DS-Probeklausur-Bearbeitung

July 23, 2024

1 Probeklausur "Einführung in Data Science"

1.0.1 FH Aachen - University of Applied Sciences

Professor Dr. Stephan Bialonski

1.1 Name und Matrikelnummer

- Ihr Vor- und Nachname: (bitte ausfüllen)
- Ihre Matrikelnummer: (bitte ausfüllen)

Beachten Sie: **Die Weitergabe dieser Probeklausur an Dritte ist untersagt.**

1.2 Klausurbearbeitung. Dies müssen Sie tun:

1. **Namen und Matrikelnummer eintragen.** Tragen Sie oben Ihren Namen sowie Ihre Matrikelnummer ein, indem Sie die Platzhalter "(bitte ausfüllen)" mit Ihren Angaben ersetzen.
2. **Notebook umbenennen.** Benennen Sie dieses Jupyter Notebook nach folgendem Schema um: `Nachname-Vorname-Matrikelnummer`, wobei Sie für die Platzhalter entsprechend Ihren Namen und Ihre Matrikelnummer eintragen. Sie können das Notebook umbenennen, indem Sie auf `File>Rename` klicken und dann den neuen Namen des Jupyter Notebooks eintragen, oder indem Sie oben auf den alten Namen dieses Notebooks klicken und dort den neuen Namen eintragen. Speichern Sie Ihr Notebook nach der Umbenennung (Shortkey `Strg+s` oder auf das Speichern-Symbol (Diskette) oben klicken).
3. **Klausur bearbeiten.** *Die Bearbeitungszeit beträgt 75 Minuten.* Achten Sie auf die Bearbeitungszeit und stellen Sie sich ggf. einen Countdown-Timer oder Wecker. **Planen Sie für die Klausurabgabe 5 Minuten Zeit ein. Die Klausurabgabe beginnt nach Ende der Bearbeitungszeit.** Sie finden direkt unter diesem Satz eine Anleitung ("Klausurabgabe. Das müssen Sie tun."), wie Sie Ihre Klausur abgeben werden.

1.3 Klausurabgabe. Dies müssen Sie tun:

1. **Bearbeitungen abspeichern.** Das Wichtigste zuerst: Speichern Sie Ihre Bearbeitungen (Shortkey `Strg+s` oder auf das Speichern-Symbol (Diskette) klicken).
2. **Jupyter Notebook Datei abspeichern.** Speichern Sie Ihre Jupyter Notebook als Datei (im Folgenden "Klausur" genannt) lokal auf Ihrem Rechner ab. (`File > Download as > Notebook (.ipynb)`).

3. **Sie haben eine Probeklausur bearbeitet. Daher geben Sie Ihre Klausur jetzt nicht ab.** Die Probeklausur wird im Verlauf der Veranstaltung innerhalb Ihrer Teams besprochen. Jedes Team erstellt eine finale Klausurversion, die sie mitsamt der erarbeiteten Übungen wie üblich als Übungsabgabe einreicht. In der echten Klausur würden Sie Ihre Klausur direkt bei ILIAS einreichen.

1.4 Klausuraufgaben

1.4.1 Hinweise

(1) **Überblick verschaffen.** Wie viele Punkte können Sie bei welchen Aufgaben erzielen?

(2) **Sie müssen die Klausur persönlich und ohne fremde Hilfe bearbeiten.**

(3) **Dies ist eine Kofferklausur**. Sie dürfen erlaubte Hilfsmittel nutzen.

(4) Führen Sie die unten stehende Codezelle aus, um die Größe der von Ihnen zu erstellenden Abbildungen zu konfigurieren.

[1]:
```
import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams["figure.figsize"] = [10, 7]
```

1.4.2 Aufgabe: Corona-Pandemie

Erreichbare Punktzahl: 40 Im Frühjahr 2020 begann sich das neuartiges Coronavirus SARS-CoV-2, welches die Lungenerkrankung Covid-19 verursachen kann, in Deutschland und Europa schlagartig zu verbreiten. In dieser Aufgabe analysieren und visualisieren Sie das Infektionsgeschehen in Deutschland.

Ihre Daten

Ein vereinfachter Datensatz des Robert Koch Instituts (RKI) steht Ihnen in dieser Aufgabe zur Verfügung.

Ihre Aufgaben

(1) Führen Sie zunächst die unten stehende Code-Zelle aus, um den Datensatz und die Pandas-Bibliothek verfügbar zu machen.

[2]:
```
import pandas as pd

# Daten des Robert Koch Instituts
RKI_data_simplified = 'RKI_data_simplified.csv'
```

(2) Importieren Sie mithilfe von Pandas den Datensatz in einen DataFrame, den Sie `df_covid` nennen. Nutzen Sie dazu die Variable `RKI_data_simplified`.[/2] Punkte

[3]:
```
# Ihr Code
df_covid: pd.DataFrame = pd.read_csv(RKI_data_simplified)
df_covid.head()
```

[3]:

	Meldedatum	Deutschland	Schleswig-Holstein	Hamburg	Niedersachsen
0	2020-01-28	2	NaN	NaN	NaN
1	2020-01-29	2	NaN	NaN	NaN
2	2020-01-31	3	NaN	NaN	NaN
3	2020-02-03	1	NaN	NaN	NaN
4	2020-02-04	4	NaN	NaN	1.0

	Bremen	Nordrhein-Westfalen	Hessen	Rheinland-Pfalz	Baden-Württemberg
0	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN

	Bayern	Saarland	Berlin	Brandenburg	Mecklenburg-Vorpommern	Sachsen
0	2.0	NaN	NaN	NaN	NaN	NaN
1	2.0	NaN	NaN	NaN	NaN	NaN
2	3.0	NaN	NaN	NaN	NaN	NaN
3	1.0	NaN	NaN	NaN	NaN	NaN
4	3.0	NaN	NaN	NaN	NaN	NaN

	Sachsen-Anhalt	Thüringen
0	NaN	NaN
1	NaN	NaN
2	NaN	NaN
3	NaN	NaN
4	NaN	NaN

(3) Der DataFrame, den Sie erhalten haben, enthält die Anzahl neu infizierter Personen für `Deutschland` sowie aufgelistet nach Bundesland, die zu einem Zeitpunkt (`Meldedatum`) gemeldet wurden. [/4] Punkte

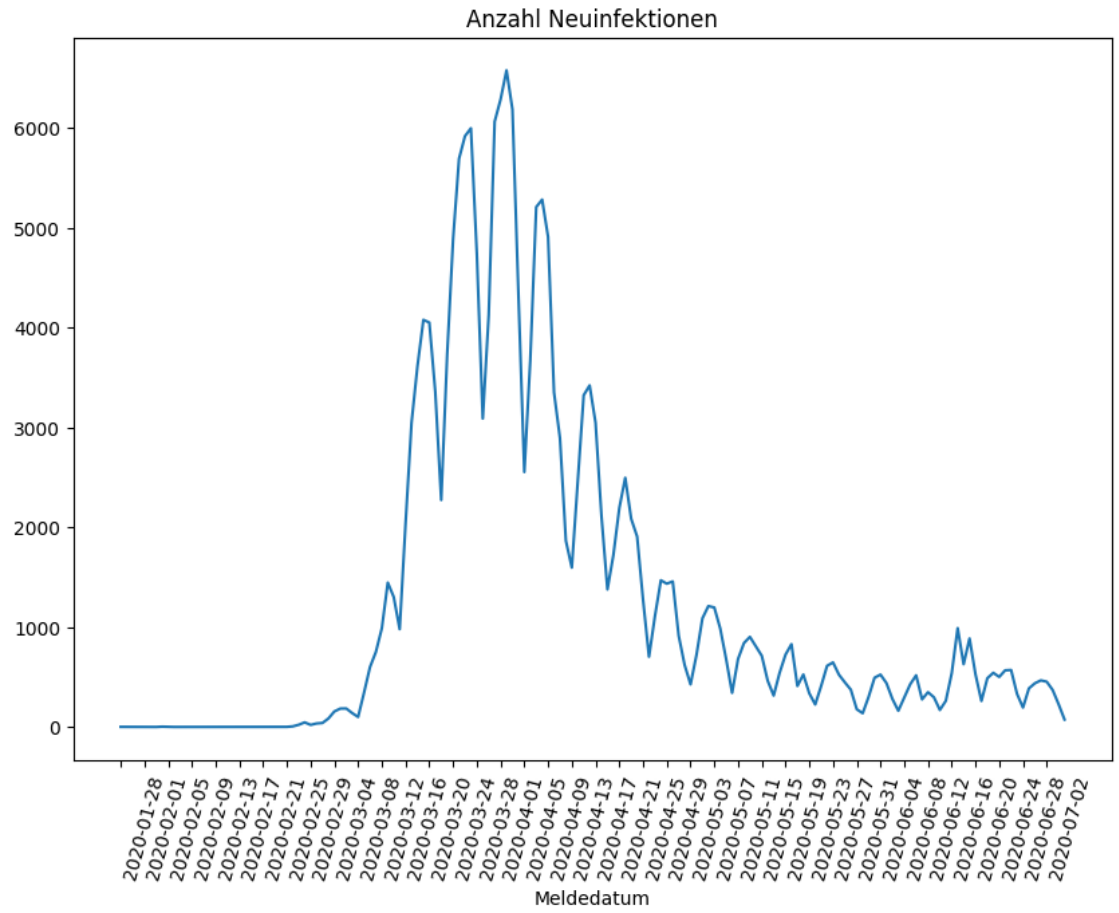
- Wandeln Sie die Spalte `Meldedatum` in einen *datetime* Typen um. Sie können die Pandas Funktion `to_datetime` dafür nutzen. (2 Punkte)
- Machen Sie anschließend `Meldedatum` zum Index Ihres DataFrames. Sie können die Pandas Funktion `set_index` dafür nutzen. (2 Punkte)

[4]:
```
# Ihr Code
df_covid["Meldedatum"] = pd.to_datetime(df_covid["Meldedatum"])
df_covid.set_index("Meldedatum", inplace=True)
```

(4) Visualisieren Sie die Anzahl neu infizierter Personen für Deutschland (Spalte: Deutschland) als Funktion des Meldedatums. Ihre Abbildung muss folgende Eigenschaften erfüllen: [/6] Punkte

- Die x-Achse ist mit “Meldedatum” beschriftet. (2 Punkte)
- Auf der x-Achse stehen Datumsangaben (z.B: ‘2020-03-17’), keine bloßen Integers. (2 Punkte)
- Der Titel der Abbildung lautet “Anzahl Neuinfektionen”. (2 Punkte)

[5]:
```
# Ihr Code
plt.xlabel("Meldedatum")
plt.title("Anzahl Neuinfektionen")
plt.plot(df_covid["Deutschland"])

xticks = pd.date_range(start=df_covid.index.min(), end=df_covid.index.max(),␣
 ↪freq='4d')
plt.xticks(xticks, xticks.strftime("%Y-%m-%d"), rotation=75, ha="left")

plt.show()
```



(5) Am 2. April 2020 begann die *Einführung in Data Science* Vorlesung in Aachen. *Bestimmen Sie* die Anzahl gemeldeter Neuinfizierter an diesem Tag und *nennen Sie diese Anzahl.* [/4] Punkte

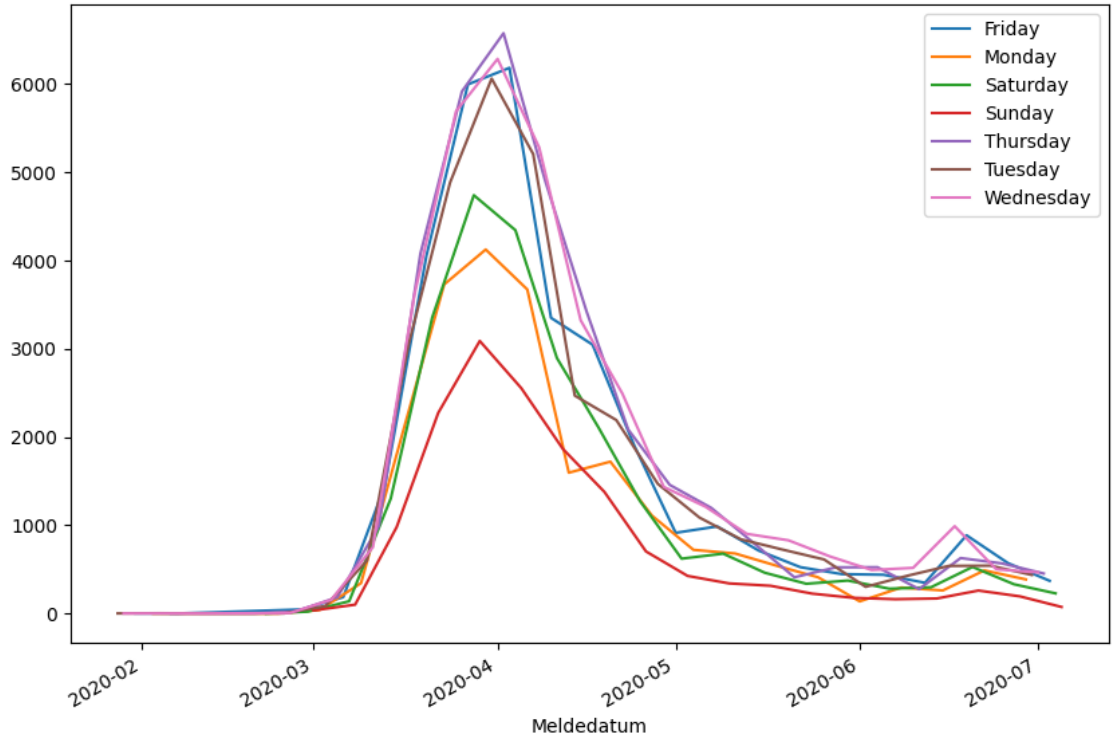
[6]:
```
# Ihr Code
df_covid.loc["2020-04-02"]["Deutschland"]
```

