GEO MAPPING

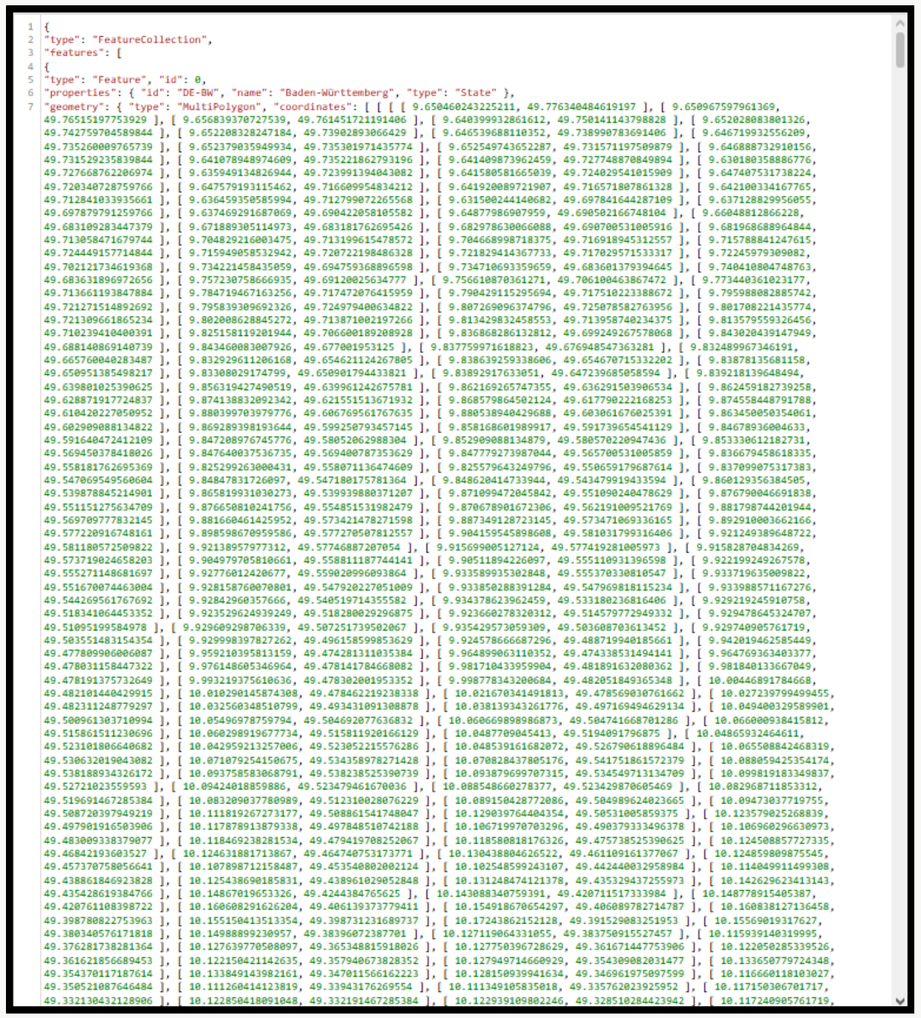
05.03.2026

Arian Temouri, Annemike Rörig,
Elina Winkler, Fabian Zirkel,
Julianne Kitzinger

Bild: https://www.heymundo.de/reise/deutschland-frankfurt-tagescamp-lerne-programmieren-mit-phython-mit-unterstuetzung-der-ki-tagescamp-2664

INHALT

DATENSET - BUNDESLAENDER.JSON



Erklärung

- **GeoJSON-Datei**
  - *FeatureCollection - Properties - Geometry*
  - geografischen Grenzen der Bundesländer
  - Speicherung Koordinaten als eigenes Objekt = *Feature*

Benutzung - Aufgabe 1

- Form
- Fläche der Bundesländer
  - Einfärbung *Choroplethenkarte*
- Namen zur Zuordnung

DATENSET - COVID_DE.CSV

```
state,date,cases,recovered,deaths
Baden-Wuerttemberg,2020-03-27,1243,1193,50
Baden-Wuerttemberg,2020-03-28,1006,949,57
Baden-Wuerttemberg,2020-04-03,1270,1189,81
Baden-Wuerttemberg,2020-10-18,540,532,8
Baden-Wuerttemberg,2020-10-22,1988,1961,27
Baden-Wuerttemberg,2020-10-27,2118,2096,22
Baden-Wuerttemberg,2020-10-30,2705,2690,15
Baden-Wuerttemberg,2020-11-03,2793,2758,35
Baden-Wuerttemberg,2020-11-07,2386,2357,29
Baden-Wuerttemberg,2020-11-10,2431,2398,33
Baden-Wuerttemberg,2020-11-15,1129,1089,40
Baden-Wuerttemberg,2020-11-17,2667,2607,60
Baden-Wuerttemberg,2020-11-18,3240,3169,71
Baden-Wuerttemberg,2020-11-19,3080,2990,90
Baden-Wuerttemberg,2020-11-25,2829,2756,73
Baden-Wuerttemberg,2020-12-05,2454,2402,52
Baden-Wuerttemberg,2020-12-13,1929,1881,48
Baden-Wuerttemberg,2020-12-14,2258,2178,80
```

Erklärung

- CSV-Datei liefert die Zahlen
  - Zeitreihen-Tabelle

- state = Name des Bundeslandes
- date = Datum der Meldung
- cases = Anzahl gemeldete Fälle
- recovered = Anzahl der Genesenen
- deaths = Anzahl der Todesfälle

Benutzung (*Niedersachsen*) - Aufgabe 2

- **Grafik 1A** - Verhältnis zwischen Todesfällen
- **Grafik 1B** - tägliche Todesfälle
- **Grafik 2** - Corona Wellen

DATENSET - COVID_PER_STATE.CSV

```
1  state,cases,recovered,deaths
2  Baden-Württemberg,5009778,4965489,18985
3  Bayern,6668169,6607699,28116
4  Berlin,1418585,1405612,5446
5  Brandenburg,1104240,1092273,6438
6  Bremen,301050,298768,956
7  Hamburg,802806,796205,3544
8  Hessen,2887077,2853678,12179
9  Mecklenburg-Vorpommern,706888,700439,2733
10 Niedersachsen,3820165,3777849,13190
11 Nordrhein-Westfalen,7918225,7830806,30862
12 Rheinland-Pfalz,1742703,1723709,6833
13 Saarland,484735,480656,2087
14 Sachsen,1944901,1920590,16744
15 Sachsen-Anhalt,954211,943045,6236
16 Schleswig-Holstein,1167600,1158395,3443
17 Thüringen,878491,868273,8224
18
19
```

Erklärung

- Gesamtsumme Fälle, Genesenen und Todesfälle je Bundesland
- state = Bundesland
- cases = Gesamtzahl der Fälle
- recovered = Gesamtzahl der Genesenen
- deaths = Gesamtzahl der Todesfälle

Benutzung - Aufgabe 1 ; 3

- **Aufgabe 1**
  - Wert *cases* bestimmt Farbe der Karte
- **Aufgabe 3**
  - Wert *recovered* sortiert

IMPORTS

```
import pandas as pd
import folium
import json
import matplotlib.pyplot as plt
import seaborn as sns
```

import pandas **as** pd

- Daten lesen und bearbeiten

import folium

- Kartentool (Erstellen, Bibliothek, ...)

import json

- damit man GeoJSON-Datei lesen kann

import matplotlib.pyplot as plt

- für Diagramme und Graphen

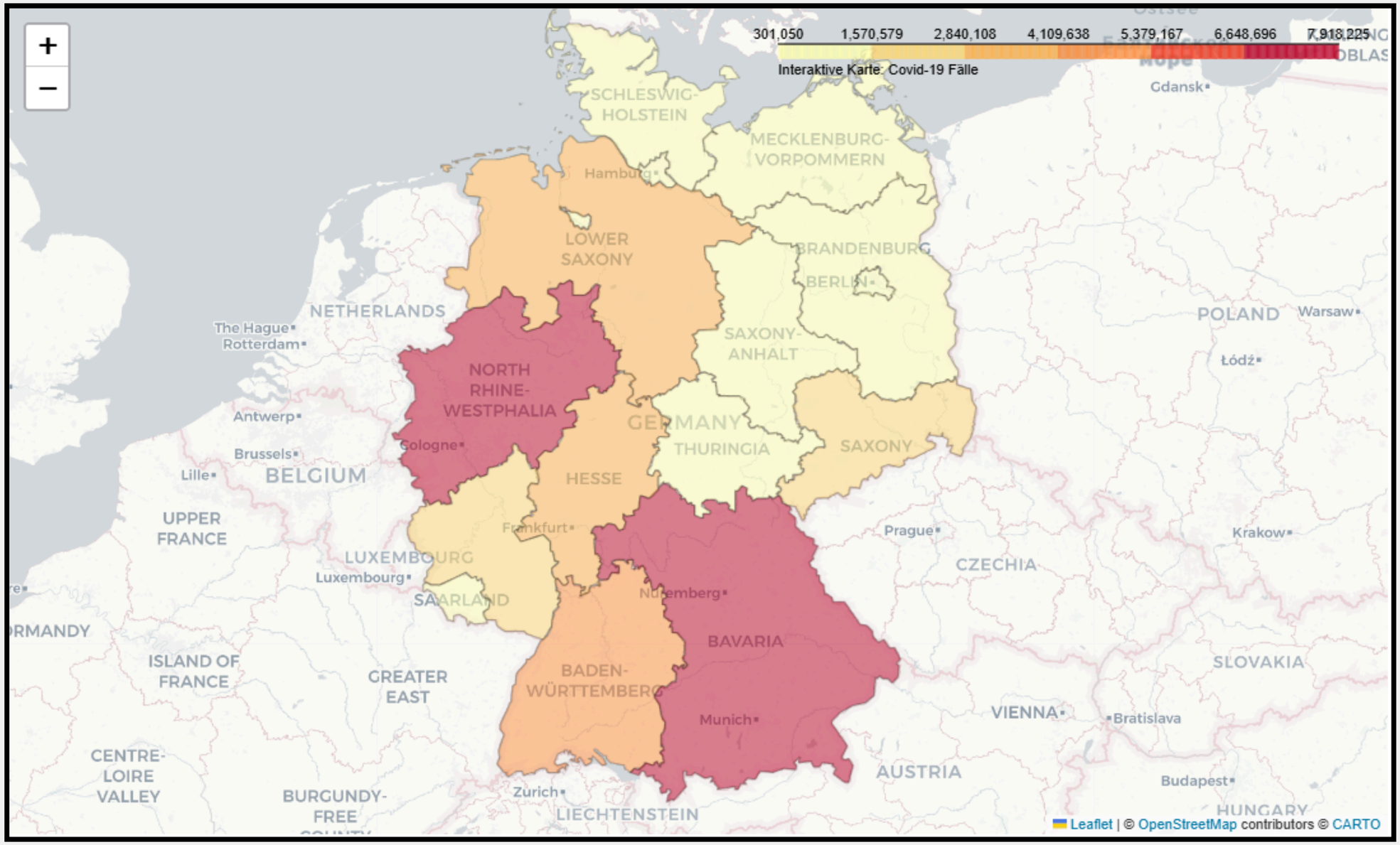
import seaborn as sns

- für die Gestaltung von Diagrammen

AUFGABE 1

Verwende die GeoJSON-Datei bundeslaender.json und den Datensatz covid_per_state.csv, um eine (Folium-)Choroplethenkarte mit den passenden Daten darzustellen.

ERGEBNIS


301,050
1,570,579
2,840,108
4,109,638
5,379,167
6,648,696
7,918,225
Interaktive Karte: Covid-19 Fälle
SCHLESWIG-HOLSTEIN
MECKLENBURG-VORPOMMERN
Hamburg
LOWER SAXONY
BRANDENBURG
BERLIN
NETHERLANDS
The Hague
Rotterdam
SAXONY-ANHALT
NORTH RHINE-WESTPHALIA
Antwerp
Brussels
Cologne
GERMANY
THURINGIA
SAXONY
Lille
BELGIUM
HESSE
UPPER FRANCE
Frankfurt
LUXEMBOURG
Luxembourg
SAARLAND
Nuremberg
BAVARIA
ISLAND OF FRANCE
GREATER EAST
BADEN-WÜRTTEMBERG
Munich
CENTRE-LOIRE VALLEY
BURGUNDY-FREE
Zurich
LIECHTENSTEIN
AUSTRIA
VIENNA
Bratislava
Budapest
HUNGARY
SLOVAKIA
CZECHIA
Prague
Krakow
POLAND
Warsaw
Łódź
Gdansk
Leaflet | © OpenStreetMap contributors © CARTO

CODE

```
#CSV Datei einlesen
covid = pd.read_csv("covid_per_state.csv")

#GeoJSON-Datei einlesen
with open ("bundeslaender.json", "r", encoding="utf-8") as f:
    geo_data = json.load(f)

#Karte erstellen
my_map = folium.Map(
    location=[51.163409, 10.447718],
    zoom_start=5.5,
    tiles="CartoDB Positron"
)

#Choroplethenkarte erstellen/hinzufügen
folium.Choropleth(
    geo_data=geo_data,
    data=covid,
    columns=["state", "cases"],
    key_on="feature.properties.name",
    fill_color="YlOrRd",
    fill_opacity=0.5,
    line_opacity=0.3,
    legend_name="Interaktive Karte: Covid-19 Fälle"
).add_to(my_map)

my_map
```

ZUSAMMENFASSUNG

- Darstellung der Covid-Fallzahlen nach Bundesland
- Abhängig von der Fallzahl farblich eingefärbt
- Helle Farben (niedrige Fallzahlen) → dunkle Rote Farben (hohe Fallzahlen)

Direkt sichtbar:

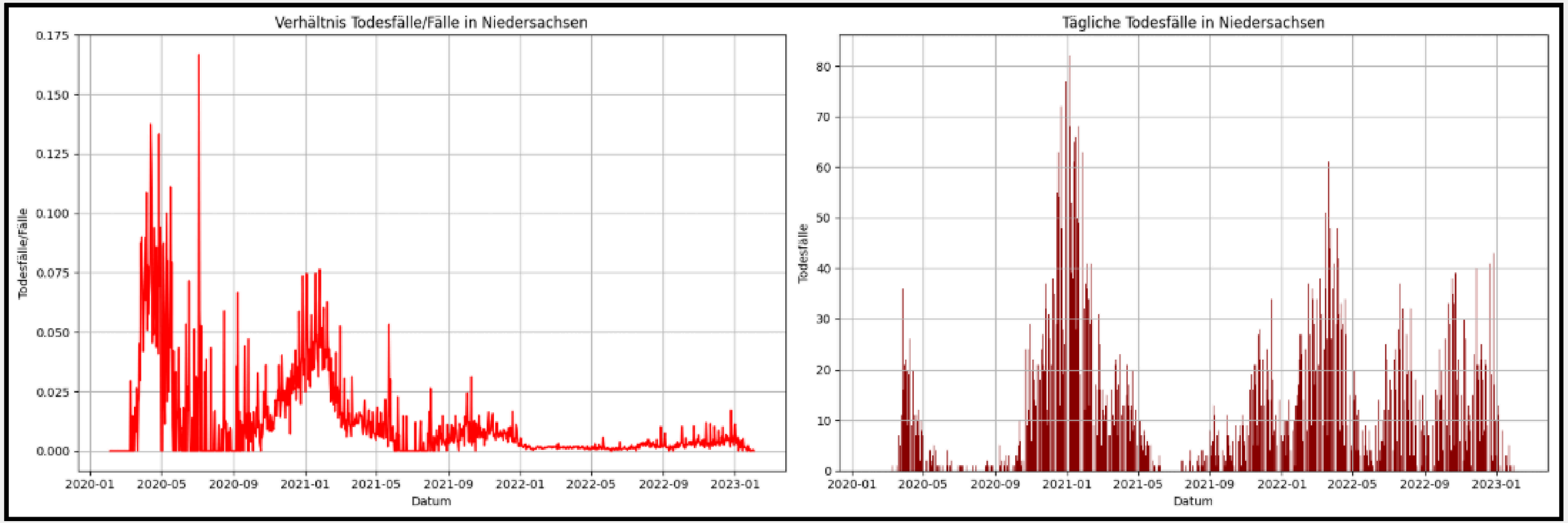
- regionale Unterschiede / räumliche Muster
- Bevölkerungsgröße und Dichte im Zusammenhang mit den Zahlen

→ Karte ermöglicht einen schnellen räumlichen Vergleich / Überblick zwischen den Bundsländern

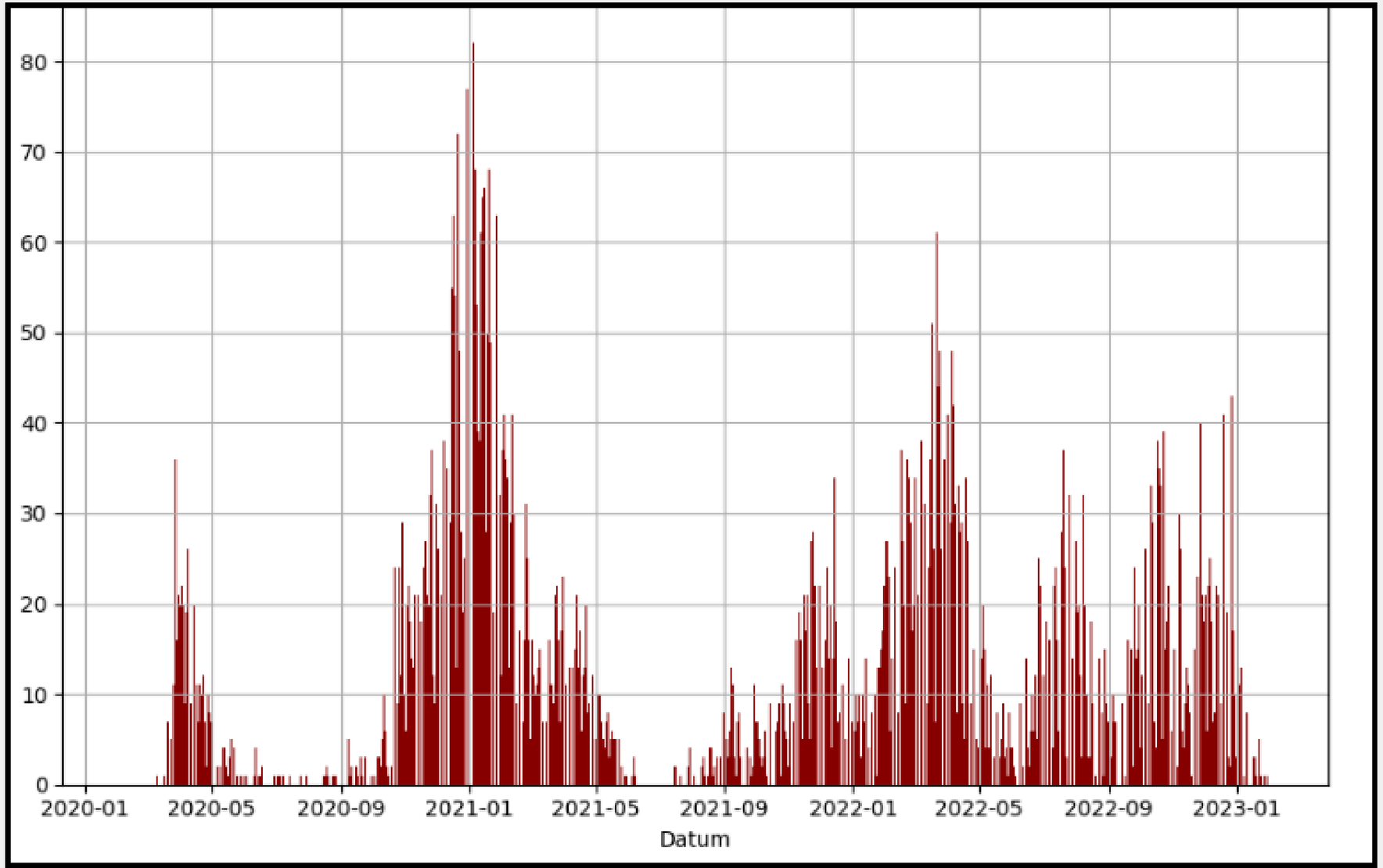
AUFGABE 2A

Zeigen Sie das Verhältnis von Todesfällen zu Infektionen in Niedersachsen und ermitteln Sie den/ die Zeitraum /Zeiträume mit den meisten Todesfällen im Zusammenhang mit Covid-19.

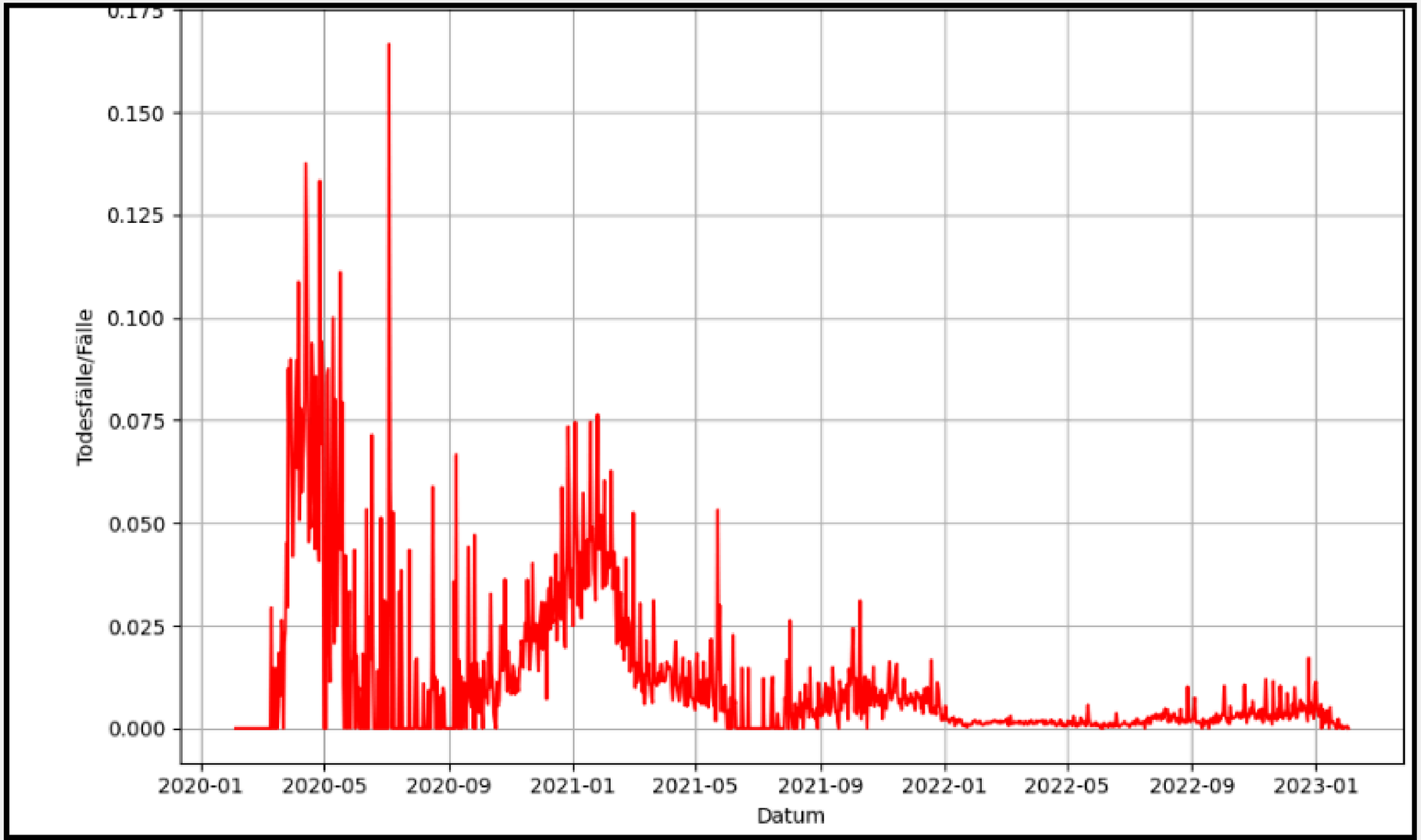
ERGEBNIS (2A)


Verhältnis Todesfälle/Fälle in Niedersachsen
Todesfälle/Fälle
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
2020-01
2020-05
2020-09
2021-01
2021-05
2021-09
2022-01
2022-05
2022-09
2023-01
Datum
Tägliche Todesfälle in Niedersachsen
Todesfälle
80
70
60
50
40
30
20
10
0
2020-01
2020-05
2020-09
2021-01
2021-05
2021-09
2022-01
2022-05
2022-09
2023-01
Datum

