Project: pay gap

Kenji, Helen, Annelie, Adrian

Ablauf

- Problematik des Datensatzes
- Erstellung einer geeigneten Karte von Deutschland
- Vergleich zwischen Ost- und Westdeutschland
- Zeitliche Entwicklung des Pay Gap
- Heatmap der (nicht) vorhanden Daten

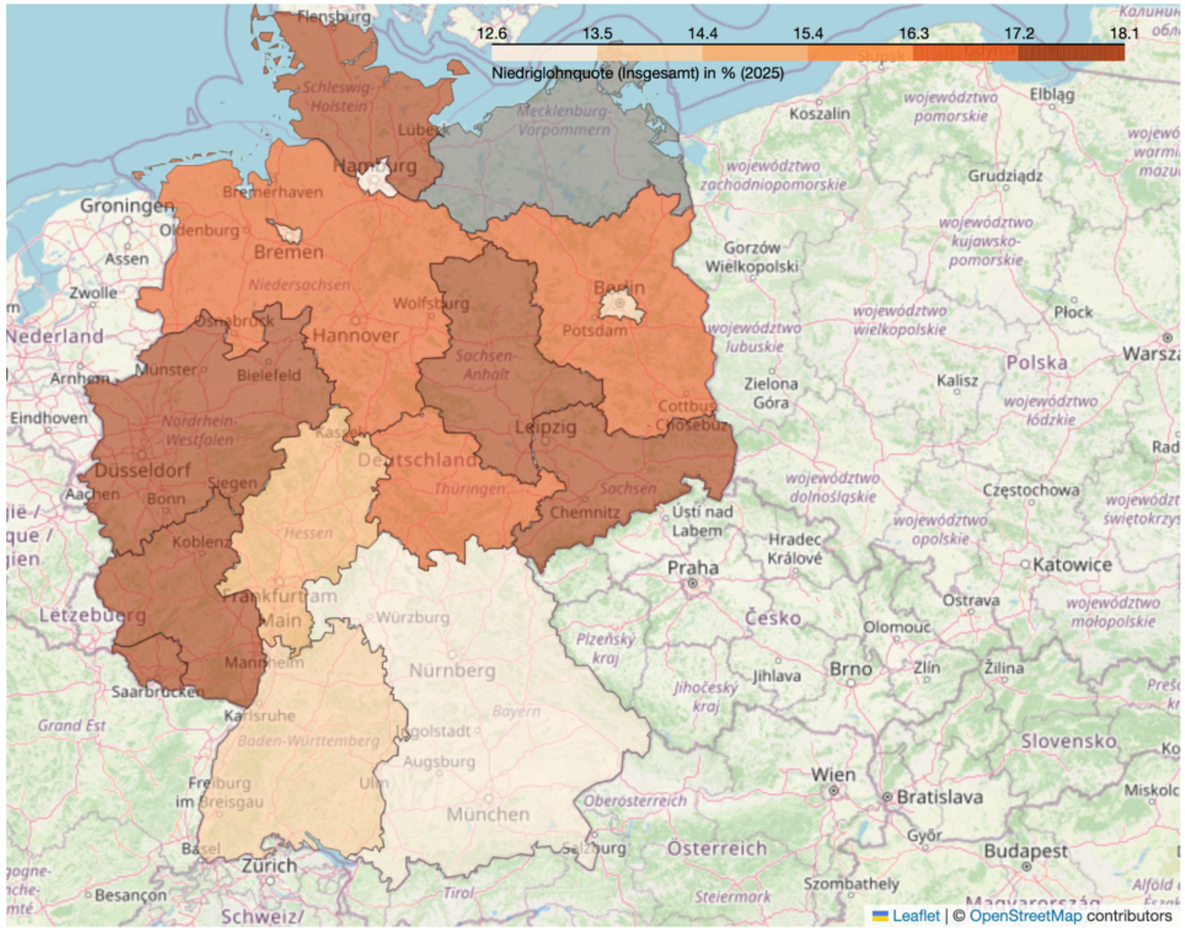
Problematik des Datensatzes

Hamburg		
WZ08-M Freiberufliche, wiss. u. techn. Dienstleistungen	126	/
WZ08-R Kunst, Unterhaltung und Erholung	(19) ()	/
Insgesamt	1.122	/
Hessen		
WZ08-M Freiberufliche, wiss. u. techn. Dienstleistungen	228	(25) ()
WZ08-R Kunst, Unterhaltung und Erholung	(34) ()	/
Insgesamt	3.071	546
Mecklenburg-Vorpommern		
WZ08-M Freiberufliche, wiss. u. techn. Dienstleistungen	(24) ()	/
WZ08-R Kunst, Unterhaltung und Erholung	(8) ()	/
Insgesamt	630	152
Niedersachsen		
WZ08-M Freiberufliche, wiss. u. techn. Dienstleistungen	185	(22) ()
WZ08-R Kunst, Unterhaltung und Erholung	(49) ()	(22) ()
Insgesamt	3.636	778
Nordrhein-Westfalen		
WZ08-M Freiberufliche, wiss. u. techn. Dienstleistungen	571	(69) ()
WZ08-R Kunst, Unterhaltung und Erholung	(124) ()	(65) ()
Insgesamt	8.585	1.762

