models.
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.cluster import AgglomerativeClustering
from collections import Counter                         # Count objects in containers.
import pickle                                           # Save Python objects as binary files.
import warnings                                         # Suppress warnings.
warnings.filterwarnings('ignore')
import yellowbrick                                      # Visualize elbow and silhouette plots.
from yellowbrick.cluster import KElbowVisualizer
from yellowbrick.cluster import SilhouetteVisualizer
# Summarize software libraries used.
print('Libraries used in this project:')
print('- Python {}'.format(sys.version))
print('- pandas {}'.format(pd.__version__))
print('- Matplotlib {}'.format(matplotlib.__version__))
print('- Seaborn {}'.format(sns.__version__))
print('- Yellowbrick {}'.format(yellowbrick.__version__))
print('- scikit-learn {}'.format(sklearn.__version__))
```

```
Libraries used in this project:
- Python 3.9.13 (main, Aug 25 2022, 23:51:50) [MSC v.1916 64 bit (AMD64)]
- pandas 1.4.4
- Matplotlib 3.5.2
- Seaborn 0.11.2
- Yellowbrick 1.5
- scikit-learn 1.0.2
```

In [59]:
```
kmeans= KMeans(n_clusters = 2, random_state = 10)

kmeans.fit(covid_data_2)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_4936\1017875545.py in <module>
      1 kmeans= KMeans(n_clusters = 2, random_state = 10)
      2
----> 3 kmeans.fit(covid_data_2)

~\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py in fit(self, X, y, sample_weight)
   1135             Fitted estimator.
   1136         """
-> 1137         X = self._validate_data(
   1138             X,
   1139             accept_sparse="csr",

~\anaconda3\lib\site-packages\sklearn\base.py in _validate_data(self, X, y, reset, validate_separately, **check_para
ms)
    564             raise ValueError("Validation should be done on X, y or both.")
    565         elif not no_val_X and no_val_y:
--> 566             X = check_array(X, **check_params)
    567             out = X
    568         elif no_val_X and not no_val_y:

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in check_array(array, accept_sparse, accept_large_sparse,
dtype, order, copy, force_all_finite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, estimator)
    798
    799         if force_all_finite:
--> 800             _assert_all_finite(array, allow_nan=force_all_finite == "allow-nan")
    801
    802     if ensure_min_samples > 0:

~\anaconda3\lib\site-packages\sklearn\utils\validation.py in _assert_all_finite(X, allow_nan, msg_dtype)
    112         ):
    113             type_err = "infinity" if allow_nan else "NaN, infinity"
--> 114             raise ValueError(
    115                 msg_err.format(
    116                     type_err, msg_dtype if msg_dtype is not None else X.dtype

ValueError: Input contains NaN, infinity or a value too large for dtype('float64').
```

In [ ]:
```
y_kmeans = kmeans.predict(covid_data_scaled)
results = pd.DataFrame(covid_data_2)
results.insert(0,'cluster', y_kmeans)
results.head()
```

In [ ]:
```
def cluster_bar(cluster_labels):
    """Create a bar chart to show number of users in each cluster."""
    pd.DataFrame(Counter(cluster_labels).most_common()). \
    set_index(0).plot.bar(legend = None)

    plt.title('Distribution of Clusters')
    plt.xlabel('Cluster ID')
    plt.xticks(rotation = 0)
    plt.ylabel('Number of people in each cluster');

cluster_bar(y_kmeans)
```

In [ ]:
```
n_clusters= 5
agglom = AgglomerativeClustering(n_clusters = n_clusters,
                                 affinity = 'euclidean',
                                 linkage = 'single')

agglom.fit(covid_data_2)
y_agglom = agglom.fit_predict(covid_data_2)
results['cluster']= y_agglom
results.head()
```

In [ ]:
```
cluster_bar(y_agglom)
```

In [ ]:
```
pca = PCA(n_components = 2, random_state = 1)

pca.fit(covid_data_2)

reduced = pca.transform(covid_data_2)


reduced_df = pd.DataFrame(reduced, columns = ['PCA1', 'PCA2'])

reduced_df
```

In [ ]:
```
pca_df = pd.concat([reduced_df, pd.DataFrame(y_kmeans)], axis =1).rename(columns={0: 'cluster'})
pca_df
```

In [ ]:
```
cmap = sns.color_palette('tab10', n_colors=n_clusters, desat=0.5)

sns.scatterplot(x='PCA1', y='PCA2',
                hue = 'cluster', data = pca_df[::25],
                palette = cmap, legend = True)

plt.title('k-means clustering with 2 dimensions');
```

In [ ]:
```
elbow = KElbowVisualizer(kmeans, k=(1,20))

elbow.fit(covid_data_2)
elbow.poof();
```

In [ ]:
```
from yellowbrick.cluster import SilhouetteVisualizer
silhouette=SilhouetteVisualizer(KMeans(2, random_state=10))
silhouette.fit(covid_data_2)
silhouette.poof();
```

In [ ]:
```
print("number of clusters:", silhouette.n_clusters_)

print("clusters score:", silhouette.silhouette_score_)
```

In [ ]:
```
#Retrain the model with optimal number of clusters :
kmeans= KMeans(n_clusters = 2, random_state = 10)