TÄGLICHE TODESFÄLLE IN NIEDERSACHSEN



VERHÄLTNIS TODESFÄLLE / KRANKHEITSFÄLLE



CODE

```
#CSV-Datei laden und die Spalte "date" als Datumsformat speichern
df = pd.read_csv("covid_de.csv", parse_dates=["date"])

#Nur Zeilen auswählen wo "Niedersachsen" das Bundesland ist
df_nd = df[df["state"] == "Niedersachsen"].copy()

#"Verhältnis Todesfälle zu Fälle berechnen. Formel: "Deaths" / "cases"
df_nd["death_ratio"] = df_nd["deaths"] / df_nd["cases"]

#Daten sortieren nach "Deaths" absteigend
top_deaths = df_nd.sort_values(by="deaths", ascending=False)

#Daten nach chronologischer Reihenfolge sortieren
df_nd = df_nd.sort_values(by='date')

#Diagramm Bereich erstellen, 1 Zeile, 2 Spalten für zwei Diagramme nebeneinander
fig, axes = plt.subplots(1, 2, figsize=(18,6))

#Linkes Diagramm: Todesfälle / Fälle Verhältnis in Niedersachsen
axes[0].plot(df_nd['date'], df_nd['death_ratio'], color='red')
axes[0].set_title('Verhältnis Todesfälle/Fälle in Niedersachsen')
axes[0].set_xlabel('Datum')
axes[0].set_ylabel('Todesfälle/Fälle')
axes[0].grid(True)

#Rechtes Diagramm: Absolute Todesfälle in Niedersachsen
axes[1].bar(df_nd['date'], df_nd['deaths'], color='darkred')
axes[1].set_title('Tägliche Todesfälle in Niedersachsen')
axes[1].set_xlabel('Datum')
axes[1].set_ylabel('Todesfälle')
axes[1].grid(True)

plt.tight_layout()
plt.show()
```

ZUSAMMENFASSUNG (2A)

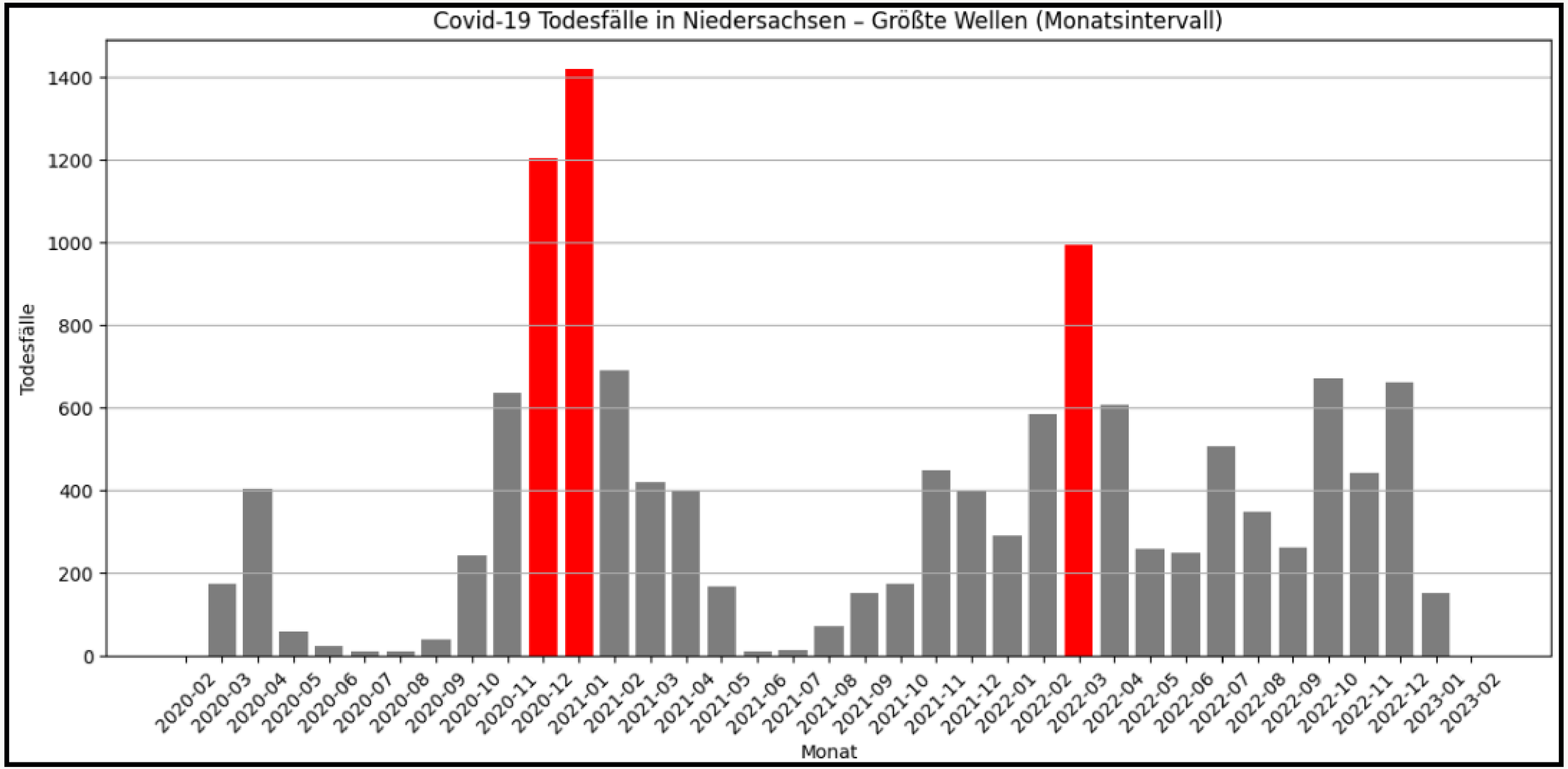
Die Plots zeigen:

- Mehrere COVID - Wellen in Niedersachsen
- Höhepunkt der Sterblichkeit im Winter 2020/21
- Deutlich sinkende relative Sterblichkeit im Zeitverlauf
- Vermutlich starker Einfluss durch Impfungen und Schutzmaßnahmen

AUFGABE 2B

Zeigen Sie das Verhältnis von Todesfällen zu Infektionen in Niedersachsen und ermitteln Sie den/ die Zeitraum /Zeiträume mit den meisten Todesfällen im Zusammenhang mit Covid-19.

ERGEBNIS


Covid-19 Todesfälle in Niedersachsen – Größte Wellen (Monatsintervall)
Todesfälle
0
200
400
600
800
1000
1200
1400
Monat
2020-02
2020-03
2020-04
2020-05
2020-06
2020-07
2020-08
2020-09
2020-10
2020-11
2020-12
2021-01
2021-02
2021-03
2021-04
2021-05
2021-06
2021-07
2021-08
2021-09
2021-10
2021-11
2021-12
2022-01
2022-02
2022-03
2022-04
2022-05
2022-06
2022-07
2022-08
2022-09
2022-10
2022-11
2022-12
2023-01
2023-02

CODE

```
#Daten nach chronologischer Reihenfolge sortieren (Wiederholung, vermutlich nicht nötig)
df_nd = df_nd.sort_values(by='date')

#Monate aus Datum erzeugen, erstellen einer neuen Spalte "year_month": Datum in Monatsperiode (YYYY-MM)
df_nd['year_month'] = df_nd['date'].dt.to_period('M')

#Todesfälle pro Monat berechnen, Tage eines Monats zusammenfassen und aufsummieren
monthly_deaths = df_nd.groupby('year_month')['deaths'].sum().reset_index()

#3 Monate mit den meisten Todesfällen finden
top_3_months = monthly_deaths.nlargest(3, 'deaths')['year_month'].tolist()

#Farben für das Diagramm festlegen: Wenn Top 3 dann "rot", sonst "grau"
colors = ['red' if ym in top_3_months else 'grey' for ym in monthly_deaths['year_month']]

#Grafik erstellen
plt.figure(figsize=(14,6))
plt.bar(monthly_deaths['year_month'].astype(str), monthly_deaths['deaths'], color=colors)
plt.title('Covid-19 Todesfälle in Niedersachsen – Größte Wellen (Monatsintervall)')
plt.xlabel('Monat')
plt.ylabel('Todesfälle')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.show()
```

ZUSAMMENFASSUNG (2B)

Die Plots zeigen:

- COVID-19-Todesfälle in Niedersachsen, im Zeitraum 2020-2023 in Balkendiagramm
- Rotmarkierte Balken zeigen drei Monate, mit den höchsten Todeszahlen
- **Ergebnis:** Höchste Todeszahlen im Winter 2020/2021 & Beginn 2022

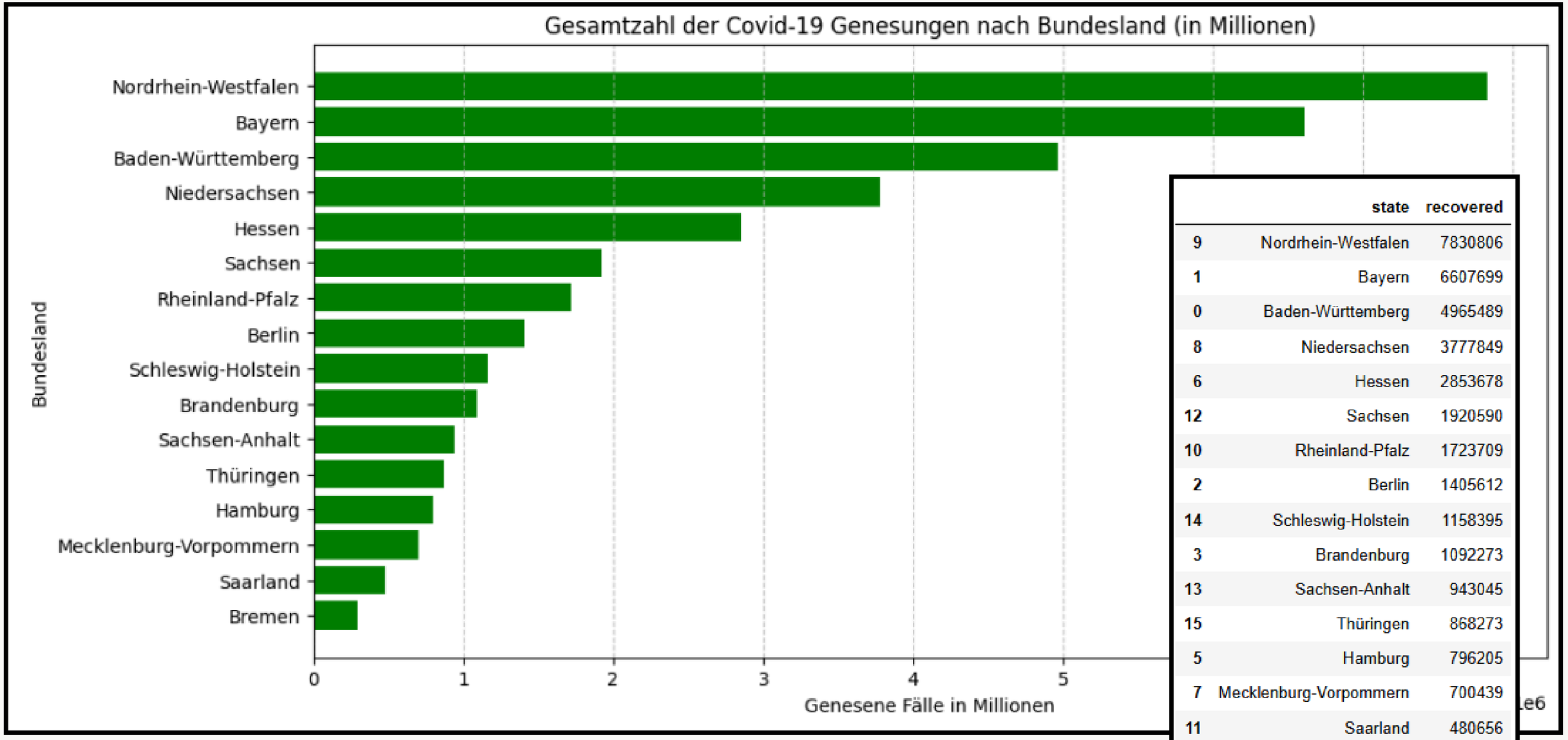
→ COVID-19 verläuft in Wellen, wobei die stärkste Belastung im Winter erfolgt

