Aufgabe_1

April 21, 2021

1 1.1 Modellierung der Wärmenachfrage eines Fernwärmenetzes mittel linearer Regression

[1]:
```
%matplotlib inline
import pandas as pd
import numpy as np
from pathlib import Path
import matplotlib.pyplot as plt
from scipy import stats, optimize
from IPython.display import display, Latex
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,  mean_squared_error
import seaborn as sns
```

[2]:
```
demand_path = Path("__file__").parent.resolve()
demand_path = demand_path / "ENTSOE_countries" / "Demand"
demand_filename = "AT_2019.csv"
waermenachfrage_filename = "Waermenachfrage_Uebung1.xlsx"
file = demand_path / demand_filename

df_demand = pd.read_csv(file)
df_waermenachfrage = pd.read_excel(waermenachfrage_filename, engine = openpyxl)
```

[3]:
```
df_waermenachfrage
```

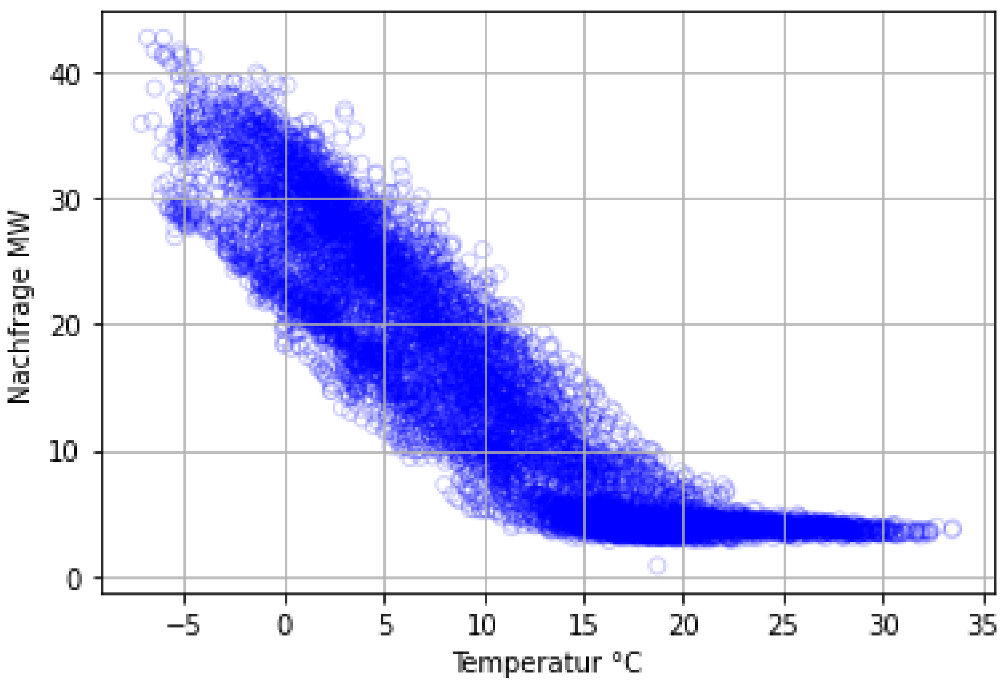
[3]:
```
                     Zeit  Monat  Tag  Stunde     Temp  Nachfrage  Feiertage  \
0     2007-10-01 01:00:01     10    1       1  11.6288   7.909745          0
1     2007-10-01 02:00:01     10    1       2  11.3327   8.046135          0
2     2007-10-01 03:00:01     10    1       3  10.8108   8.104064          0
3     2007-10-01 04:00:01     10    1       4  10.3233   8.795061          0
4     2007-10-01 05:00:01     10    1       5   9.9904  10.192541          0
...                   ...    ...  ...     ...      ...        ...        ...
8779 2008-09-30 20:00:01      9   30      20  15.4673  12.406460          0
8780 2008-09-30 21:00:01      9   30      21  14.9188  12.693875          0
8781 2008-09-30 22:00:01      9   30      22  14.3348  12.210588          0
8782 2008-09-30 23:00:01      9   30      23  13.9884   9.258651          0
8783 2008-10-01 00:00:01     10    1       0  13.2498   8.798517          0
```

```
      Wochentag
0             2
1             2
2             2
3             2
4             2
...         ...
8779          3
8780          3
8781          3
8782          3
8783          4

[8784 rows x 8 columns]
```

[4]:
```
nachfrage = df_waermenachfrage.loc[:, "Nachfrage"].to_numpy()
temperatur = df_waermenachfrage.loc[:, "Temp"].to_numpy()
```

[5]:
```
plt.scatter(temperatur, nachfrage, alpha=0.2, facecolors=none, edgecolors=b)
plt.xlabel("Temperatur řC")
plt.ylabel("Nachfrage MW")
plt.grid()
```