Zeichenerklärung

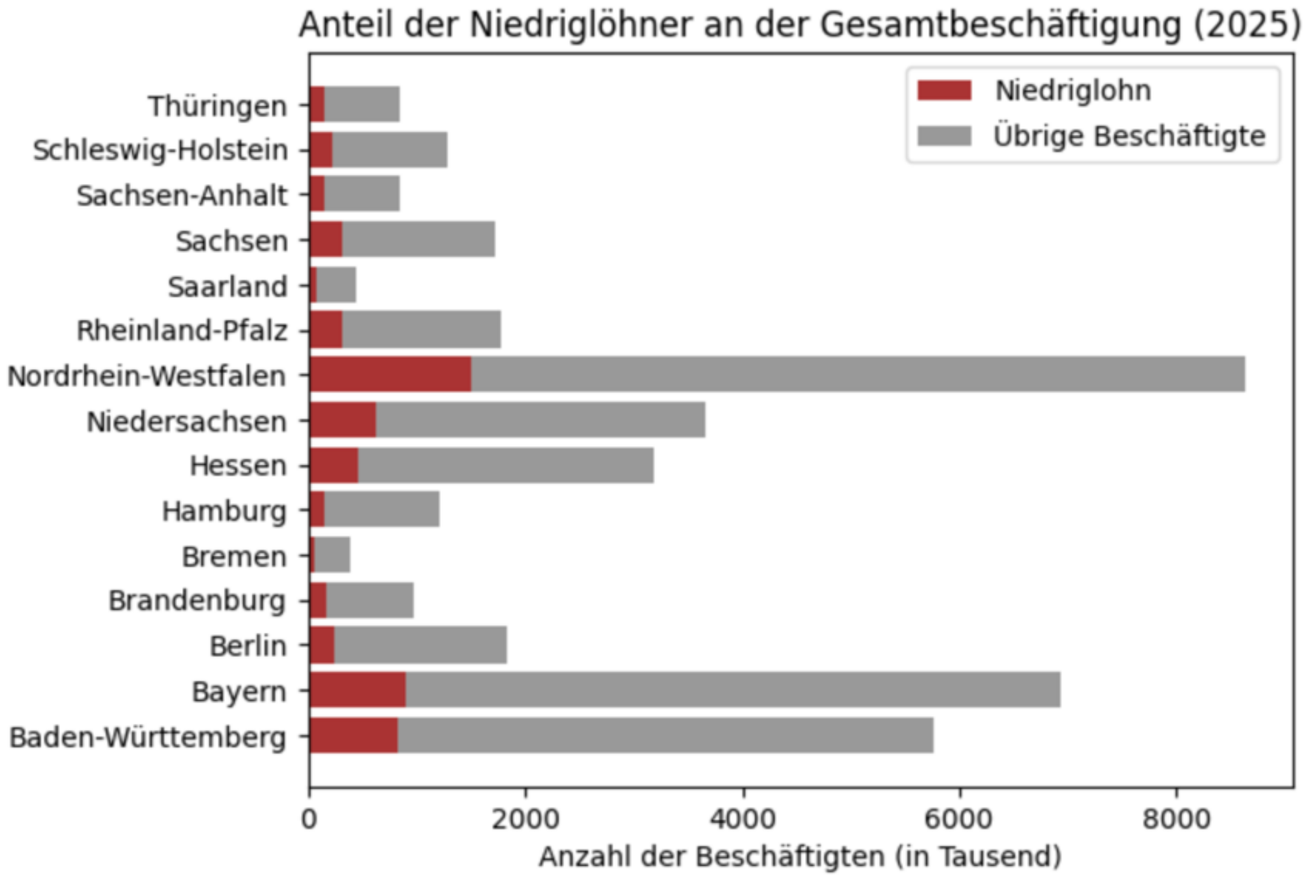
-	nichts vorhanden, genau Null oder ggf. zur Sicherstellung der statistischen Geheimhaltung auf Null geändert
e	endgültiger Wert
0	Zahlenwert von Null verschieden, jedoch so nahe an Null, dass auf Null gerundet
r	berichtigte Zahl
p	vorläufige Zahl
s	geschätzte Zahl
()	Aussagewert eingeschränkt, da der Zahlenwert statistisch relativ unsicher ist
/	keine Angaben, da Zahlenwert nicht sicher genug
.	Zahlenwert unbekannt oder geheim zu halten
...	Angabe fällt später an
x	Tabellenfach gesperrt, weil Aussage nicht sinnvoll