[6]:
```
np.float64(6574.0)
```

6574 gemeldete Neuinfektionen an dem Tag

(6) Die Anzahl gemeldeter Neuinfizierter in Ihrer Visualisierung aus Schritt (4) zeigt periodische Schwingungen. Wie erklären Sie das Auftreten dieser Schwingungen? (Stichpunkte oder 1-3 Sätze) [/6] Punkte

[7]:
```
df_covid["Weekday"] = df_covid.index.day_name()
df_covid.groupby("Weekday")["Deutschland"].plot()
plt.legend()
plt.show()
```



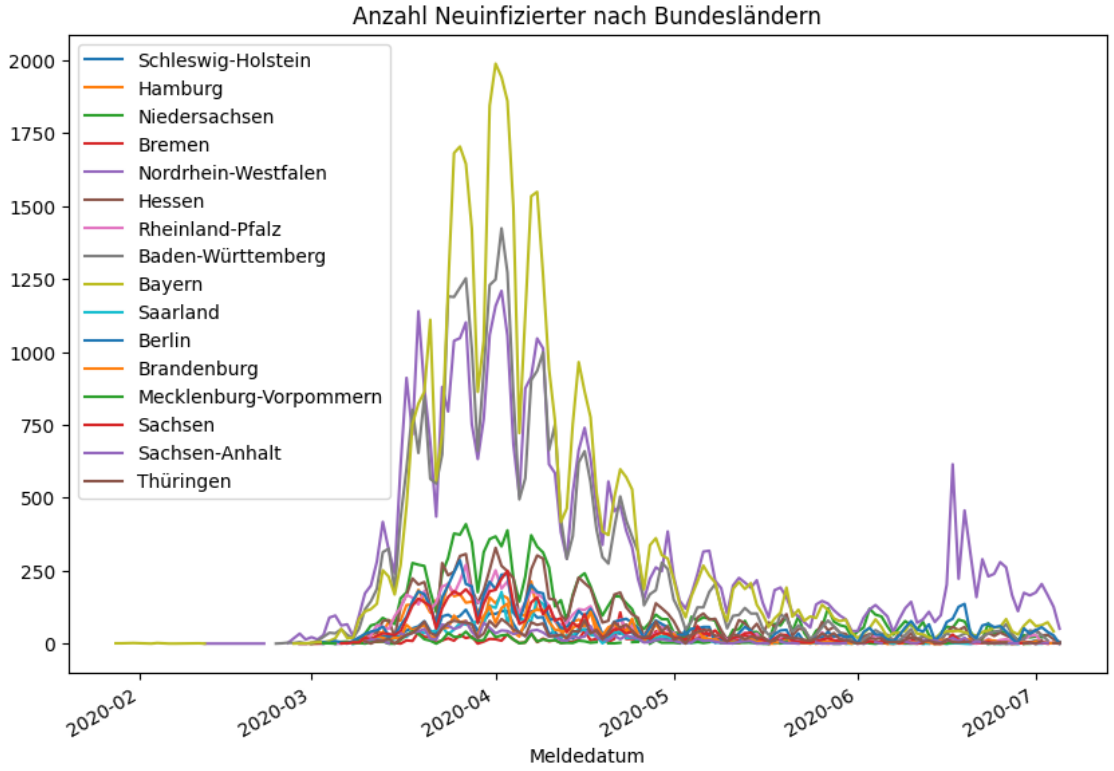
Die Schwankungen entstehen durch den Wochenverlauf. Sonntags werden nur sehr wenige Infektionen gemeldet, gefolgt von Montagen und Samstagen.

(7) Die Anzahl Neuinfizierter steigt kurzzeitig im Juni 2020 etwas an und flacht dann wieder ab. Untersuchen Sie, auf welches Bundesland sich dieser Anstieg zurückführen lässt: [/8] Punkte

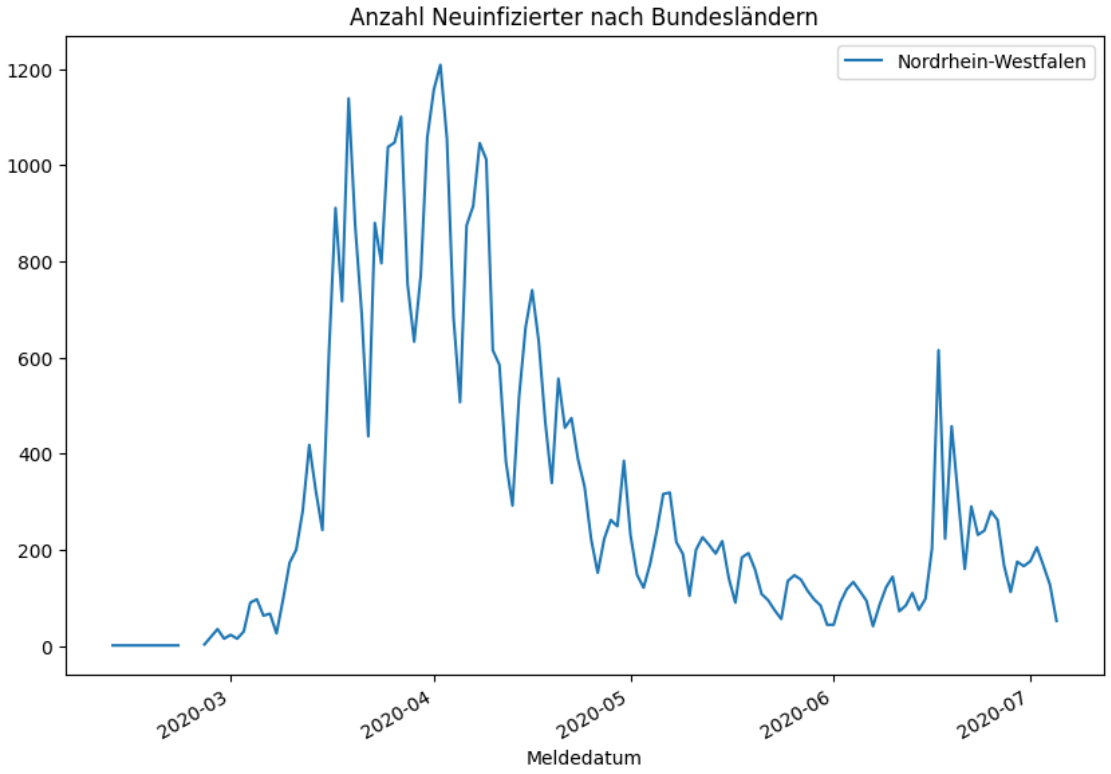
1. Visualisieren Sie die Anzahl gemeldeter Neuinfizierter als Funktion des Meldedatums *für alle Bundesländer* in derselben Abbildung. Ihre Abbildung erfüllt die folgenden Eigenschaften:
   - Die x-Achse ist mit "Meldedatum" beschriftet. (1 Punkt)
   - Auf der x-Achse stehen Datumsangaben (z.B: '2020-03-17'), keine bloßen Integers. (1 Punkt)
   - Der Titel der Abbildung lautet "Anzahl Neuinfizierter nach Bundesländern". (1 Punkt)
   - Eine Legende erlaubt eine farbkodierte Unterscheidung der dargestellten Graphen nach Bundesländern. (1 Punkt)

- Ihre Abbildung enthält **nicht** die Gesamtzahl aller Infizierter Deutschlands.