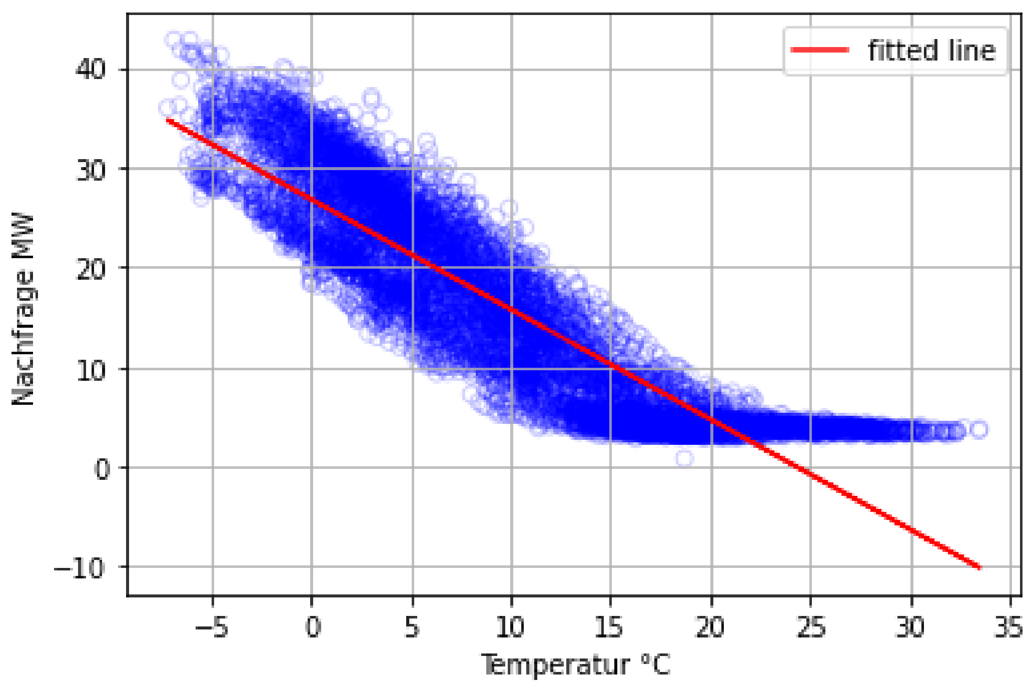


Einfache Regression:

$$y_t = b_0 + b_1 * T_0$$

[6]:
```
regression = stats.linregress(temperatur, nachfrage)
display(Latex(f"R-squared: {regression.rvalue**2:.6f}"))
display(Latex(f"P-value: {regression.pvalue:.6f}"))
display(Latex(f"$b_{0}$: {regression.intercept:.6f}"))
display(Latex(f"$b_{1}$: {regression.slope:.6f}"))
display(Latex(f"$sqrt(Var(b_{1}))$: {regression.stderr:.6f}"))
display(Latex(f"$sqrt(Var(b_{0}))$: {regression.intercept_stderr:.6f}"))
plt.scatter(temperatur, nachfrage, alpha=0.2, facecolors=none, edgecolors=b)
plt.xlabel("Temperatur řC")
plt.ylabel("Nachfrage MW")
plt.plot(temperatur, regression.intercept + regression.slope*temperatur, r,␣
 ↪label=fitted line)
plt.legend()
plt.grid()
```

R-squared: 0.809534
P-value: 0.000000
b_0: 26.818651
b_1: -1.102223
$sqrt(Var(b_1))$: 0.005705
$sqrt(Var(b_0))$: 0.077027



c) Liegt der P-Wert unter 0.05, so ist der Koeffizient einer Variable im Modell signifikant. Da der P-Wert bei uns = 0 ist, bedeuted das, dass die Temperatur einen signifikanten Einfluss auf die Nachfrage hat.

Der t-Wert ergibt sich aus

$$t(b_i) = \frac{b_i - \beta}{Var(b_i)} \tag{1}$$

β ist der Schätzer der Nullhypothese H_0 und entspricht dem Wert 0. Somit berechnet sich der $t - Wert$ von b_1 zu -193.2 und von b_2 zu 348.2.Da der t-Wert weit entfernt vom Schätzer der Nullhyphotese ist, kann diese verworfen werden.

R^2 ist der Determinationskoeffizient. Er gibt an wie sehr die Varianz der abhängigen Variable durch die erklärende Variable erklärt wird. Werte für R^2 liegen immer zwischen 0 und 1. Umso näher der Wert bei 1 liegt, umso besser nähert sich das Model den real gemessenen Werten an. In unserem Fall, mit $R^2 = 81% $ bedeuted es, dass die Variable "Temperatur" 81% der Nachfragewerte erklärt.

Nullhypothese: Die Nullhypothese kann nicht verifiziert sondern nur falsifizert werden. Wenn der P-Wert sehr "klein" ist, kann man die Nullhypothese ablehnen. In unserem Fall ist die Nullhypothese: "Die Nachfrage ist nicht abhängig von der Temperatur." Insgesamt gibt es 8785 Messwerte. Da die Freiheitsgrade einer Statistik, immer die Anzahl der Messpunkte minus den zu bestimmenden Parametern entspricht, so haben wir in diesem Modell 8783 Freiheitsgrade.

[ ]:

```

```

Vergeich zu sklearn:

[7]:

```
# transforming the data to include another axis
#x = temperatur[:, np.newaxis]
#y = nachfrage[:, np.newaxis]

#model = LinearRegression()
#model.fit(x, y)
#y_pred = model.predict(x)



#plt.scatter(x, y, alpha=0.2, facecolors=none, edgecolors=b)
#plt.plot(x, y_pred, color=r, label=fitted line)
#plt.xlabel("Temperatur řC")
#plt.ylabel("Nachfrage MW")
#plt.legend()
#plt.grid()
```

2 Modell 3

Regression mit Polynom 3.ten Grades:

$$y_t = b_0 + b_1 * T_1 + b_2 * h_t + b_3 * h_t^2 + b_4 * h_t^3$$

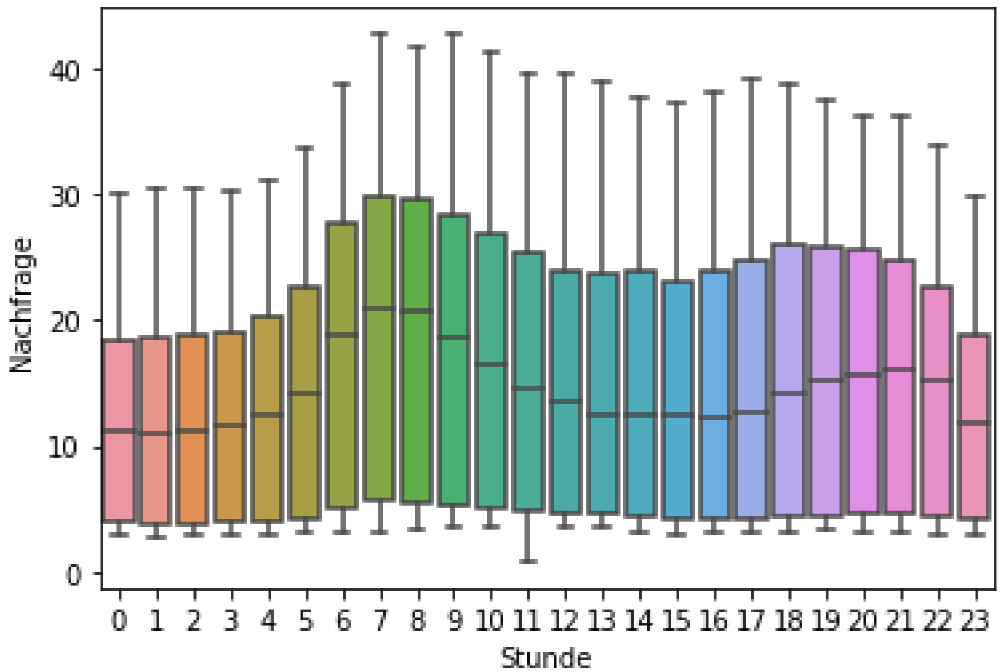
[8]:

```
h_t = df_waermenachfrage.loc[:, "Stunde"]
nachfrage = df_waermenachfrage.loc[:, "Nachfrage"]
data_1 = pd.concat([h_t, nachfrage],axis=1)

sns.boxplot(x = "Stunde", y="Nachfrage", data= data_1)
```

```
#plt.xlabel("Zeit")
#plt.ylabel("Nachfrage MW")
#plt.grid()
```

[8]: <AxesSubplot:xlabel=Stunde, ylabel=Nachfrage>



[9]:
```
def function_1(data, b0, b1, b2, b3, b4):
    return b0 + b1*data[0]+ b2*data[1] + b3*data[1]**2 + b4*data[1]**3
```

[10]:
```
data = ([temperatur, h_t])
opt, cov = optimize.curve_fit(function_1, data, nachfrage)
```

[11]:
```
display(Latex(f"$b_{0}$:{opt[0]:.6f}"))
display(Latex(f"$b_{1}$:{opt[1]:.6f}"))
display(Latex(f"$b_{2}$:{opt[2]:.6f}"))
display(Latex(f"$b_{3}$:{opt[3]:.6f}"))
display(Latex(f"$b_{4}$:{opt[4]:.6f}"))
perr = np.sqrt(np.diag(cov))
```

b_0:22.374836
b_1:-1.174532
b_2:0.213884
b_3:0.076774
b_4:-0.003492
Die Standardabweichungen der einzelnen Parameter entsprechen:

[12]:
```
display(Latex(f"$\sqrt var(b_{0})$:{perr[0]:.6f}"))
display(Latex(f"$\sqrt var(b_{1})$:{perr[1]:.6f}"))
```

```
display(Latex(f"$\sqrt var(b_{2})$:{perr[2]:.6f}"))
display(Latex(f"$\sqrt var(b_{3})$:{perr[3]:.6f}"))
display(Latex(f"$\sqrt var(b_{4})$:{perr[4]:.6f}"))
```

$\sqrt var(b_{0})$:0.136118
$\sqrt var(b_{1})$:0.004561
$\sqrt var(b_{2})$:0.049616
$\sqrt var(b_{3})$:0.005107
$\sqrt var(b_{4})$:0.000146

Daraus ergeben sich wiederum die t-Werte unter Annahme der Nullhypothese H_0:

[13]:
```
t_werte = opt/perr
display(Latex(f"$t(b_{0})$:{t_werte[0]:.6f}"))
display(Latex(f"$t(b_{1})$:{t_werte[1]:.6f}"))
display(Latex(f"$t(b_{2})$:{t_werte[2]:.6f}"))
display(Latex(f"$t(b_{3})$:{t_werte[3]:.6f}"))
display(Latex(f"$t(b_{4})$:{t_werte[4]:.6f}"))
```

$t(b_{0})$:164.378464
$t(b_{1})$:-257.538297
$t(b_{2})$:4.310802
$t(b_{3})$:15.032861
$t(b_{4})$:-23.897745

Das BestimmtheitsmaSS ist ein GütemaSS der linearen Regression und dient eigentlich nicht als GütemaSS bei eines nichtlinearen Fit. Nichtsdestotrotz enspricht der R^2 Wert hier:

[15]:
```
linie = function_1(data, opt[0], opt[1], opt[2], opt[3], opt[4])
```

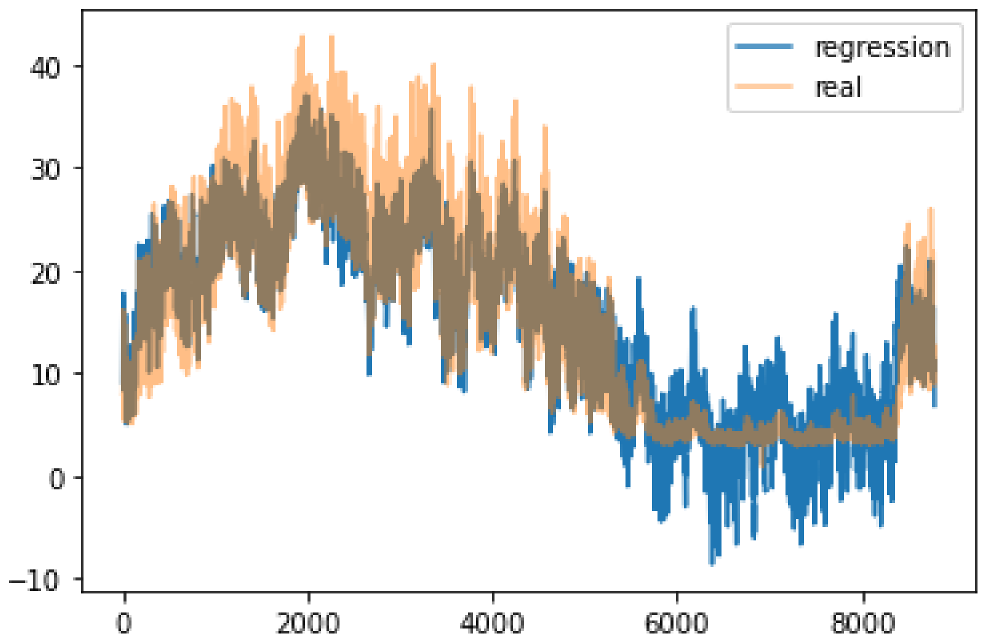
[16]:
```
R_2 = sum((linie-np.mean(nachfrage))**2)/sum((nachfrage - np.
 ↪mean(nachfrage))**2)
display(Latex(f"$R^{2}$={R_2:.6f}"))
```

R^{2}=0.885581

Somit beschreibt dieses Modell unsere Nachfrage besser als das lineare Modell.

[17]:
```
plt.plot(linie, label="regression")
plt.plot(nachfrage, alpha=0.5, label="real")
plt.legend()
```

[17]:
```
<matplotlib.legend.Legend at 0x18b5b3cbd00>
```



3 Modell 3

Die Nachfrage wird nun anhand des stündlichen Nachfrage-Verlaufs modelliert. Dabei wird für jede Stunde eine lineare Regression durchgeführt anhand:

$$y_t^j = b_0^j + b_1^j * T_0^j \tag{2}$$

wobei der index j für die jeweilige Stunde im Tagesverlauf steht: $j \subseteq [0, 23]$