Karte von Deutschland



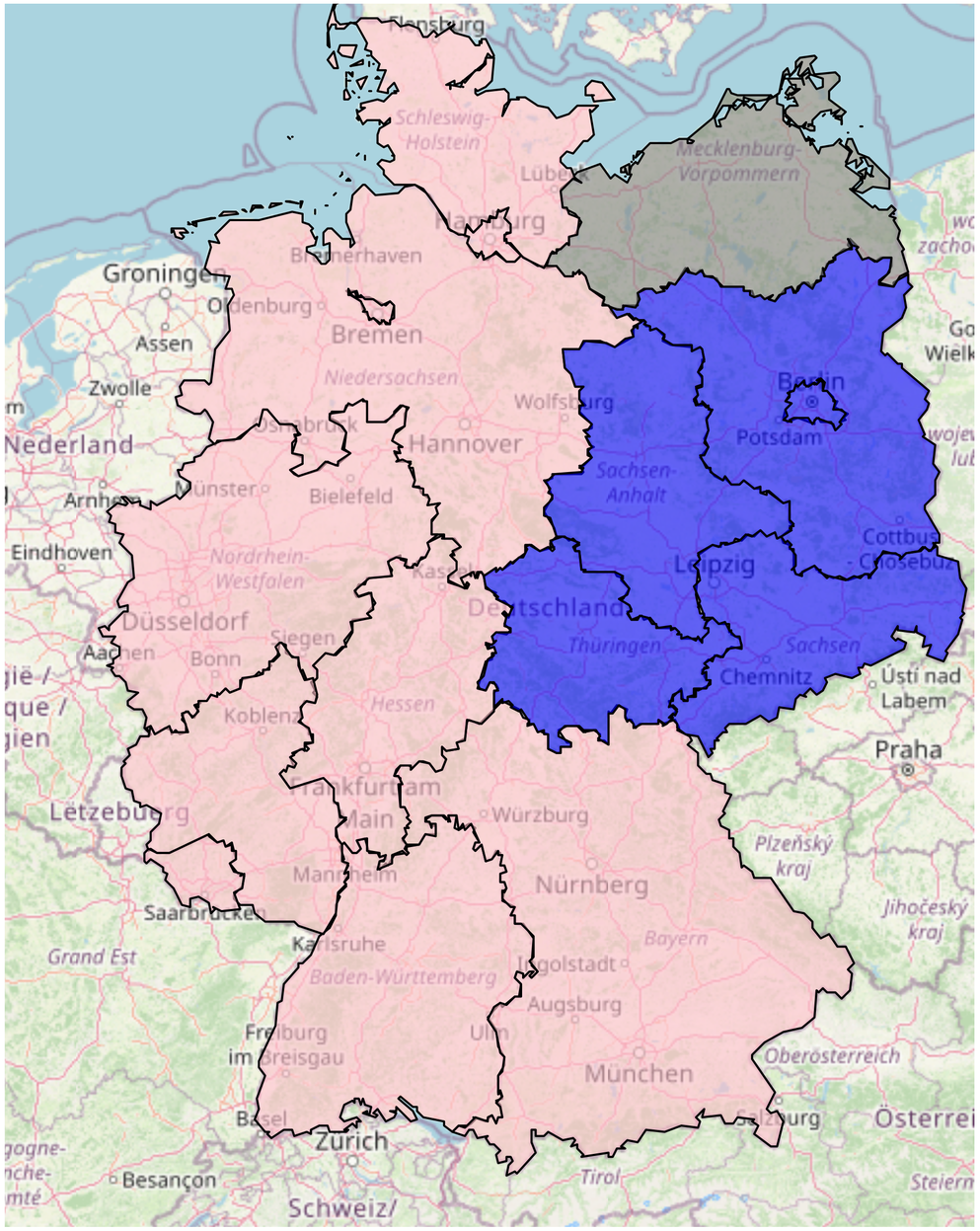

```python
import pandas as pd
import folium
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('62361-0050.csv')

# Kommawerte in "." ändern und als float statt str speichern, für "/" in Daten "NaN" setzen
df['Wert'] = pd.to_numeric(df['Wert'].astype(str).str.replace(',', '.'), errors='coerce')


# df auf 2025 und Zahlen für "Insgesamt" filtern
df_2025_insgesamt = df[
    (df['Jahr'] == 2025) &
    (df['Wirtschaftszweig'] == 'Insgesamt')
]

bundeslaender_liste = df_2025_insgesamt['Bundesland'].unique()
ergebnisse = []

# for-loop, pro Eintrag in "bundesländer_liste" gültige Daten aus "df_2025_insgesamt" in Liste "ergebnisse"
for bl in bundeslaender_liste:
    df_bl = df_2025_insgesamt[df_2025_insgesamt['Bundesland'] == bl]

    anteil_prozent = df_bl[df_bl['Messgröße'] == 'Anteil abh. Beschäftigungsverhältn.mit Niedriglohn']['Wert'].value
    absolut_niedrig = df_bl[df_bl['Messgröße'] == 'Abhäng. Beschäftigungsverhältnisse mit Niedriglohn']['Wert'].valu
    absolut_gesamt = df_bl[df_bl['Messgröße'] == 'Abhängige Beschäftigungsverhältnisse']['Wert'].values[0]

    if pd.notna(anteil_prozent) and pd.notna(absolut_niedrig) and pd.notna(absolut_gesamt):
        ergebnisse.append({
            'Bundesland': bl,
            'Niedriglohn_Prozent': anteil_prozent,
            'Niedriglohn_Absolut_tsd': absolut_niedrig,
            'Gesamt_Absolut_tsd': absolut_gesamt
        })
```

```python
df_clean = pd.DataFrame(ergebnisse)

# Folium-Map
m = folium.Map(location=[51.163409, 10.447718], zoom_start=6)

folium.Choropleth(
    geo_data='bundeslaender.json',
    data=df_clean,
    columns=['Bundesland', 'Niedriglohn_Prozent'],
    key_on='feature.properties.name',
    fill_color='Oranges',
    fill_opacity=0.6,
    line_opacity=0.3,
    nan_fill_color='grey',
    legend_name='Niedriglohnquote (Insgesamt) in % (2025)'
).add_to(m)

m
```




```python
# Balkendiagramm
niedrig = df_clean['Niedriglohn_Absolut_tsd']
uebrige = df_clean['Gesamt_Absolut_tsd'] - niedrig
laender = df_clean['Bundesland']

fig, ax = plt.subplots()

ax.barh(laender, niedrig, label='Niedriglohn', color='#aa3333')
ax.barh(laender, uebrige, left=niedrig, label='Übrige Beschäftigte', color='#999999')

ax.set_title('Anteil der Niedriglöhner an der Gesamtbeschäftigung (2025)')
ax.set_xlabel('Anzahl der Beschäftigten (in Tausend)')
ax.legend(loc='upper right')

plt.show()
```

