WS 2022/23
Statistische Modellierung
Dozent: Dr. Steffen Pielström
08. Dezember 2022
Corinna Keupp

Polynomiale Regression

- Wird für Daten verwendet, die eine nicht-lineare Beziehung aufweisen
- Auch hier wird die Methode der kleinsten Quadrate angewendet
- Folgen folgender Form:

$$\hat{y} = b_n x^n + \cdots + b_3 x^3 + b_2 x^2 + b_1 x + a$$

Der höchste Exponent bestimmt den Grad (auch Degree) der Funktion, dieser hat Einfluss auf die Eigenschaften (wie das Aussehen) der Funktion. Der „beste" Degree ist von den Daten abhängig und kann nicht allgemein vorgegeben werden.

Herleitung:

(nach Bortz)

- Anhand von einer quadratischen Funktion –> Annahme einer Polynomial Funktion 2. Grades

$$\hat{y} = a + b_1 \cdot x + b_2 \cdot x^2.$$

- Summe der Residuen minimieren

$$\sum_{i=1}^{n} \left[y_i - (a + b_1 \cdot x_i + b_2 \cdot x_i^2) \right]^2$$

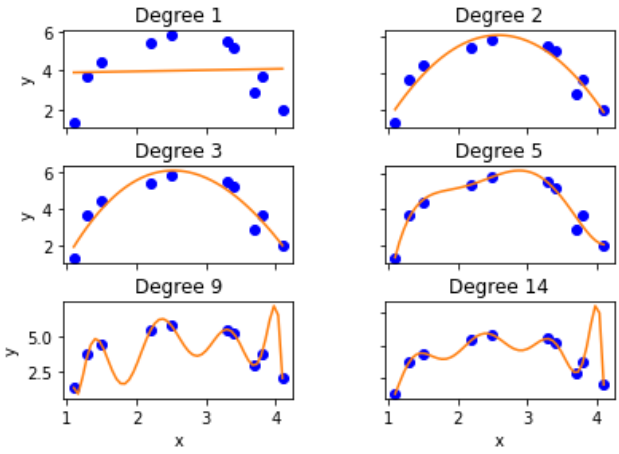
- Daraus folgt nach einigen mathematischen Operationen (partiellen Ableitungen und Nullsetzung dieser)

$$\begin{aligned} \sum_i y_i &= an &&+ b_1 \sum_i x_i + b_2 \sum_i x_i^2, \\ \sum_i x_i \cdot y_i &= a \sum_i x_i &&+ b_1 \sum_i x_i^2 + b_2 \sum_i x_i^3, \\ \sum_i x_i^2 \cdot y_i &= a \sum_i x_i^2 &&+ b_1 \sum_i x_i^3 + b_2 \sum_i x_i^4. \end{aligned}$$

- Dies entspricht somit einem linearen Gleichungssystem, welches durch verschiedene Lösungswege aufgelöst werden kann, bspw. Additionsverfahren oder das Gauß-Verfahren
- Dieses Prinzip kann auf Polynome höheren Grades ebenfalls angewendet werden.

Frage nach Degree

- Problem Overfitting



Umsetzung Python

```
#data
x = np.array([0.0, 1.0, 2.0, 3.0, 4.0, 5.0])
y = np.array([0.0, 0.8, 0.9, 0.1, -0.8, -1.0])
```

##Umsetzung mit numpy

```
import numpy as np

#grad der Funktion
degree = 3

#für die Summary benötigt
df = pd.DataFrame(columns=['y', 'x'])
df['x'] = x
df['y'] = y

weights = np.polyfit(x, y, degree)
model = np.poly1d(weights)

#Ergebnisse
results = smf.ols(formula='y ~ model(x)', data=df).fit()
print(results.summary())

#Visualisierung
plt.scatter(x, y, color = 'blue')
```

```
xModel = np.linspace(min(x), max(x))
yModel = np.polyval(weights, xModel)
plt.plot(xModel, yModel)
plt.title('Polynomial Regression')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
```

```
##Umsetzung mit SKLearn
```

```
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures

#data
x = np.array([0.0, 1.0, 2.0, 3.0, 4.0, 5.0])
y = np.array([0.0, 0.8, 0.9, 0.1, -0.8, -1.0])

poly = PolynomialFeatures(degree = 3)
X_poly = poly.fit_transform(x)
X_vals = np.linspace(min(x), max(x))
lin2 = LinearRegression()
lin2.fit(X_poly, y)
X_vals_poly = poly.transform(X_vals)
y_vals = lin2.predict(X_vals_poly)

#Ergebnisse
model = sm.OLS(y, X_poly).fit()
print(model.summary())

#Visualisierung
plt.scatter(x, y)
plt.plot(X_vals, y_vals)
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```

```
praxis-Daten: df1 =
pd.read_csv(r"https://raw.githubusercontent.com/ck282/statistic0812/main/salaries.t
xt", sep="\t")
```

Hausaufgabe:

Daten Aufgabe 1: https://raw.githubusercontent.com/ck282/statistic0812/main/salaries.txt

- A) Plotte die Daten mit einer polynomialen Regression mit einem Grad von 3. Benutze dabei entweder numpy oder sklearn.
- B) Was ist die Funktionsgleichung?
- C) Treffe eine Vorhersage für einen x-Wert von 6.5
- D) Wähle eine bereits behandelte Modellierung aus und vergleiche, ob diese besser oder schlechter auf die Daten passt.

Daten Aufgabe 2: https://raw.githubusercontent.com/ck282/statistic0812/main/values.txt

- Lade die Daten ein, führe eine polynomiale Regression durch und finde heraus welcher Grad/Degree für die Daten am besten geeignet ist, begründe dies.

Quellen:

- Bortz, Jürgen. Statistik: Für Human- und Sozialwissenschaftler (Springer Lehrbuch). 6., Vollst. überarb. u. aktualisierte, Springer, 2022.
- Jackson, Dr. S. Chapter 7 Polynomial Regression | Machine Learning. 6. April 2022, bookdown.org/ssjackson300/Machine-Learning-Lecture-Notes/polynomial-regression.html.
- Loong, Joshua. „Fitting Polynomial Regressions in Python“. Joshua Loong, 3. Oktober 2018, joshualoong.com/2018/10/03/Fitting-Polynomial-Regressions-in-Python.
- Polynomial Regression - which python package to use? https://zerowithdot.com/polynomial-regression-in-python/
- Python Machine Learning Polynomial Regression. www.w3schools.com/python/python_ml_polynomial_regression.asp.