[8]:
```
# Ihr Code
df_covid.iloc[:, 1:].plot()
plt.xlabel("Meldedatum")
plt.title("Anzahl Neuinfizierter nach Bundesländern")
plt.legend()
plt.show()
```



[9]:
```
df_covid.loc[:, "Nordrhein-Westfalen"].plot()
plt.xlabel("Meldedatum")
plt.title("Anzahl Neuinfizierter nach Bundesländern")
plt.legend()
plt.show()
```



2. Identifizieren Sie über Ihre Abbildung das Bundesland, auf das sich der kurzzeitige Anstieg der Neuinfizierten im Juni 2020 zurückführen lässt, und nennen Sie dieses Bundesland. (2 Punkte)

   - Bonus: Benennen Sie die Ursache dieser erhöhten Infektionszahlen. (+2 Punkte)

Nordrhein-Westfalen. In dem Kreis Gütersloh kam es im Juni 2020 zu dem bisher „größten Infektionsgeschehen“ in Deutschland. Corona-Ausbruch bei Tönnies – Lockdown in Gütersloh. In: tagesschau.de. ARD, 23. Juni 2020, abgerufen am 23. Juni 2020.

3. In welchem Bundesland Deutschlands wurden als erstes Corona-Infizierte gemeldet? (2 Punkte)

[10]:
```
# Ihr Code
df_covid.sort_index()
df_covid
```

[10]:

Meldedatum	Deutschland	Schleswig-Holstein	Hamburg	Niedersachsen	Bremen	\
2020-01-28	2	NaN	NaN	NaN	NaN	
2020-01-29	2	NaN	NaN	NaN	NaN	
2020-01-31	3	NaN	NaN	NaN	NaN	
2020-02-03	1	NaN	NaN	NaN	NaN	
2020-02-04	4	NaN	NaN	1.0	NaN	

```
…                  …                    …       …              …      …
2020-07-01           467                 12.0     9.0           25.0    4.0
2020-07-02           454                  5.0     2.0           22.0    4.0
2020-07-03           371                  2.0     2.0           30.0    NaN
2020-07-04           229                  2.0     NaN            NaN    3.0
2020-07-05            75                  6.0     NaN            NaN    1.0

             Nordrhein-Westfalen  Hessen  Rheinland-Pfalz  Baden-Württemberg  \
Meldedatum
2020-01-28                   NaN     NaN              NaN                NaN
2020-01-29                   NaN     NaN              NaN                NaN
2020-01-31                   NaN     NaN              NaN                NaN
2020-02-03                   NaN     NaN              NaN                NaN
2020-02-04                   NaN     NaN              NaN                NaN
…                              …       …                …                  …
2020-07-01                 176.0    49.0             30.0               42.0
2020-07-02                 205.0    29.0              8.0               30.0
2020-07-03                 167.0    31.0              6.0               12.0
2020-07-04                 127.0    16.0              2.0                NaN
2020-07-05                  52.0     7.0              NaN                NaN

             Bayern  Saarland  Berlin  Brandenburg  Mecklenburg-Vorpommern  \
Meldedatum
2020-01-28      2.0       NaN     NaN          NaN                     NaN
2020-01-29      2.0       NaN     NaN          NaN                     NaN
2020-01-31      3.0       NaN     NaN          NaN                     NaN
2020-02-03      1.0       NaN     NaN          NaN                     NaN
2020-02-04      3.0       NaN     NaN          NaN                     NaN
…               …         …       …            …                       …
2020-07-01     61.0       1.0    47.0          8.0                     NaN
2020-07-02     64.0       1.0    57.0          8.0                     1.0
2020-07-03     74.0       NaN    41.0          2.0                     NaN
2020-07-04     43.0       1.0    29.0          3.0                     NaN
2020-07-05      NaN       NaN     1.0          NaN                     NaN

             Sachsen  Sachsen-Anhalt  Thüringen    Weekday
Meldedatum
2020-01-28       NaN             NaN        NaN    Tuesday
2020-01-29       NaN             NaN        NaN  Wednesday
2020-01-31       NaN             NaN        NaN     Friday
2020-02-03       NaN             NaN        NaN     Monday
2020-02-04       NaN             NaN        NaN    Tuesday
…                  …               …          …         …
2020-07-01       1.0             1.0        1.0  Wednesday
2020-07-02       4.0             4.0       10.0   Thursday
2020-07-03       1.0             3.0        NaN     Friday
2020-07-04       NaN             NaN        3.0   Saturday
```