Vergleich Ost- und Westdeutschland



In [5]:

```
import pandas as pd
import folium
import json

df = pd.read_csv('62361-0050.csv')


df['Wert'] = pd.to_numeric(df['Wert'].astype(str).str.replace(',', '.'), errors='coerce')

neue_bundeslaender = [
    'Berlin', 'Brandenburg', 'Mecklenburg-Vorpommern', 'Sachsen',
    'Sachsen-Anhalt', 'Thüringen',
]

df_2025_insgesamt = df[
    (df['Jahr'] == 2025) &
    (df['Wirtschaftszweig'] == 'Insgesamt')
]

bundeslaender_liste = df_2025_insgesamt['Bundesland'].unique()

ergebnisse = []

for bl in bundeslaender_liste:
    df_bl = df_2025_insgesamt[df_2025_insgesamt['Bundesland'] == bl]

    anteil_prozent = df_bl[df_bl['Messgröße'] == 'Anteil abh. Beschäftigungsverhältn.mit Niedriglohn']['Wert'].value

    absolut_niedrig = df_bl[df_bl['Messgröße'] == 'Abhäng. Beschäftigungsverhältnisse mit Niedriglohn']['Wert'].valu

    absolut_gesamt = df_bl[df_bl['Messgröße'] == 'Abhängige Beschäftigungsverhältnisse']['Wert'].values[0]

    if pd.notna(anteil_prozent) and pd.notna(absolut_niedrig) and pd.notna(absolut_gesamt):
        ergebnisse.append({
            'Bundesland': bl,
            'Niedriglohn_Prozent': anteil_prozent,
            'Niedriglohn_Absolut_tsd': absolut_niedrig,
            'Gesamt_Absolut_tsd': absolut_gesamt,
            'Region': 'Ost' if bl in ost_bundeslaender else 'West'
        })

df_clean = pd.DataFrame(ergebnisse)

vergleich = df_clean.groupby('Region').agg(
    Durchschnitt_Prozent=('Niedriglohn_Prozent', 'mean'),
    Gesamt_Niedriglohn_tsd=('Niedriglohn_Absolut_tsd', 'sum'),
    Gesamt_Beschaeftigt_tsd=('Gesamt_Absolut_tsd', 'sum')

).round(2)

# Gewichtete Gesamtquote berechnen: Niedriglohn-Jobs / alle Jobs × 100
# Genauer als der einfache Durchschnitt, da Bundesländer unterschiedlich groß sind
vergleich['Gewichtete_Quote'] = (
    vergleich['Gesamt_Niedriglohn_tsd'] / vergleich['Gesamt_Beschaeftigt_tsd'] * 100
).round(2)


with open('bundeslaender.json', 'r', encoding='utf-8') as f:
    geo_data = json.load(f)


for feature in geo_data['features']:
    bl_name = feature['properties']['name']
    match = df_clean[df_clean['Bundesland'] == bl_name]
    if not match.empty:
        feature['properties']['Niedriglohn_Prozent'] = f"{match['Niedriglohn_Prozent'].values[0]} %"
        feature['properties']['Region'] = match['Region'].values[0]
    else:
        feature['properties']['Niedriglohn_Prozent'] = 'keine Daten'
        feature['properties']['Region'] = '–'

m = folium.Map(location=[51.163409, 10.447718], zoom_start=6)
```

```
def style_function(feature):
    bl_name = feature['properties']['name']
    region = df_clean.loc[df_clean['Bundesland'] == bl_name, 'Region']
    if not region.empty:
        farbe = 'blue' if region.values[0] == 'Ost' else 'pink'
    else:
        farbe = 'grey'
    return {
        'fillColor': farbe,   # Füllfarbe der Fläche
        'color': 'black',     # Farbe Bundesland-Grenzen
        'weight': 1,          # Stärke der Grenzlinien (Pixel)
        'fillOpacity': 0.6    # Transparenz der Fläche (0 = unsichtbar, 1 = voll)
    }

folium.GeoJson(
    geo_data,
    name='Ost/West',
    style_function=style_function
).add_to(m)

m
```

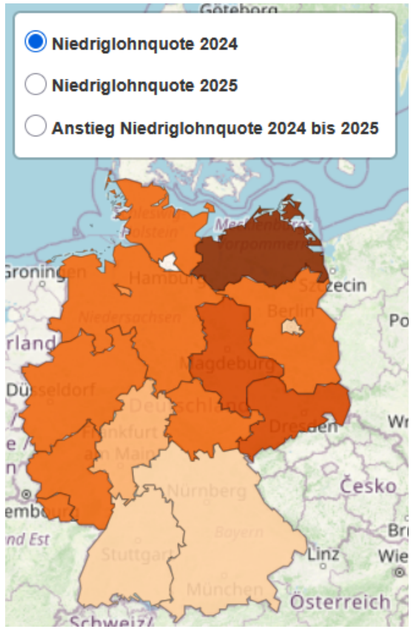
Zeitliche Entwicklung des Pay Gap

```python
import pandas as pd
import folium

df = pd.read_csv('62361-0050.csv')

df['Wert'] = pd.to_numeric(df['Wert'].astype(str).str.replace(',', '.'), errors='coerce')

bundeslaender_liste = df['Bundesland'].unique()

# jedes Jahr ein neues Dataframe, filter nur Daten für das jeweilige Jahr.
#25
df_2025 = df[
    (df['Jahr'] == 2025) &
    (df['Wirtschaftszweig'] == 'Insgesamt')
]
#24
df_2024 = df[
    (df['Jahr'] == 2024) &
    (df['Wirtschaftszweig'] == 'Insgesamt')
]

# Listen eröffnen
ergebnisse_2025 = []
ergebnisse_2024 = []
ergebnisse_2024_2025 = []

#for-loop
for bl in bundeslaender_liste:
    df_bl = df_2025[df_2025['Bundesland'] == bl]

    anteil_prozent = df_bl[df_bl['Messgröße'] == 'Anteil abh. Beschäftigungsverhältn.mit Niedriglohn']['Wert'].values[0]
    absolut_niedrig = df_bl[df_bl['Messgröße'] == 'Abhäng. Beschäftigungsverhältnisse mit Niedriglohn']['Wert'].values[0]
    absolut_gesamt = df_bl[df_bl['Messgröße'] == 'Abhängige Beschäftigungsverhältnisse']['Wert'].values[0]

    if pd.notna(anteil_prozent) and pd.notna(absolut_niedrig) and pd.notna(absolut_gesamt):
        ergebnisse_2025.append({
            'Bundesland': bl,
            'Niedriglohn_Prozent': anteil_prozent,
            'Niedriglohn_Absolut_tsd': absolut_niedrig,
            'Gesamt_Absolut_tsd': absolut_gesamt
        })
```

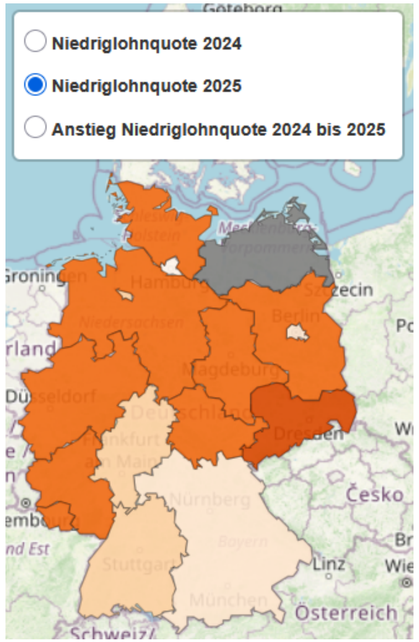
```python
for loop für 2024 wie bei kenji's 2025
r bl in bundeslaender_liste:
  df_bl = df_2024[df_2024['Bundesland'] == bl]

  anteil_prozent = df_bl[df_bl['Messgröße'] == 'Anteil abh. Beschäftigungsverhältn.mit Niedriglohn']['Wert'].values[0]
  absolut_niedrig = df_bl[df_bl['Messgröße'] == 'Abhäng. Beschäftigungsverhältnisse mit Niedriglohn']['Wert'].values[0]
  absolut_gesamt = df_bl[df_bl['Messgröße'] == 'Abhängige Beschäftigungsverhältnisse']['Wert'].values[0]

  if pd.notna(anteil_prozent) and pd.notna(absolut_niedrig) and pd.notna(absolut_gesamt):
      ergebnisse_2024.append({
          'Bundesland': bl,
          'Niedriglohn_Prozent': anteil_prozent,
          'Niedriglohn_Absolut_tsd': absolut_niedrig,
          'Gesamt_Absolut_tsd': absolut_gesamt
      })

or-loop, in neue Liste "ergebnisse_2025_2024"
r bl in bundeslaender_liste:
  df_bl_2024 = df_2024[df_2024['Bundesland'] == bl]
  df_bl_2025 = df_2025[df_2025['Bundesland'] == bl]

  anteil_prozent_2024 = df_bl_2024[df_bl_2024['Messgröße'] == 'Anteil abh. Beschäftigungsverhältn.mit Niedriglohn']['Wert'].v
  absolut_niedrig_2024 = df_bl_2024[df_bl_2024['Messgröße'] == 'Abhäng. Beschäftigungsverhältnisse mit Niedriglohn']['Wert'].
  absolut_gesamt_2024 = df_bl_2024[df_bl_2024['Messgröße'] == 'Abhängige Beschäftigungsverhältnisse']['Wert'].values[0]

  anteil_prozent_2025 = df_bl_2025[df_bl_2025['Messgröße'] == 'Anteil abh. Beschäftigungsverhältn.mit Niedriglohn']['Wert'].v
  absolut_niedrig_2025 = df_bl_2025[df_bl_2025['Messgröße'] == 'Abhäng. Beschäftigungsverhältnisse mit Niedriglohn']['Wert'].
  absolut_gesamt_2025 = df_bl_2025[df_bl_2025['Messgröße'] == 'Abhängige Beschäftigungsverhältnisse']['Wert'].values[0]
```

```
# conditions sammlung
    conditions = [pd.notna(anteil_prozent_2024),
                  pd.notna(absolut_niedrig_2024),
                  pd.notna(absolut_gesamt_2024),
                  pd.notna(anteil_prozent_2025),
                  pd.notna(absolut_niedrig_2025),
                  pd.notna(absolut_gesamt_2025)
                 ]

    if all(conditions):
        ergebnisse_2024_2025.append({
            'Bundesland': bl,
            'Niedriglohn_Prozent': anteil_prozent_2025 - anteil_prozent_2024,
            'Niedriglohn_Absolut_tsd': absolut_niedrig_2025 - anteil_prozent_2024,
            'Gesamt_Absolut_tsd': absolut_gesamt_2025 - anteil_prozent_2024
        })


df_clean_2025 = pd.DataFrame(ergebnisse_2025)
df_clean_2024 = pd.DataFrame(ergebnisse_2024)
df_clean_2024_2025 = pd.DataFrame(ergebnisse_2024_2025)

# map erstellen
m = folium.Map(location=[51.163409, 10.447718], zoom_start=5.5, tiles=None, zoom_control=False)


# Hintergrund Ebene
folium.TileLayer(
    tiles="OpenStreetMap",
    name="Background",
    overlay=True,
    control=False,
    show=True
).add_to(m)
```

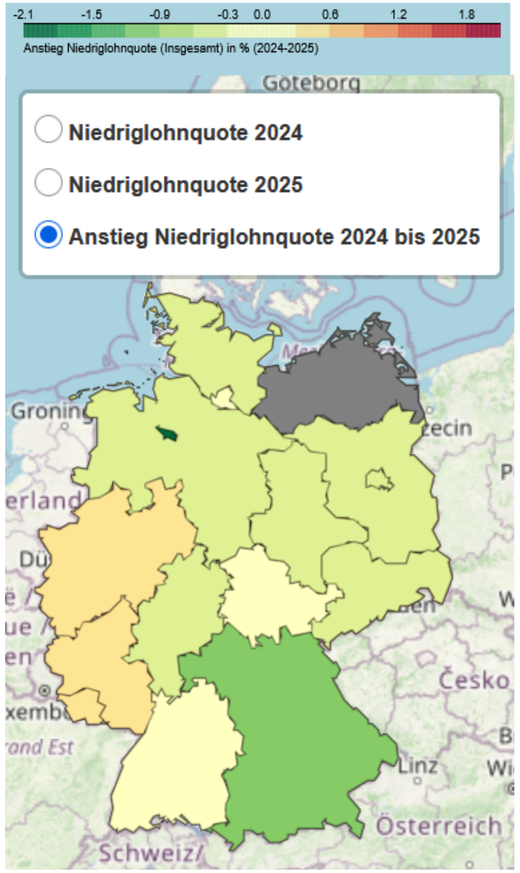
```
4
lium.Choropleth(
  name = "Niedriglohnquote 2024",
  overlay = False,
  geo_data='bundeslaender.json',
  data=df_clean_2024,
  columns=['Bundesland', 'Niedriglohn_Prozent'],
  key_on='feature.properties.name',
  fill_color='Oranges',
  fill_opacity=0.9,
  line_opacity=0.3,
  nan_fill_color='grey',
  bins=[12,13,14,15,16,17,18,19,20],
  legend_name='Niedriglohnquote (Insgesamt) in % (2024)'
add_to(m)
```