[44]:
```
parameters_stunden = []
nachfrage_stunden = []

for i in range(24):
    df_waermenachfrage_stunde =␣
 ↪df_waermenachfrage[df_waermenachfrage[Stunde]==i].copy()
    nachfrage_stunden.append(df_waermenachfrage_stunde)
    temperatur = df_waermenachfrage_stunde[Temp]
    nachfrage = df_waermenachfrage_stunde[Nachfrage]
    regression = stats.linregress(temperatur, nachfrage)
    parameters_stunden.append(regression)
    print("Für Stunde %i beträgt b_0= %.6f und b_1= %.6f. Das Bestimmtheitsma␣
 ↪beträgt %.6f" % (i, regression.intercept, regression.slope, regression.
 ↪rvalue**2))
```

```
Für Stunde 0 beträgt b_0= 21.498538 und b_1= -0.994402. Das Bestimmtheitsma
beträgt 0.927203
```

```
Für Stunde 1 beträgt b_0= 21.284146 und b_1= -1.018228. Das Bestimmtheitsma
beträgt 0.924408
Für Stunde 2 beträgt b_0= 21.501871 und b_1= -1.058290. Das Bestimmtheitsma
beträgt 0.924869
Für Stunde 3 beträgt b_0= 21.810618 und b_1= -1.096958. Das Bestimmtheitsma
beträgt 0.922130
Für Stunde 4 beträgt b_0= 22.638799 und b_1= -1.160629. Das Bestimmtheitsma
beträgt 0.926320
Für Stunde 5 beträgt b_0= 24.948538 und b_1= -1.288817. Das Bestimmtheitsma
beträgt 0.925060
Für Stunde 6 beträgt b_0= 30.313237 und b_1= -1.565600. Das Bestimmtheitsma
beträgt 0.919358
Für Stunde 7 beträgt b_0= 32.513835 und b_1= -1.658584. Das Bestimmtheitsma
beträgt 0.911542
Für Stunde 8 beträgt b_0= 32.195898 und b_1= -1.589896. Das Bestimmtheitsma
beträgt 0.917352
Für Stunde 9 beträgt b_0= 31.499577 und b_1= -1.486253. Das Bestimmtheitsma
beträgt 0.922885
Für Stunde 10 beträgt b_0= 30.337651 und b_1= -1.358132. Das Bestimmtheitsma
beträgt 0.919526
Für Stunde 11 beträgt b_0= 29.442326 und b_1= -1.255967. Das Bestimmtheitsma
beträgt 0.910420
Für Stunde 12 beträgt b_0= 28.807459 und b_1= -1.172428. Das Bestimmtheitsma
beträgt 0.904219
Für Stunde 13 beträgt b_0= 28.559487 und b_1= -1.109519. Das Bestimmtheitsma
beträgt 0.896673
Für Stunde 14 beträgt b_0= 28.573331 und b_1= -1.073758. Das Bestimmtheitsma
beträgt 0.892053
Für Stunde 15 beträgt b_0= 28.753121 und b_1= -1.054321. Das Bestimmtheitsma
beträgt 0.888296
Für Stunde 16 beträgt b_0= 29.235306 und b_1= -1.058179. Das Bestimmtheitsma
beträgt 0.887333
Für Stunde 17 beträgt b_0= 29.973512 und b_1= -1.080798. Das Bestimmtheitsma
beträgt 0.888343
Für Stunde 18 beträgt b_0= 30.320551 und b_1= -1.090658. Das Bestimmtheitsma
beträgt 0.890741
Für Stunde 19 beträgt b_0= 29.756508 und b_1= -1.076983. Das Bestimmtheitsma
beträgt 0.900467
Für Stunde 20 beträgt b_0= 29.241898 und b_1= -1.095359. Das Bestimmtheitsma
beträgt 0.912258
Für Stunde 21 beträgt b_0= 28.556620 und b_1= -1.127965. Das Bestimmtheitsma
beträgt 0.924304
Für Stunde 22 beträgt b_0= 26.778458 und b_1= -1.128576. Das Bestimmtheitsma
beträgt 0.937202
Für Stunde 23 beträgt b_0= 22.009168 und b_1= -0.962979. Das Bestimmtheitsma
beträgt 0.935646
```

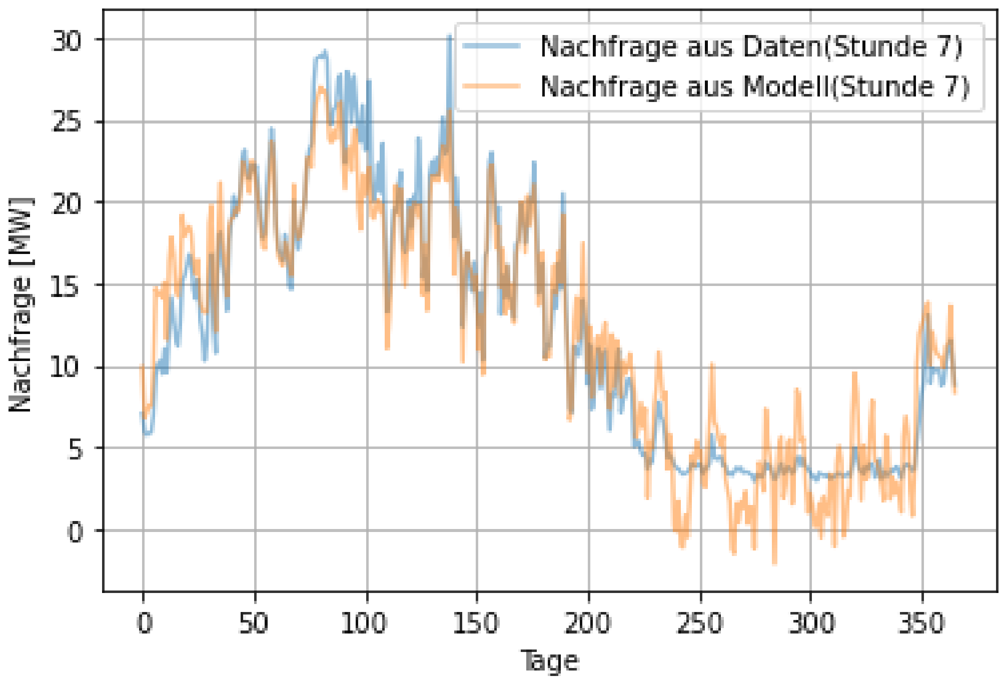
Die gröSSten Unterschiede können ausgemacht werde zwischen Stunde 0 und Stunde 7:

Koeffizient	Wert	Standardabweichung	t-Statistik
β_0^7	32.514	.283	114.89
β_1^7	-1.659	.027	-61.444
R^2	0.912		

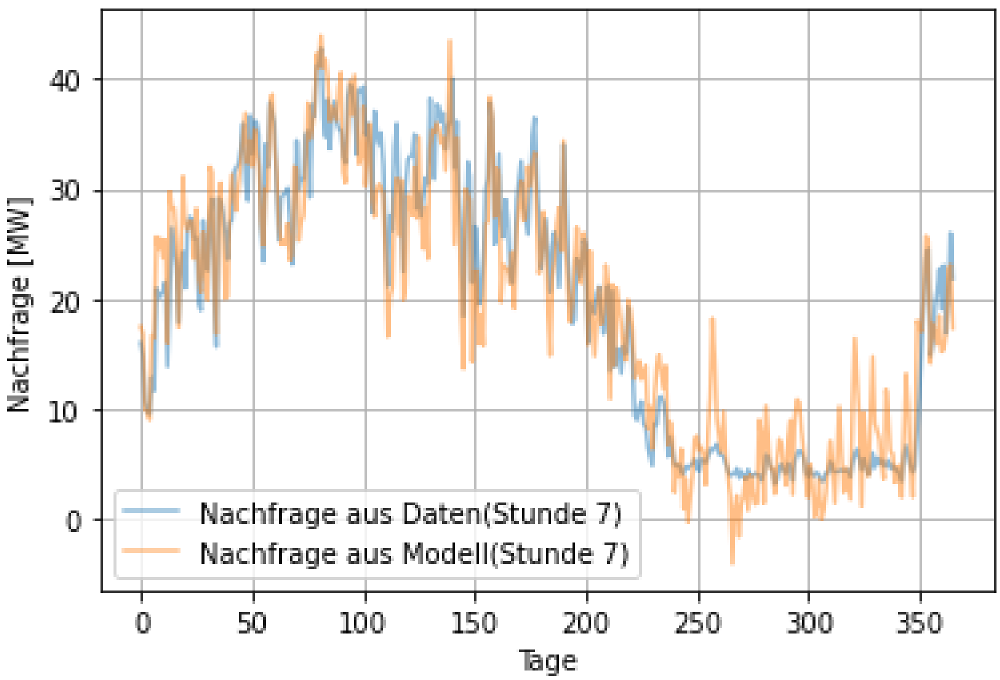
Koeffizient	Wert	Standardabweichung	t-Statistik
β_0^0	21.499	.283	75.968
β_1^0	-0.994	.027	-36.815
R^2	0.927		

Die unterschiede der beiden Kurven lassen sich dadurch erklären, dass im Jahresmittel die Nachfrage um 7 Uhr stärkeren Schwankungen ausgesetzt ist, als die Nachfrage um Mitternacht. Die BestimmtheitsmaSSe zu den jeweiligen Stunden treffen besser zu als zu den vorherigen Modellen. Besonders zu den Nachtstunden, wo der Wärmebedarf im Jahresmittel geringere Schwankungen beinhaltet, scheint die lineare Regression besser zuzutreffen, als über das ganze Jahr.

[54]:
```
nachfrage_stunden[0][Tage_rel] = np.arange(0,nachfrage_stunden[0].shape[0])
nachfrage_stunden[0][Nachfrage_modell] = 21.499 + (-0.994 *␣
↪nachfrage_stunden[0][Temp])
x = nachfrage_stunden[0][Tage_rel].to_numpy()
y_1 = nachfrage_stunden[0][Nachfrage].to_numpy()
y_2 = nachfrage_stunden[0][Nachfrage_modell].to_numpy()
plt.plot(x , y_1, alpha=0.5, label = Nachfrage aus Daten(Stunde 7))
plt.plot(x , y_2, alpha=0.5, label = Nachfrage aus Modell(Stunde 7))
plt.xlabel("Tage")
plt.ylabel("Nachfrage [MW]")
plt.legend()
plt.grid()
```



[59]:
```
nachfrage_stunden[7][Tage_rel] = np.arange(0,nachfrage_stunden[7].shape[0])
nachfrage_stunden[7][Nachfrage_modell] = 32.514 + (-1.659 *␣
↪nachfrage_stunden[7][Temp])
x = nachfrage_stunden[7][Tage_rel].to_numpy()
y_1 = nachfrage_stunden[7][Nachfrage].to_numpy()
y_2 = nachfrage_stunden[7][Nachfrage_modell].to_numpy()
plt.plot(x , y_1, alpha=0.5, label = Nachfrage aus Daten(Stunde 7))
plt.plot(x , y_2, alpha=0.5, label = Nachfrage aus Modell(Stunde 7))
plt.xlabel("Tage")
plt.ylabel("Nachfrage [MW]")
plt.legend()
plt.grid()
```



3.1 Interpretation

Die beiden vorherigen Abbildungen zeigen, dass die Modelle sowohl zur Stunde 7 als auch zur S-tunde 0 in den Wintermonaten sehr gut zutreffen. In den Sommermonaten kommt es jedoch trotz relativ stabiler Nachfage zu einer hohen Schwankungsbreite des Modells gegenüber der echten Daten. Der Grund hierfür liegt in der Abhängigkeit des Modells an der Temperatur. Da die Temperatur einen nur sehr geringen Einfluss auf die Nachfrage in den Sommermonaten hat, kommt es hier zu einer starken Abweichung vom "echten" Verlauf.