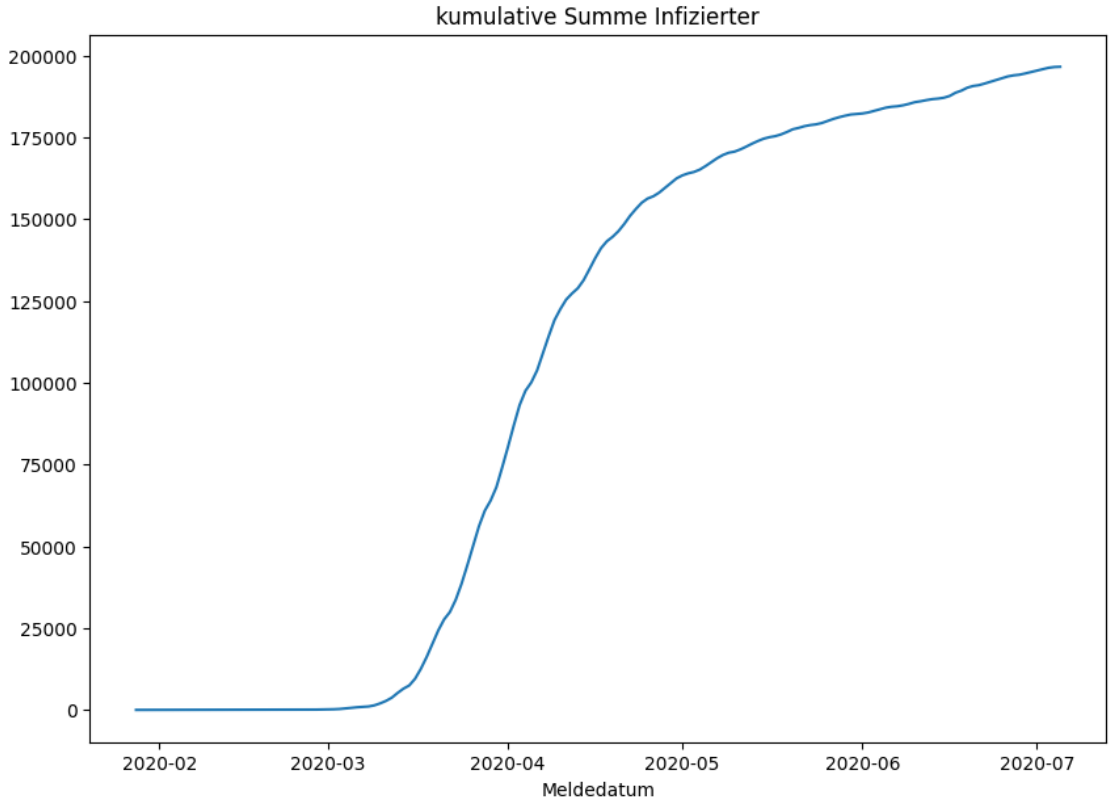
```
2020-07-05       NaN              6.0         2.0      Sunday

[144 rows x 18 columns]
```

Bayern

(8) Visualisieren Sie die kumulative Summe der Neuinfizierten in Deutschland (Spalte: Deutschland) als Funktion des Meldedatums. [/4] Punkte

- Die x-Achse ist mit “Meldedatum” beschriftet. (1 Punkte)
- Auf der x-Achse stehen Datumsangaben (z.B: ‘2020-03-17’), keine bloßen Integers. (1 Punkte)
- Der Titel der Abbildung lautet “kumulative Summe Infizierter”. (1 Punkte)

[11]:
```
# Ihr Code
plt.xlabel("Meldedatum")
plt.title("kumulative Summe Infizierter")
plt.plot(df_covid["Deutschland"].cumsum())
plt.show()
```



(9) Bestimmen Sie die Zahl aller Menschen, die in Deutschland bis zum 5. Juli 2020 als mit Covid-19 infiziert gemeldet wurden. [/6] Punkte

[12]:
```
# Ihr Code
df_covid["Deutschland"].cumsum().loc["2020-07-05"]
```

[12]:
```
np.int64(196550)
```

196.550 Personen wurden bis zum 5. Juli 2020 infiziert.

(10) *Bonus:* (Bearbeiten Sie diese Teilaufgabe am besten erst, wenn Sie bereits die anderen Aufgaben dieser Klausur bearbeitet haben.)

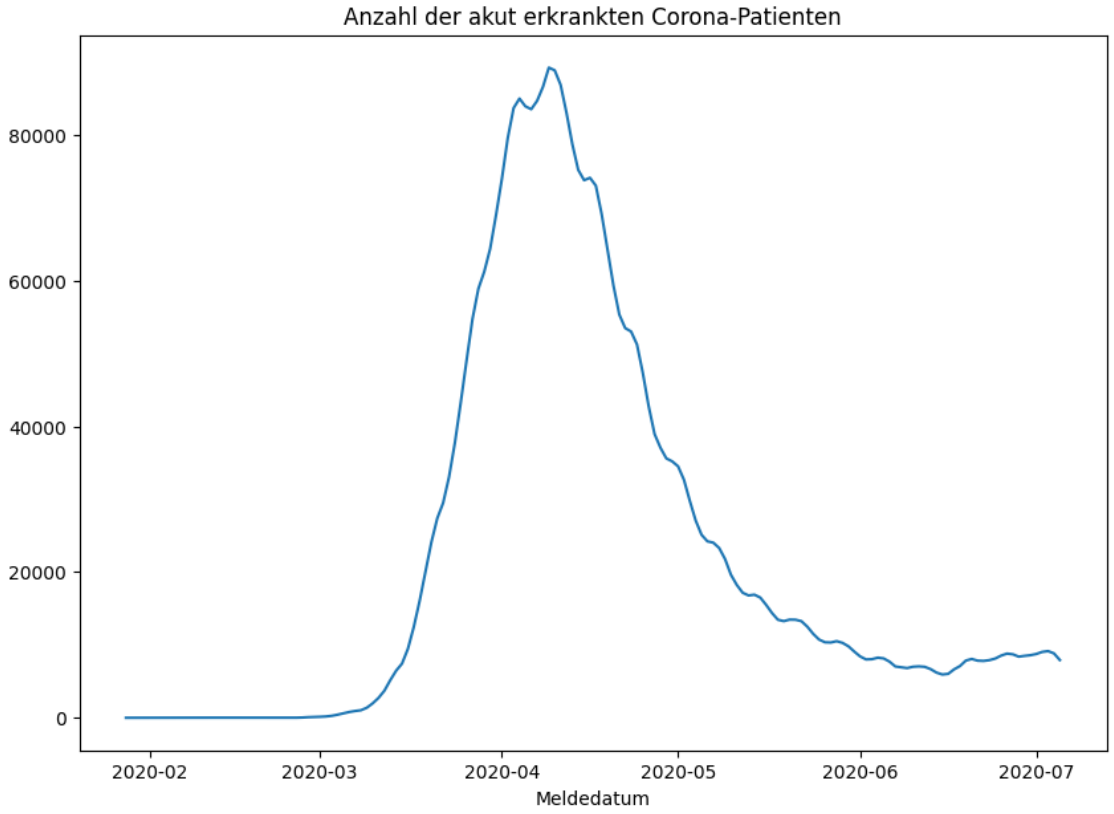
Visualisieren Sie die geschätzte Anzahl der akut erkrankten Coronapatienten ganz Deutschlands als Funktion des Meldedatums. [/8] Punkte

Für die Schätzung der akut erkrankten Patienten (5 Punkte) nehmen wir Folgendes an:

- Wir nehmen an, dass die Infektion 18 Tage nach dem Meldedatum vorbei ist und der Patient gesund geworden ist.
- Wir nehmen an, dass keine Menschen durch eine Corona-Infektion versterben.

Ihre Visualisierung erfüllt die folgenden Eigenschaften: - Die x-Achse ist mit "Meldedatum" beschriftet. (1 Punkt) - Auf der x-Achse stehen Datumsangaben (z.B: '2020-03-17'), keine bloßen Integers. (1 Punkt) - Der Titel der Abbildung lautet "Anzahl der akut erkrankten Corona-Patienten". (1 Punkt)

[13]:
```
# Ihr Code
plt.xlabel("Meldedatum")
plt.title("Anzahl der akut erkrankten Corona-Patienten")
plt.plot(df_covid.rolling('18D')["Deutschland"].sum())
plt.show()
```



1.4.3 Aufgabe: Reproduktionszahl

Erreichbare Punktzahl: 28 Sie bestimmen in dieser Aufgabe die Reproduktionszahl R für die Covid-19 Pandemie in Deutschland anhand der Daten des Robert-Koch-Instituts. Die Reproduktionszahl R gibt an, wie viele Menschen von einer infektiösen Person durchschnittlich angesteckt werden, wenn kein Mitglied der Population gegenüber dem Erreger immun ist.