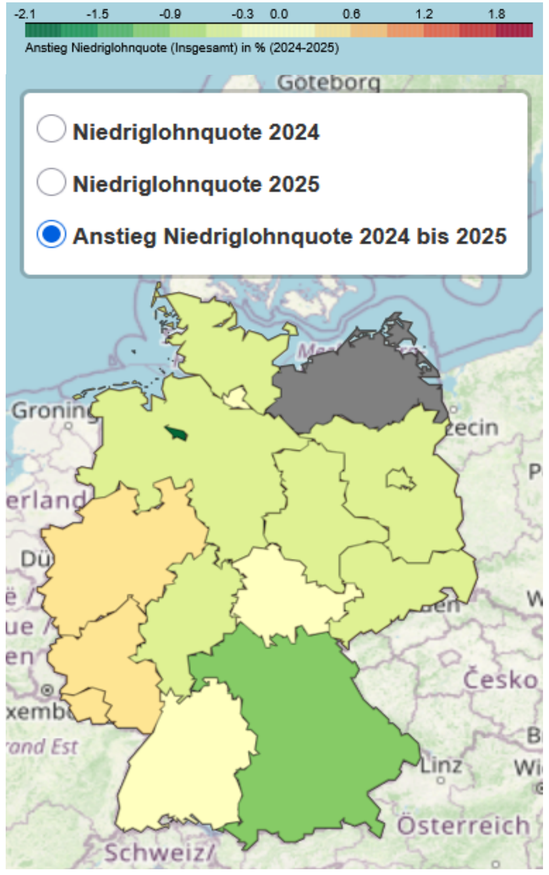


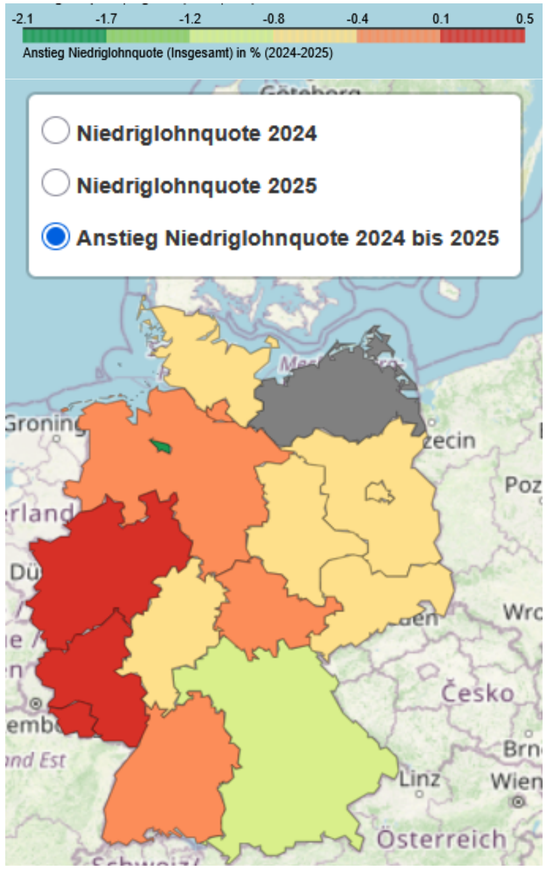

```
#25
folium.Choropleth(
    name = "Niedriglohnquote 2025",
    overlay = False,
    geo_data='bundeslaender.json',
    data=df_clean_2025,
    columns=['Bundesland', 'Niedriglohn_Prozent'],
    key_on='feature.properties.name',
    fill_color='Oranges',
    fill_opacity=0.9,
    line_opacity=0.3,
    bins=[12,13,14,15,16,17,18,19,20],
    nan_fill_color='grey',
    legend_name='Niedriglohnquote (Insgesamt) in % (2025)
).add_to(m)
```



```
der Vergleich
lium.Choropleth(
  name = "Anstieg Niedriglohnquote 2024 bis 2025",
  overlay = False,
  geo_data='bundeslaender.json',
  data=df_clean_2024_2025,
  columns=['Bundesland', 'Niedriglohn_Prozent'],
  key_on='feature.properties.name',
  fill_color='RdYlGn_r',
  fill_opacity=1,
  line_opacity=0.3,
  nan_fill_color='grey',
  bins=[-2.1,-1.8,-1.5,-1.2,-0.9,-0.6,-0.3,0.3,0.6,0.9,1.2,1.5,1.8,2.1],
  legend_name='Anstieg Niedriglohnquote (Insgesamt) in % (2024-2025)'
add_to(m)

die Auswahlbox
lium.LayerControl(collapsed=False).add_to(m)
```