3.2 Modell 4

Bei Betrachtung des Verlaufes der Nachfrage nach der Temperatur erkennt man einen stark negativ linearen Zusammenhang unterhalb der Temperatur von 18řC. Der Vorschlag für ein verbessertes Modell liegt somit auf der Hand als ein Modell, welches aus zwei linearen Funktionen zusammengesetzt wird. Diese gelten je nach Jahreszeit für unterschiedliche Bereiche. Dabei werden die Daten geteilt in einen Datensatz der alle Daten vom 1. Oktober 2007 bis 25. April 2008 und von 15. Oktober 2008 bis 10.Oktober 2008 zusammenfasst und ein Datensatz der die Daten von 26. April 2008 bis 14. Oktober 2008 zusammenfasst.

Je nach Zeit gilt entweder y_t^W für den Winter und y_t^S für den Sommer.

$$T \subseteq [-\infty, 18] \rightarrow y_t^W = b_0 + b_1 * T$$

$$T \subseteq]18, \infty] \rightarrow y_t^S = b_2 + b_3 * T$$

[224]:
```
temp = 21
df_winter = df_waermenachfrage[df_waermenachfrage[Temp] <= temp]
```

```
df_sommer = df_waermenachfrage[df_waermenachfrage[Temp] > temp]
lin_regression_winter = stats.linregress(df_winter[Temp].to_numpy(),␣
↪df_winter[Nachfrage].to_numpy())
lin_regression_sommer = stats.linregress(df_sommer[Temp].to_numpy(),␣
↪df_sommer[Nachfrage].to_numpy())
display(Latex(f"$b_{0}$: {lin_regression_winter.intercept:.6f}"))
display(Latex(f"$b_{1}$: {lin_regression_winter.slope:.6f}"))
display(Latex(f"$R^2$: {lin_regression_winter.rvalue**2:.6f}"))
display(Latex(f"$b_{2}$: {lin_regression_sommer.intercept:.6f}"))
display(Latex(f"$b_{3}$: {lin_regression_sommer.slope:.6f}"))
display(Latex(f"$R^2$: {lin_regression_sommer.rvalue**2:.6f}"))
```

b_0: 28.175870
b_1: -1.330530
R^2: 0.837825
b_2: 5.453186
b_3: -0.062266
R^2: 0.124462

Für die kalte Jahreszeit gilt:

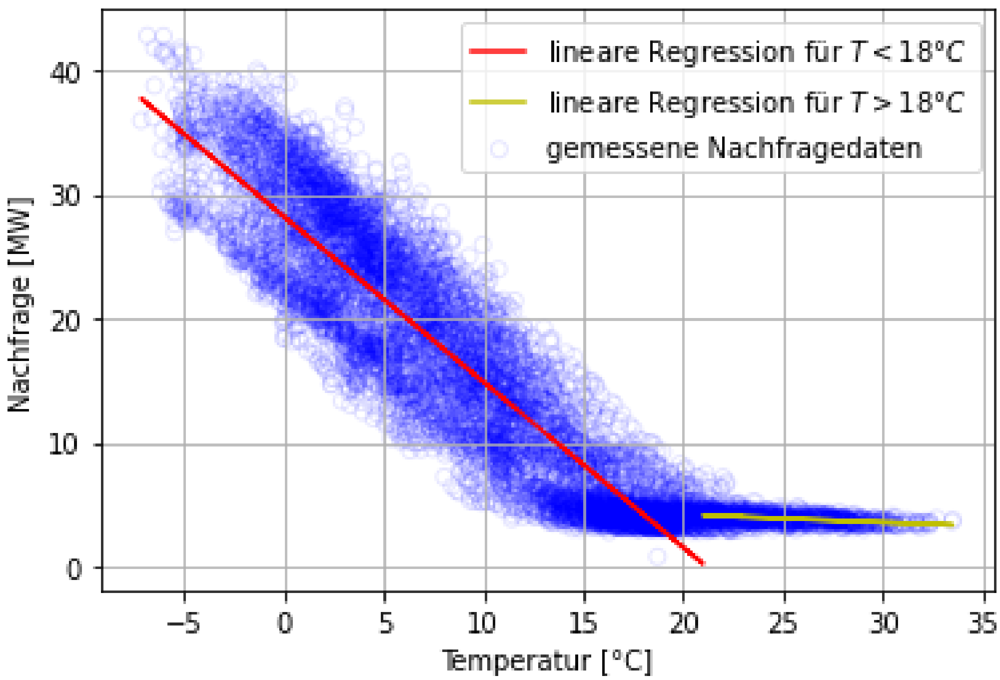
$$y_t^W = 28.542 - 1.416 \cdot T$$

bei einem BestimmtheitsmaSS von 0.809 Für die warme Jahreszeit gilt:

$$y_t^S = 5.628 - 0.069 \cdot T$$

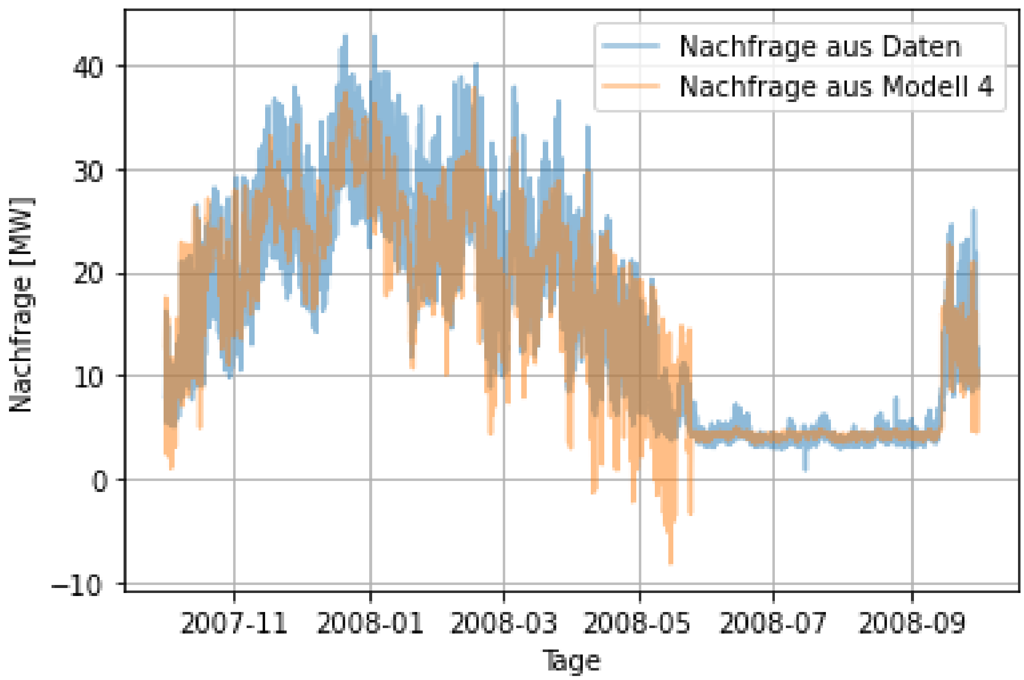
bei einem BestimmtheitsmaSS von 0.073411

[225]:
```
plt.scatter(df_waermenachfrage[Temp], df_waermenachfrage[Nachfrage], alpha=0.1,␣
↪facecolors=none, edgecolors=b, label= "gemessene Nachfragedaten")
plt.xlabel("Temperatur [řC]")
plt.ylabel("Nachfrage [MW]")
plt.plot(df_winter[Temp], lin_regression_winter.intercept +␣
↪lin_regression_winter.slope * df_winter[Temp], r, label=lineare Regression␣
↪für $T<18řC$)
plt.plot(df_sommer[Temp], lin_regression_sommer.intercept +␣
↪lin_regression_sommer.slope * df_sommer[Temp], y, label=lineare Regression␣
↪für $T>18řC$)
plt.legend()
plt.grid()
```



Der Verlauf der Nachfrage über alle Tage eines Jahres wird somit modelliert zu:

[226]:
```
df_waermenachfrage[Tage_rel] = np.arange(0,df_waermenachfrage.shape[0])
filter1 = df_waermenachfrage[Zeit] < pd.Timestamp(2008-5-25)
filter2 = df_waermenachfrage[Zeit] > pd.Timestamp(2008-09-15)
df_waermenachfrage[Nachfrage_modell] = 0
df_waermenachfrage[Nachfrage_modell].where(filter1 |␣
↪filter2,lin_regression_sommer.intercept + lin_regression_sommer.slope *␣
↪df_waermenachfrage[Temp],
                                              inplace = True)
df_waermenachfrage[Nachfrage_modell].where((filter1 | filter2)==␣
↪False,lin_regression_winter.intercept + lin_regression_winter.slope *␣
↪df_waermenachfrage[Temp],
                                             inplace = True)
x = df_waermenachfrage[Zeit].to_numpy()
y_1 = df_waermenachfrage[Nachfrage].to_numpy()
y_2 = df_waermenachfrage[Nachfrage_modell].to_numpy()
plt.plot(x , y_1, alpha=0.5, label = Nachfrage aus Daten)
plt.plot(x , y_2, alpha=0.5, label = Nachfrage aus Modell 4)
plt.xlabel("Tage")
plt.ylabel("Nachfrage [MW]")
plt.legend()
plt.grid()
```



3.3 Interpretation

Wie man anhand der Abbildung erkennen kann, bildet das Modell 3 die Daten besser ab als Modell 2. Besonders in den Sommermonaten ist eine Verbesserung zu erkennen. Aufgrund der so gut wie fehlenden Temperaturabhängigkeit kann dieser Wert besser mit einer Konstante und einer sehr schwachen Temperaturabhängigkeit vorhergesagt werden.