Sie werden eine vereinfachte Schätzung der R-Zahl durchführen. Die R-Zahl eines Tages t sei definiert als

$$R(t) := \frac{\sum_{x=t-3}^{t} I_x}{\sum_{x=t-7}^{t-4} I_x},$$

wobei I_x die Anzahl der Neuinfizierten eines Tages x bezeichnet.

Die R-Zahl entspricht also der Summe der Infizierten innerhalb eines 4-Tageszeitraums dividiert durch die Summe der Infizierten des davorliegenden 4-Tageszeitraums.

Ihre Daten

Ein vereinfachter Datensatz des Robert Koch Instituts (RKI) steht Ihnen in dieser Aufgabe zur Verfügung.

Ihre Aufgaben

(1) Führen Sie zunächst die unten stehende Code-Zelle aus, um den Datensatz und die Pandas-Bibliothek verfügbar zu machen.

[14]:
```
import pandas as pd

# Daten des Robert Koch Instituts
RKI_data_simplified = 'RKI_data_simplified.csv'
```

(2) Importieren Sie mithilfe der Variablen `RKI_data_simplified` den Datensatz in einen Pandas DataFrame, den Sie `df_covid` nennen. [/4] Punkte

- Wandeln Sie die Spalte `Meldedatum` in einen *datetime* Typen um. Sie können die Pandas Funktion `to_datetime` dafür nutzen. (2 Punkte)
- Machen Sie anschließend `Meldedatum` zum Index Ihres DataFrames. Sie können die Pandas Funktion `set_index` dafür nutzen. (2 Punkte)

[15]:
```
# Ihr Code
df_covid: pd.DataFrame = pd.read_csv(RKI_data_simplified)
df_covid["Meldedatum"] = pd.to_datetime(df_covid["Meldedatum"])
df_covid.set_index("Meldedatum", inplace=True)
```

(3) Bestimmen Sie nun die $R(t)$-Zahl für Deutschland (Spalte: Deutschland) aus den Daten, die Sie im vorherigen Schritt verfügbar gemacht haben. [/12] Punkte

Gehen Sie dabei am besten in zwei Schritten vor:

1. Bestimmen Sie die rollierenden 4-Tagessummen der Infiziertenzahlen. Sie erhalten dadurch einen neuen DataFrame, der eine Zeitreihe enthält, deren Einträge jeweils einer Summe über die Anzahl der Infizierten über 4 aufeinanderfolgende Tage darstellt. Nutzen Sie für diesen Schritt am besten die Funktionalität von Pandas. (6 Punkte)
   - **Wichtig**: Die Summen in der Definition von $R(t)$ (siehe obige Gleichung) verlaufen jeweils über 4 direkt aufeinanderfolgenden Tagen.
2. Bestimmen Sie $R(t)$ gemäß der Definition zu Beginn dieser Aufgabe. Nutzen Sie aus, dass Sie die Zeitreihe (der rollierenden 4-Tagessummen) um 4 Tage mithilfe von Pandas verschieben können. (6 Punkte)

Wir führen einen Shift $r \to r_{\text{shft}}$ mit `r_shft = r.shift(4, axis=0)` durch, da

$$R(5) = \frac{r(5)}{r_{\text{shft}}(5)} = \frac{r(5)}{r(1)}$$

sein soll. Aus dieser Betrachtung geht hervor, warum der frühere Zeitraum (Nenner) verschoben werden muss und nicht der spätere.

[16]:
```
df_covid['date'] = df_covid.index

r1: pd.DataFrame = df_covid.rolling('4D', on='date')["Deutschland"].sum()
```

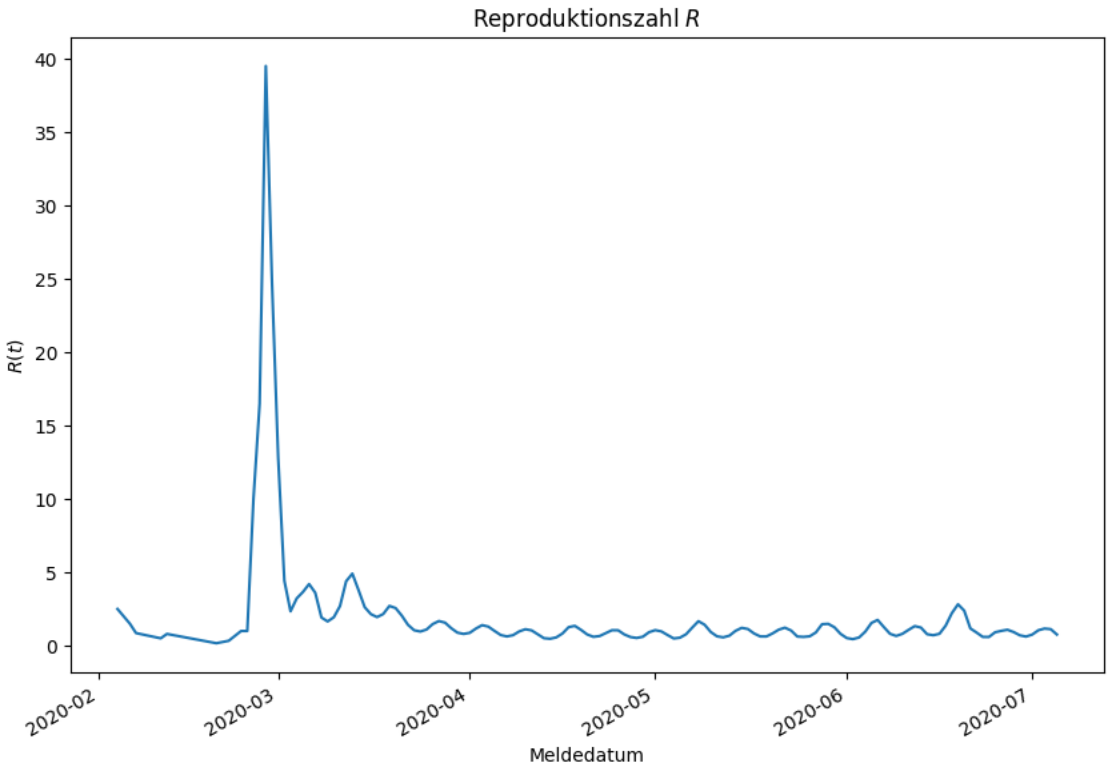
```
# Nenner in die Zukunft shiften bedeutet, dass bei index=5 r(5) = r1(5) /␣
  ↪r1_shifted(5) und da r1_shifted(5) = r1(1) ist, passt das
r_t: pd.DataFrame = r1 / r1.shift(4, axis=0)
r_t.head(10)
```