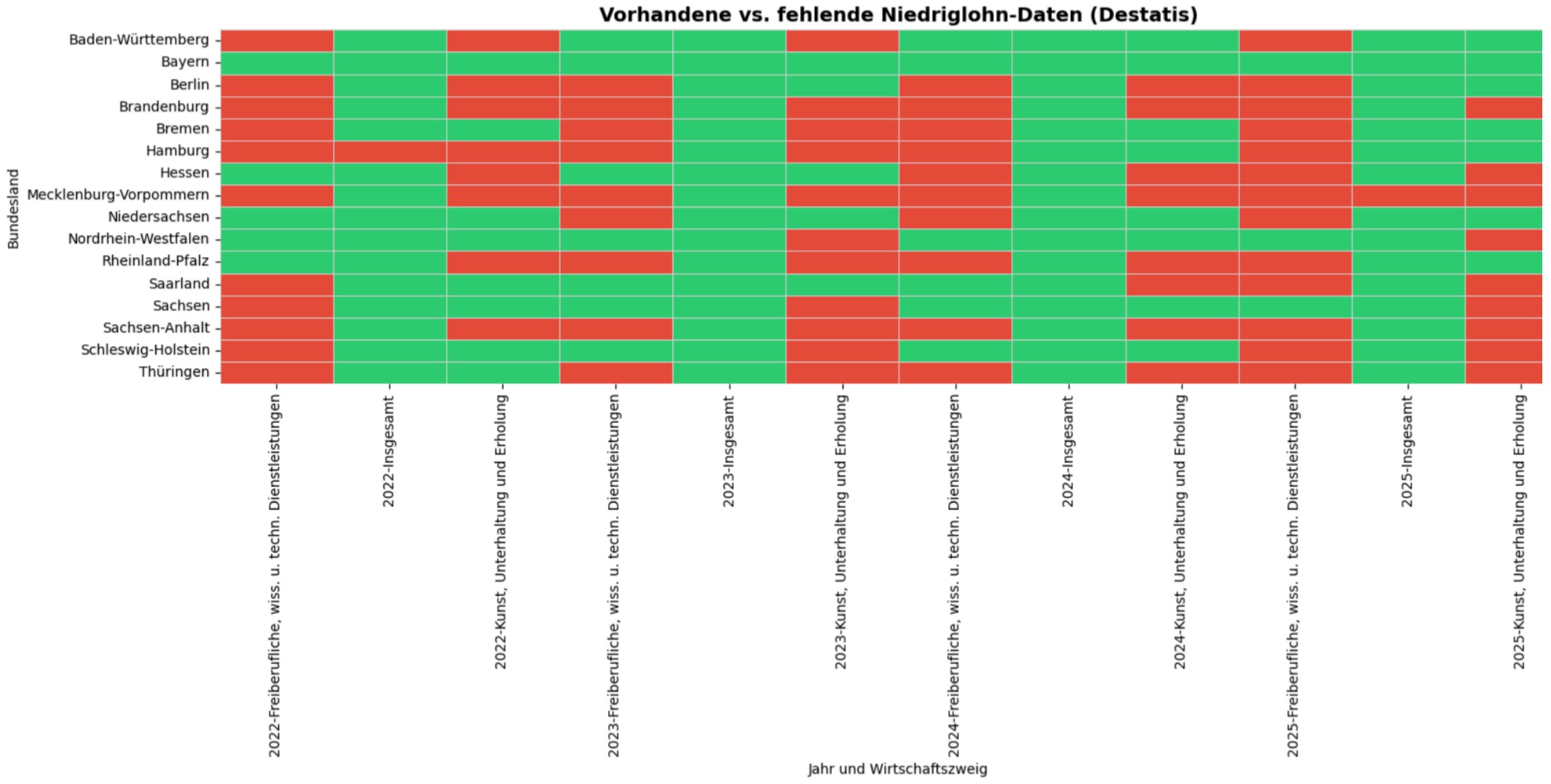



-2.1 -1.5 -0.9 -0.3 0.0 0.6 1.2 1.8
Anstieg Niedriglohnquote (Insgesamt) in % (2024-2025)
Niedriglohnquote 2024
Niedriglohnquote 2025
Anstieg Niedriglohnquote 2024 bis 2025
Česko
Österreich
Linz


-2.1 -1.7 -1.2 -0.8 -0.4 0.1 0.5
Anstieg Niedriglohnquote (Insgesamt) in % (2024-2025)
Niedriglohnquote 2024
Niedriglohnquote 2025
Anstieg Niedriglohnquote 2024 bis 2025
Česko
Österreich
Linz

Heatmap

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap


df = pd.read_csv('62361-0050.csv')


df['Wert'] = pd.to_numeric(df['Wert'].astype(str).str.replace(',', '.'), errors='coerce')


df_fehlend = df[
    df['Messgröße'] == 'Anteil abh. Beschäftigungsverhältn.mit Niedriglohn'
]

# Pivot-Matrix
matrix = df_fehlend.pivot_table(
    index='Bundesland',
    columns=['Jahr', 'Wirtschaftszweig'],
    values='Wert',
    dropna=False
)

matrix = df_fehlend.pivot_table(
    index='Bundesland',
    columns=['Jahr', 'Wirtschaftszweig'],
    values='Wert',
    dropna=False
)


# Fehlende Werte (True = fehlt, False = vorhanden)
missing = matrix.isna()


missing_int = missing.astype(int)


cmap = ListedColormap(["#2ecc71", "#e74c3c"])
# Index 0 = grün (False / vorhanden)
# Index 1 = rot (True / fehlt)


plt.figure(figsize=(16, 8))

sns.heatmap(
    missing_int,
    cmap=cmap,
    cbar=False,
    linewidths=0.5,
    linecolor='lightgrey'
)

plt.title('Vorhandene vs. fehlende Niedriglohn-Daten', fontsize=14, weight='bold')
plt.xlabel('Jahr und Wirtschaftszweig')
plt.ylabel('Bundesland')

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


Vorhandene vs. fehlende Niedriglohn-Daten (Destatis)
Bundesland
Baden-Württemberg
Bayern
Berlin
Brandenburg
Bremen
Hamburg
Hessen
Mecklenburg-Vorpommern
Niedersachsen
Nordrhein-Westfalen
Rheinland-Pfalz
Saarland
Sachsen
Sachsen-Anhalt
Schleswig-Holstein
Thüringen
2022-Freiberufliche, wiss. u. techn. Dienstleistungen
2022-Insgesamt
2022-Kunst, Unterhaltung und Erholung
2023-Freiberufliche, wiss. u. techn. Dienstleistungen
2023-Insgesamt
2023-Kunst, Unterhaltung und Erholung
2024-Freiberufliche, wiss. u. techn. Dienstleistungen
2024-Insgesamt
2024-Kunst, Unterhaltung und Erholung
2025-Freiberufliche, wiss. u. techn. Dienstleistungen
2025-Insgesamt
2025-Kunst, Unterhaltung und Erholung
Jahr und Wirtschaftszweig

Danke für Ihre Aufmerksamkeit