→ Zusammengefasst bedeutet das: Deutlich erkennbare Hochphasen der Todesrate in den Wintermonaten 2020/21 sowie Anfang 2022, was wohlmöglich auf saisonale Effekte zurückzuführen ist

AUFGABE 3

Sortiere die Bundesländer nach ihren Genesungszahlen in einer sinnvollen Reihenfolge.

ERGEBNIS



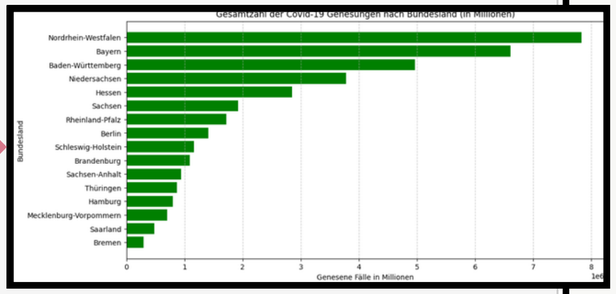
	state	recovered
9	Nordrhein-Westfalen	7830806
1	Bayern	6607699
0	Baden-Württemberg	4965489
8	Niedersachsen	3777849
6	Hessen	2853678
12	Sachsen	1920590
10	Rheinland-Pfalz	1723709
2	Berlin	1405612
14	Schleswig-Holstein	1158395
3	Brandenburg	1092273
13	Sachsen-Anhalt	943045
15	Thüringen	868273
5	Hamburg	796205
7	Mecklenburg-Vorpommern	700439
11	Saarland	480656
4	Bremen	298768

CODE

```python
#CSV-Datei einlesen
df_state = pd.read_csv("covid_per_state.csv")

#Gesamt-Genesungen pro Bundesland berechnen
recovered_sum = df_state.groupby('state')['recovered'].sum().reset_index()

#Nach Genesungen absteigend sortieren
recovered_sorted = recovered_sum.sort_values(by='recovered', ascending=False)

#Ergebnis anzeigen
recovered_sorted
```

	state	recovered
9	Nordrhein-Westfalen	7830806
1	Bayern	6607699
0	Baden-Württemberg	4965489
8	Niedersachsen	3777849
6	Hessen	2853678
12	Sachsen	1920590
10	Rheinland-Pfalz	1723709
2	Berlin	1405612
14	Schleswig-Holstein	1158395
3	Brandenburg	1092273
13	Sachsen-Anhalt	943045
15	Thüringen	868273
5	Hamburg	796205
7	Mecklenburg-Vorpommern	700439
11	Saarland	480656
4	Bremen	298768

```python
#Grafik zur Veranschaulichung aus den Daten erstellen
plt.figure(figsize=(12,6))
plt.barh(recovered_sorted['state'], recovered_sorted['recovered'], color='green')
plt.xlabel('Genesene Fälle in Millionen')
plt.ylabel('Bundesland')
plt.title('Gesamtzahl der Covid-19 Genesungen nach Bundesland (in Millionen)')
plt.gca().invert_yaxis()
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.show()
```



ZUSAMMENFASSUNG

1.Code Abschnitt

- Gesamt-Genesung pro Bundesland
- absteigend sortiert
- Tabelle

2.Code Abschnitt

- horizontales Balkendiagramm
- Bremen: geringste Genesungen
- NRW: höchste Genesung

VIELEN DANK