[16]:
```
Meldedatum
2020-01-28         NaN
2020-01-29         NaN
2020-01-31         NaN
2020-02-03         NaN
2020-02-04    2.500000
2020-02-06    1.500000
2020-02-07    0.857143
2020-02-11    0.500000
2020-02-12    0.800000
2020-02-20    0.166667
Name: Deutschland, dtype: float64
```

(4) Visualisieren Sie die $R(t)$-Zahl als Funktion des Meldedatums t. [/4] Punkte

- Sie haben $R(t)$ als Funktion des Meldedatums in einer Abbildung visualisiert. (1 Punkt)
- Die x-Achse ist mit "Meldedatum" beschriftet. (1 Punkt)
- Auf der x-Achse stehen Datumsangaben (z.B: '2020-03-17' oder 'Apr 2020'), keine bloßen Integers. (1 Punkt)
- Der Titel der Abbildung lautet "Reproduktionszahl R". (1 Punkt)

[17]:
```
# Ihr Code
r_t.plot()
plt.xlabel("Meldedatum")
plt.ylabel("$R(t)$")
plt.title("Reproduktionszahl $R$")
plt.show()
```



(5) An welchen Tagen im Juni 2020 können Sie Reproduktionszahlen beobachten, die größer sind als 2? [/8] Punkte

- Angabe der Tage im Juni 2020, deren Reproduktionszahlen größer sind als 2. (6 Punkte)
- Welcher Ausbruch steht mit den erhöhten Reproduktionszahlen im Juni im Zusammenhang stehen? (1-2 Sätze oder Stichworte) (2 Punkte)

[18]:
```
# Ihr Code
r_t_juni = r_t.loc["2020-06-01":"2020-06-30"]
r_t_juni[r_t_juni > 2]
```

[18]:
```
Meldedatum
2020-06-18    2.210959
2020-06-19    2.820370
2020-06-20    2.387097
Name: Deutschland, dtype: float64
```

18., 19. und am 20. Juni 2020

Outbreaks of COVID-19 have been reported in several federal states (including in institutions for asylum seekers and refugees, in connection with a religious events or in meat processing plants).

1.4.4 Aufgabe: Corona-Warn-App Analyse

Erreichbare Punktzahl: 20 Die Corona-Warn-App wurde im Juni 2020 in Deutschland veröffentlicht, um die Kontaktnachverfolgung infizierter Personen zu erleichtern. Die App wurde bis Anfang Juli 2020 bereits über 14 Millionen Mal heruntergeladen.

Über die App-Infrastruktur werden jeden Tag Menschen vor potentiellen Infektionen gewarnt. Aus den dabei benötigten täglichen Daten lassen sich die Anzahl der Menschen ermitteln, die pro Tag über die Warn-App als infiziert gemeldet wurden.

Sie werden in dieser Aufgabe untersuchen, wieviel Prozent aller neuinfizierten Fälle (laut Robert-Koch-Institut) bereits über die Corona-Warn-App gemeldet werden.

Ihre Daten

- Ein vereinfachter Datensatz des Robert Koch Instituts (RKI) steht Ihnen in dieser Aufgabe zur Verfügung.
- Corona-Warn-App Datensatz, der die Schätzung der Anzahl der als infiziert gemeldeten Menschen enthält

Ihre Aufgaben

(1) Führen Sie zunächst die unten stehende Code-Zelle aus, um die Datensätze und die Pandas-Bibliothek verfügbar zu machen.

[19]:
```
import pandas as pd

# Daten des Robert Koch Instituts
RKI_data_simplified = 'RKI_data_simplified.csv'

# Daten der Corona Warn App Infrastruktur
cwa_data = 'cwa_data.csv'
```

(2) Importieren Sie mithilfe der Variablen `RKI_data_simplified` und `cwa_data` die Datensätze in Pandas DataFrames, die Sie `df_covid` bzw. `df_cwa` nennen. Für beide DataFrames machen Sie Folgendes: [/4] Punkte

- Wandeln Sie die Spalte `Meldedatum` in einen *datetime* Typen um. Sie können die Pandas Funktion `to_datetime` dafür nutzen. (2 Punkte)
- Machen Sie anschließend `Meldedatum` zum Index Ihres jeweiligen DataFrames. Sie können die Pandas Funktion `set_index` dafür nutzen. (2 Punkte)

[20]:
```
# Ihr Code
df_covid: pd.DataFrame = pd.read_csv(RKI_data_simplified)
df_cwa: pd.DataFrame = pd.read_csv(cwa_data)

df_covid["Meldedatum"] = pd.to_datetime(df_covid["Meldedatum"])
df_covid.set_index("Meldedatum", inplace=True)

df_cwa["Meldedatum"] = pd.to_datetime(df_cwa["Meldedatum"])
df_cwa.set_index("Meldedatum", inplace=True)
```

```
df_cwa.head()
```

[20]:
```
            AnzahlFall
Meldedatum
2020-06-22           0
2020-06-23          27
2020-06-24          21
2020-06-25          19
2020-06-26          23
```

(3) Bestimmen Sie tagesgenau den prozentualen Anteil der Neuinfizierten, die durch die Corona-Warn-App gemeldet wurden, an der gesamten Anzahl der Neuinfizierten (Spalte "Deutschland"), die durch das Robert-Koch-Institut gemeldet wurden. [/6] Punkte

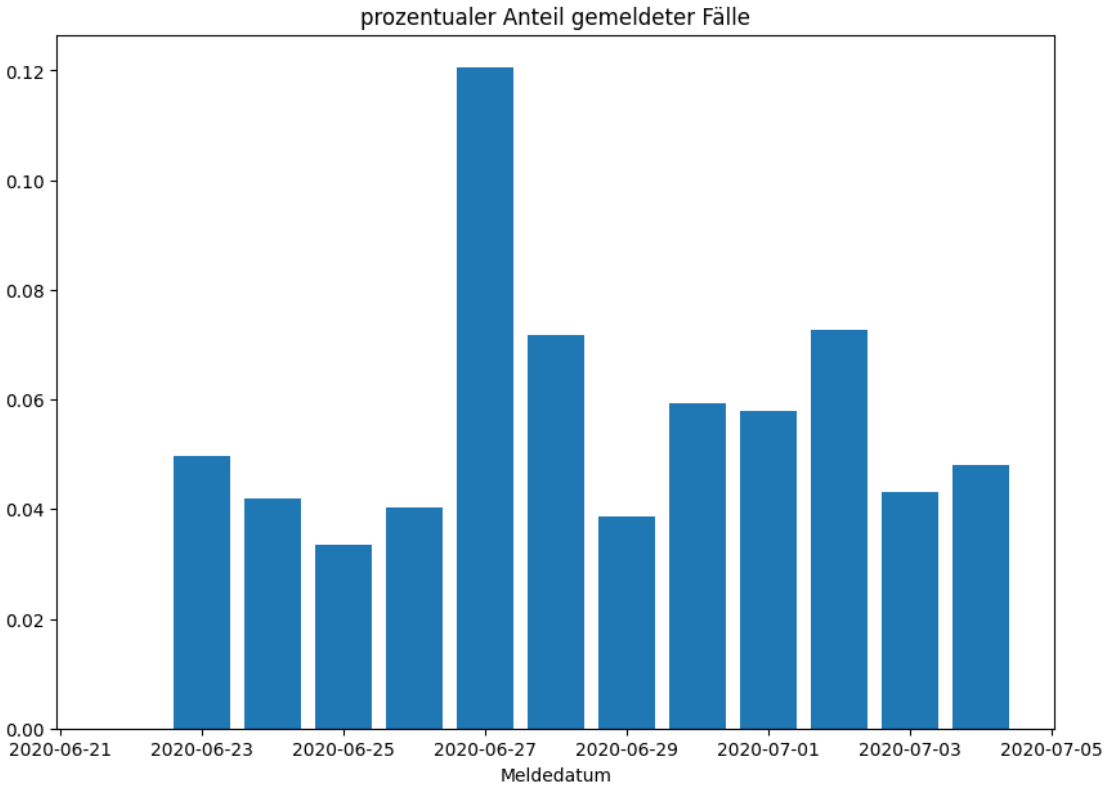
[21]:
```
# Ihr Code
df: pd.DataFrame = df_covid.join(df_cwa, on="Meldedatum")
prozentualAnteil = df["AnzahlFall"] / df["Deutschland"]
prozentualAnteil.dropna()
```

[21]:
```
Meldedatum
2020-06-22    0.000000
2020-06-23    0.049724
2020-06-24    0.041833
2020-06-25    0.033451
2020-06-26    0.040280
2020-06-27    0.120482
2020-06-28    0.071795
2020-06-29    0.038760
2020-06-30    0.059361
2020-07-01    0.057816
2020-07-02    0.072687
2020-07-03    0.043127
2020-07-04    0.048035
dtype: float64
```

(4) Visualisieren Sie in einem Bar-Plot den prozentualen Anteil, den Sie in Schritt (3) ermittelt haben, als Funktion des Meldedatums. [/5] Punkte

- Die x-Achse ist mit "Meldedatum" beschriftet. (1 Punkte)
- Auf der x-Achse stehen Datumsangaben (z.B: '2020-03-17' oder 'Apr 2020'), keine bloßen Integers. (2 Punkte)
- Der Titel der Abbildung lautet "prozentualer Anteil gemeldeter Fälle". (1 Punkte)

[22]:
```
# Ihr Code
plt.bar(prozentualAnteil.index, prozentualAnteil)
plt.xlabel("Meldedatum")
plt.title("prozentualer Anteil gemeldeter Fälle")
plt.show()
```



(5) Wieviel Prozent aller Neuinfizierten vom 2. Juli wurden durch die Corona-Warn-App gemeldet? Ermitteln Sie diese Prozentzahl und nennen Sie sie. [/5] Punkte

[23]:
```
# Ihr Code
prozentualAnteil.loc["2020-07-02"]
```

[23]:
```
np.float64(0.07268722466960352)
```

7,27 %

1.4.5 Aufgabe: Werbeindustrie

Erreichbare Punktzahl: 12 Der folgende Datenfall ist echt und wurde für die Klausurstellung anonymisiert:

Ein Data Scientist einer Firma, die im Bereich Onlinewerbung tätig ist, fragt um Rat. Die Firma möchte anhand von Beobachtungen verschiedene Personengruppen identifizieren. Es liegen Daten von $N = 700000$ Personen mit je $D = 100000$ Merkmalen (Features) vor.

Der Data Scientist hat eine explorative Datenanalyse (EDA) wie folgt vorgenommen:

1. Die Dimension des Feature-Raums wurde mithilfe einer PCA (Principal Component Analysis) von 100000 auf $D' = 3500$ Dimensionen reduziert.

2. Um eine Intuition über die in diesem dimensionsreduzierten Feature-Raum etwaig vorhandenen Cluster zu erhalten, wurden mit einem nichtlinearen Verfahren des Multidimensionalen Skalierens (MDS) die Daten in zwei Dimensionen dargestellt. Folgende Abbildung ergab sich:

Arbeiten Sie unter folgenden Annahmen:

- Das MDS-Verfahren funktioniert wahrheitsgetreu.
- Die PCA wurde korrekt implementiert.

Ihre Aufgaben

(1) Untersuchen Sie die oben dargestellte Analysekette des Data Scientist: Welche Aspekte müssen Sie hinterfragen bzw. kritisieren? Notieren Sie stichwortartig Ihre Fragen bzw. Kritikpunkte. (Bei zwei Kritikpunkten bzw. Fragen sind Sie schon gut unterwegs). [/6] Punkte

- Wie wurde die Anzahl der Komponenten gewählt, auf die der Featureraum über die PCA reduziert wurde? Wurde hier ein Ellbogenverlauf in der Proportion of Variance Explained (PVE) zu Rate gezogen?
- Warum meint der Data Scientist, dass sich der Raum durch eine lineare Transformation dimensionsreduzieren lässt? Wäre es nicht angebracht, auch nach nichtlinearen Strukturen zu suchen?

(2) Der Data Scientist hat die in der obigen Abbildung erkennbaren Clusterstrukturen untersucht. Der rot eingekreiste Bereich markiert einen Cluster, der tatsächlich einer Personengruppe entspricht, wie er herausgefunden hat. Solche Cluster möchte er über ein geeignetes Clusterverfahren aus dem Feature-Raum extrahieren. [/6] Punkte

- Nennen Sie genau ein Clusterverfahren aus der Vorlesung *Data Science*, welches Sie ihm vorschlagen. (4 Punkte)

K-Means

- Begründen Sie Ihre Entscheidung für das von Ihnen vorgeschlagene Clusterverfahren und grenzen Sie es zu alternativen Verfahren ab. (1-5 Sätze) (2 Punkte)

Ich schlage K-Means vor, da hier nach der Vorverarbeitung ein sphärische Objekte im Feature-Space entstanden ist, welches über die euklidische Distanz zum Clusterschwerpunkt gut identifizierbar sein sollte. Ein hierarchisches Clustern könnte dann in Frage kommen, wenn wir hierarchisch organisierte Daten vorleigen haben. Dies könnte nach Art der Daten der Fall sein und das auftreten zusammenhängender Stränge würde auch dafür sprechen, jedoch ist der marktierte Bereich spährisch.