kmeans.fit(covid_data_2)
y_kmeans = kmeans.predict(covid_data_2)
```

In [ ]:
```
results['cluster'] = y_kmeans
results
```

In [ ]:
```
cluster_bar(y_kmeans)
```

In [ ]:
```
cluster1= results[results.cluster==1]
cluster1.describe()
```

In [ ]:
```
cluster0= results[results.cluster==0]
cluster0.describe()
```

In [ ]:
```
covid_data_cleaned.to_pickle('covid_data_cleaned.pickle')
```

In [ ]:
```
pickle.dump(kmeans, open('kmeans.pickle', 'wb'))
```

In [62]:
```
q1 = np.percentile(covid_data_encoded.total_cases,25)
q3= np.percentile(covid_data_encoded.total_cases,75)
iqr = q3-q1

lb = q1-iqr*1.5
ub= q3+iqr*1.5

print("lower bound :", round(lb,2))
print("upper bound :", round(ub,2))
```

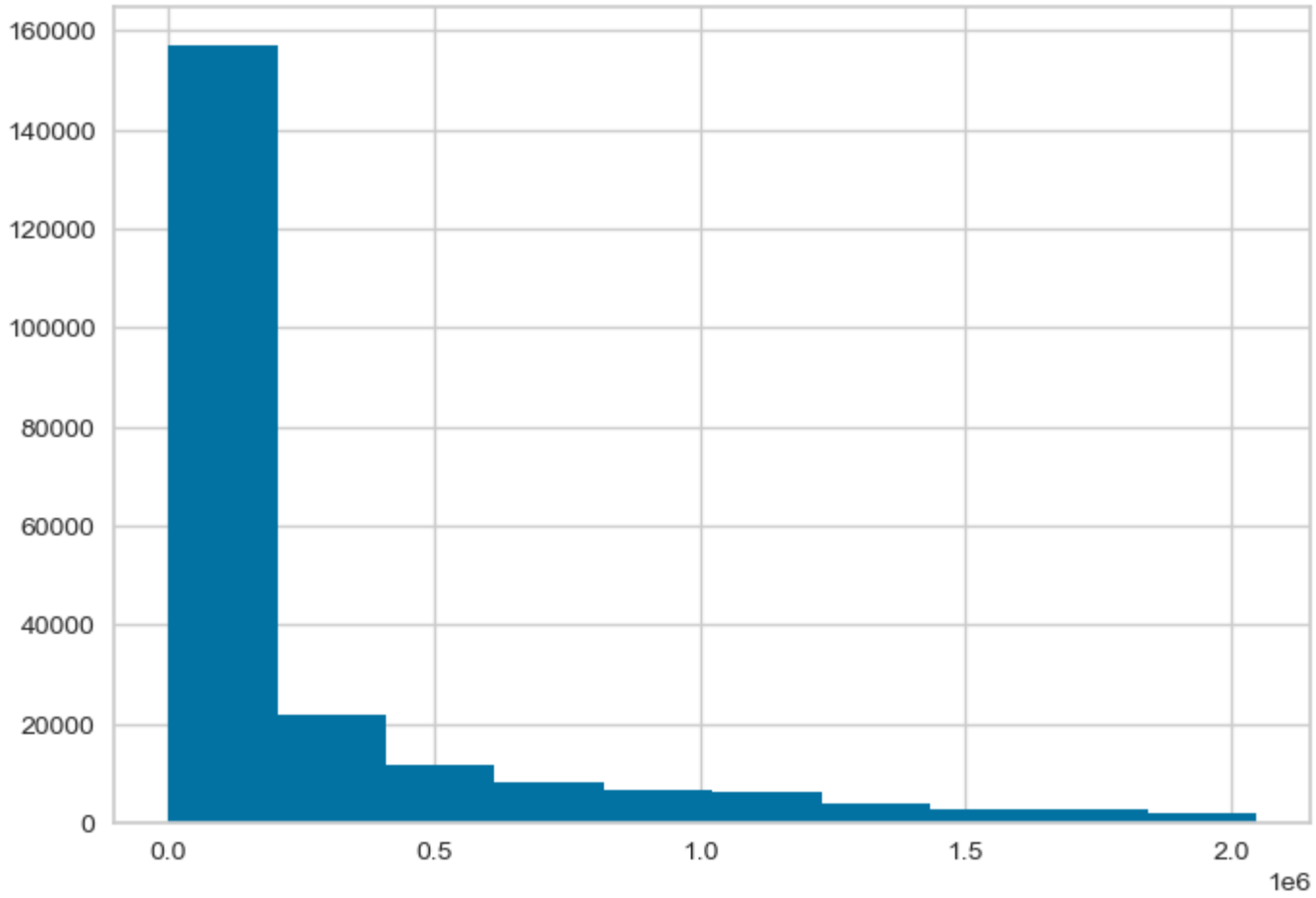
```
lower bound : -1209416.88
upper bound : 2047212.12
```

In [64]:
```
covid_data_wout_outliers = covid_data_encoded[(covid_data_encoded.total_cases < ub) & (covid_data_encoded.total_cases

covid_data_wout_outliers.shape
```

Out[64]:
```
(223168, 540)
```

In [65]:
```
covid_data_wout_outliers.total_cases.hist()
```

Out[65]:
```
<AxesSubplot:>
```


160000
140000
120000
100000
80000
60000
40000
20000
0
0.0
0.5
1.0
1.5
2.0
1e6

In [ ]: