Lineær regression i Python

Introduktion til Lineær Regression i Python

Henrik Sterner

Introduktion til Lineær Regression i Python

Af Henrik Sterner (henrik.sterner@gmail.com)

Lineær regression er en metode til at finde en lineær funktion, der passer til et datasæt. Den lineære funktion kan bruges til at forudsige værdier for nye data.

Et eksempel på lineær regression: Verdensrekord i 100 meter løb

Herunder er vist tiderne for de 100 meter løb, der er blevet sat i OL siden 1896, indlæst i python. Tiderne er i sekunder.

```python
import numpy as np
import matplotlib.pyplot as plt

years = np.array([1896, 1900, 1904, 1908, 1912, 1920, 1924,

times = np.array([12.0, 11.0, 11.0, 10.8, 10.8, 10.8, 10.6,
```

Et eksempel på lineær regression: Hvad er verdensrekorden i 100 meter løb i 2100?

Vi kan bruge lineær regression til at forudsige, hvad verdensrekorden i 100 meter løb vil være i 2100. Vi kan gøre det ved at finde en lineær funktion, der passer til datasættet, og så bruge den til at forudsige tiden for 2100. Herunder viser vi det i python og ved brug af `LinearRegression`-klassen fra `sklearn.linear_model`-modulet.:

````
```python
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(years.reshape(-1, 1), times)

print(model.predict([[2100]]))
```

```
[9.58636364]
````
````

Et eksempel på lineær regression: Hvad er verdensrekorden i 100 meter løb i 2200 og 2300?

Herunder viser vi det i python og ved brug af `LinearRegression`-klassen fra `sklearn.linear_model`-modulet.:

```python
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(years.reshape(-1, 1), times)

print(model.predict([[2200], [2300]]))
```

```
[8.38636364 7.24636364]
```

Hvad er problemet med at bruge lineær regression til at forudsige verdensrekorden i 100 meter løb i 2200 og 2300?

Matematikken bag lineær regression i to variable

Lad os antage, at vi har et datasæt med n datapunkter. Hvert datapunkt har en x-værdi og en y-værdi. Vi kan repræsentere datasættet som en liste af n par (x_i, y_i), hvor x_i er x-værdien for det i'te datapunkt, og y_i er y-værdien for det i'te datapunkt. Da handler lineær regression om at finde en lineær funktion $f(x) = ax + b$, der passer til datasættet. Funktionen $f(x)$ kaldes en lineær model. Lineær regression handler om at finde værdierne af a og b, der gør, at lineær modellen passer bedst muligt til datasættet.

Matematikken bag lineær regression i to variable: Hvordan måler vi, hvor godt en lineær model passer til et datasæt?

Vi kan måle, hvor godt en lineær model passer til et datasæt, ved at måle, hvor langt datapunkterne er fra modellen. Hvis datapunkterne er langt fra modellen, passer modellen ikke godt til datasættet. Hvis datapunkterne er tæt på modellen, passer modellen godt til datasættet.

Denne metode kaldes mindste kvadraters metode.

Mindste kvadraters metode

Mindste kvadraters metode går ud på at finde den lineære model, der gør, at summen af kvadraterne på afstandene fra datapunkterne til modellen er mindst mulig. Vi postulerer formlerne herunder. Lad gennemsnittet af x-værdierne være $\bar{x}$, og lad gennemsnittet af y-værdierne være $\bar{y}$. Vi kan udregne $\bar{x}$ og $\bar{y}$ således:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}, \bar{y} = \frac{\sum_{i=1}^{n} y_i}{n}$$

Da er

$$a = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$b = \bar{y} - a\bar{x}$$

Udledning af formlerne for a og b

Vi kan udlede formlerne for a og b ved at - differentiere summen af kvadraterne på afstandene fra datapunkterne til modellen med hensyn til a og b. - sætte de afledte lig med 0, og løs de to ligninger med hensyn til a og b.

Vi får en formel for a og en formel for b, der indsættes i modellen $f(x) = ax + b$. Vi har nu en lineær model, der passer til datasættet.

Udledning af formlerne for a og b

Lad $(x1, y1), (x2, y2), ..., (xn, yn)$ være datapunkterne. Residualet for det i'te datapunkt er afstanden fra datapunktet til modellen:

$$r_i = y_i - (ax_i + b),$$

hvor y_i er den faktiske y-værdi for det i'te datapunkt, og $ax_i + b$ er den forudsagte y-værdi for det i'te datapunkt, mens r_i kan betragtes som fejlen for det i'te datapunkt.

Dernæst defineres summen af kvadraterne på afstandene fra datapunkterne til modellen:

$$\sum_{i=1}^{n}(y_i - (ax_i + b))^2 = r_i^2$$

Vi skal nu minimere denne sum ved at differentiere den med hensyn til a og b.

Udled udtrykkene for a og b - del 1

Differentier summen af kvadraterne på afstandene fra datapunkterne til modellen med hensyn til a. Dvs. vi differentierer summen af kvadraterne på afstandene fra datapunkterne til modellen med hensyn til a:

$$\frac{\partial}{\partial a}\sum_{i=1}^{n}(y_i-(ax_i+b))^2=0$$

Udled udtrykkene for a og b - del 2

Før vi differentierer, kan vi gange summen ud:

$$\sum_{i=1}^{n}(y_i - (ax_i + b))^2 = \sum_{i=1}^{n}(y_i^2 - 2y_i(ax_i + b) + (ax_i + b)^2)$$

og tilsvarende med sidste led:

$$\sum_{i=1}^{n}(y_i - (ax_i + b))^2 = \sum_{i=1}^{n}(y_i^2 - 2y_i(ax_i + b) + a^2x_i^2 + 2abx_i + b^2)$$

Udled udtrykkene for *a* og *b* - del 3

Dernæst differentierer vi summen med hensyn til a:

$$\sum_{i=1}^{n}(y_i^2 - 2y_i(ax_i + b) + a^2x_i^2 + 2abx_i + b^2)$$

$$= \sum_{i=1}^{n}(-2y_ix_i + 2ax_i^2 + 2bx_i)$$

Sættes den afledte lig med 0, får vi:

$$\sum_{i=1}^{n}(-2y_ix_i + 2ax_i^2 + 2bx_i) = 0$$

Udled udtrykkene for a og b - del 4

Dernæst summes over hvert led:

$$\sum_{i=1}^{n} -2y_i x_i + \sum_{i=1}^{n} 2ax_i^2 + \sum_{i=1}^{n} 2bx_i = 0$$

Vi isolerer leddet med a:

$$\sum_{i=1}^{n} 2ax_i^2 = -\sum_{i=1}^{n} -2y_i x_i - \sum_{i=1}^{n} 2bx_i$$

Udled udtrykkene for a og b - del 5

og trækkker a ud af summen:

$$2a\sum_{i=1}^{n} x_i^2 = -\sum_{i=1}^{n} -2y_i x_i - \sum_{i=1}^{n} 2bx_i$$

Til sidst isolerer vi a:

$$a = \frac{\sum_{i=1}^{n} y_i x_i - b\sum_{i=1}^{n} x_i}{\sum_{i=1}^{n} x_i^2}$$

Udled udtrykkene for a og b - del 6

På tilsvarende vis kan vi udlede udtrykket for b, og vi får:

$$b = \frac{\sum_{i=1}^{n} -y_i + a\sum_{i=1}^{n} x_i}{-n}$$

Ved at indføre gennemsnittet af x-værdierne $\bar{x}$ og gennemsnittet af y-værdierne $\bar{y}$, som er givet ved:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}, \bar{y} = \frac{\sum_{i=1}^{n} y_i}{n}$$

Udled udtrykkene for a og b - del 7

Kan vi endelige skrive formlerne for a og b på følgende måde:

$$a = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$b = \bar{y} - a\bar{x}$$

Funktioner der implementerer udtrykkene for *a* og *b*

Vi kan let implementere udtrykkene for *a* og *b* i python:

```
def a(x, y):
    return (np.sum(y * x) - b(x, y) * np.sum(x)) / np.sum(x

def b(x, y):
    return (np.sum(-y) + a(x, y) * np.sum(x)) / -len(x)
```

Afprøve funktionerne der implementerer udtrykkene for a og b

Vi kan afprøve funktionerne der implementerer udtrykkene for a og b i python på vores datasæt med verdensrekorder i 100 meter løb:

```python
print("a= ", a(years, times))
print("b= ", b(years, times))
```

```bash
a=  -0.013330885952031327
b=  36.41645590250286
```

Lineær regression med flere variable

Vi kan også bruge lineær regression til at finde en lineær model, der passer til datasæt med flere variable. Hvis vi har et datasæt med n datapunkter, hvor hvert datapunkt har m variable, kan vi repræsentere datasættet som en liste af n vektorer med m elementer. Hvert datapunkt er en vektor med m elementer:

$$\vec{x_i} = (x_{i1}, x_{i2}, ..., x_{im})$$

hvor x_{ij} er værdien af den j'te variable for det i'te datapunkt.

Eksempel på lineær regression med flere variable: Vægt, højde og skostørrelse

Herunder er vist vægt, højde og skostørrelse for 10 personer, indlæst i python. Vægt er i kg, højde er i cm, og skostørrelse er i EU-størrelse.

```python
import numpy as np
import matplotlib.pyplot as plt

weights = np.array([70, 80, 100, 60, 90, 80, 70, 100, 60, 9
heights = np.array([180, 175, 190, 170, 185, 175, 180, 190,
shoe_sizes = np.array([43, 44, 46, 42, 45, 44, 43, 46, 42,
```

Kan vi ud fra vægt og højde forudsige skostørrelsen?

Eksempel på lineær regression med flere variable: Vægt, højde og skostørrelse

Til at forudsige skostørrelsen bruger vi en lineær model med to variable, vægt og højde:

$$f(\vec{x}) = a_1 x_1 + a_2 x_2 + b$$

hvor x_1 er vægt, x_2 er højde, a_1 er "vægtningen" af vægt, a_2 er "vægtningen" af højde, og b er skostørrelsen, når vægt og højde er 0. Funktionen udtrykker altså, at skostørrelsen afhænger af vægt og højde.

Eksempel på lineær regression med flere variable: Vægt, højde og skostørrelse

Vi kan bruge lineær regression til at finde værdierne af a_1, a_2 og b, der gør, at lineær modellen passer bedst muligt til datasættet. Herunder formlerne for a_1, a_2 og b:

$$a_1 = \frac{\sum_{i=1}^{n} y_i x_{i1} - \frac{\sum_{i=1}^{n} -y_i + a_1 \sum_{i=1}^{n} x_{i1} + a_2 \sum_{i=1}^{n} x_{i2}}{-n} \sum_{i=1}^{n} x_{i1}}{\sum_{i=1}^{n} x_{i1}^2}$$

$$a_2 = \frac{\sum_{i=1}^{n} y_i x_{i2} - \frac{\sum_{i=1}^{n} -y_i + a_1 \sum_{i=1}^{n} x_{i1} + a_2 \sum_{i=1}^{n} x_{i2}}{-n} \sum_{i=1}^{n} x_{i2}}{\sum_{i=1}^{n} x_{i2}^2}$$

$$b = \frac{\sum_{i=1}^{n} -y_i + a_1 \sum_{i=1}^{n} x_{i1} + a_2 \sum_{i=1}^{n} x_{i2}}{-n}$$

Udregn formlerne for a_1, a_2 og b i python

Vi kan implementere formlerne for a_1, a_2 og b i python:

```
def a1(x1, x2, y):
    return (np.sum(y * x1) - b(x1, x2, y) * np.sum(x1)) / n

def a2(x1, x2, y):
    return (np.sum(y * x2) - b(x1, x2, y) * np.sum(x2)) / n

def b(x1, x2, y):
    return (np.sum(-y) + a1(x1, x2, y) * np.sum(x1) + a2(x1
```

Afprøv formlerne for a_1, a_2 og b i python

Vi kan afprøve formlerne for a_1, a_2 og b i python på vores datasæt med vægt, højde og skostørrelse:

```
print("a1= ", a1(weights, heights, shoe_sizes))
print("a2= ", a2(weights, heights, shoe_sizes))
print("b= ", b(weights, heights, shoe_sizes))
```

```
```bash
a1=  0.5
a2=  0.1
b=  35.0
```
```

Definer en funktion, der forudsiger skostørrelsen ud fra vægt og højde

Vi kan definere en funktion, der forudsiger skostørrelsen ud fra vægt og højde i python:

```
def f(x1, x2):
    return a1(weights, heights, shoe_sizes) * x1 + a2(weigh
```

Afprøv funktionen, der forudsiger skostørrelsen ud fra vægt og højde

Vi kan afprøve funktionen, der forudsiger skostørrelsen ud fra vægt og højde i python. Her forudsiger vi skostørrelsen for en person, der vejer 80 kg og er 180 cm høj:

```python
print(f(80, 180))
```

````
```bash
43.0
```
````

Lineær regression med n variable

Vi kan også bruge lineær regression til at finde en lineær model, der passer til datasæt med n variable. Hvis vi har et datasæt med n datapunkter, hvor hvert datapunkt har m variable, kan vi repræsentere datasættet som en liste af n vektorer med m elementer. Hvert datapunkt er en vektor med m elementer:

$$\vec{x_i} = (x_{i1}, x_{i2}, ..., x_{im})$$

hvor x_{ij} er værdien af den j'te variable for det i'te datapunkt.

Lineær regression med n variable på matrix form

Vi kan repræsentere datasættet som en matrix med n rækker og m kolonner:

$$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}$$

Funktionsudtrykket for en lineær model med m variable:

$$f(\vec{x}) = a_1 x_1 + a_2 x_2 + \cdots + a_m x_m + b,$$

hvor x_1 er den første variable, x_2 er den anden variable, ..., x_m er den m'te variable, a_1 er "vægtningen" af den første variable, a_2 er "vægtningen" af den anden variable, ..., a_m er "vægtningen" af den m'te variable, og b er værdien af den lineære model, når alle variable er 0.

Find udtrykkene for $a_1, a_2, \ldots, a_m$ og b

Vi kan bruge mindste kvadraters metode til at finde udtrykkene for $a_1, a_2, \ldots, a_m$ og b:

$$a_1 = \frac{\sum_{i=1}^n y_i x_{i1} - \frac{\sum_{i=1}^n -y_i + a_1 \sum_{i=1}^n x_{i1} + a_2 \sum_{i=1}^n x_{i2} + \cdots + a_m \sum_{i=1}^n x_{im}}{-n} \sum_{i=1}^n x_{i1}}{\sum_{i=1}^n x_{i1}^2}$$

$$a_2 = \frac{\sum_{i=1}^n y_i x_{i2} - \frac{\sum_{i=1}^n -y_i + a_1 \sum_{i=1}^n x_{i1} + a_2 \sum_{i=1}^n x_{i2} + \cdots + a_m \sum_{i=1}^n x_{im}}{-n} \sum_{i=1}^n x_{i2}}{\sum_{i=1}^n x_{i2}^2}$$

$$\vdots$$

$$a_m = \frac{\sum_{i=1}^n y_i x_{im} - \frac{\sum_{i=1}^n -y_i + a_1 \sum_{i=1}^n x_{i1} + a_2 \sum_{i=1}^n x_{i2} + \cdots + a_m \sum_{i=1}^n x_{im}}{-n} \sum_{i=1}^n x_{im}}{\sum_{i=1}^n x_{im}^2}$$

Udregn formlerne for $a_1, a_2, \ldots, a_m$ og b i python

Lad X være en matrix med n rækker og m kolonner, hvor hver række er et datapunkt, og hver kolonne er en variabel. Lad y være en vektor med n elementer, hvor hvert element er værdien af den afhængige variable for det tilsvarende datapunkt. Vi kan implementere formlerne for $a_1, a_2, \ldots, a_m$ og b i python:

```
def a(X, y, j):
    return (np.sum(y * X[:, j]) - b(X, y) * np.sum(X[:, j])

def b(X, y):
    return (np.sum(-y) + np.sum(a(X, y, j) * np.sum(X[:, j
```

Konstruktion af funktion for lineær regression med n variable

Til sidst kan vi konstruere en funktion, der finder en lineær model, der passer til et datasæt med n variable:

```
def linear_regression(X, y):
    def f(x):
        return np.sum(a(X, y, j) * x[j] for j in range(X.sh
    return f
```

Test funktionen for lineær regression med n variable

Vi kan teste funktionen for lineær regression med n variable på vores datasæt med vægt, højde og skostørrelse:

~~~
X = np.array([weights, heights]).T
y = shoe_sizes

f = linear_regression(X, y)

print(f([80, 180]))
~~~

~~~
    ```bash
    43.0
    ```
~~~

Lineær regression med n variable i scikit-learn

Vi kan også bruge `LinearRegression`-klassen fra `sklearn.linear_model`-modulet til at finde en lineær model, der passer til et datasæt med *n* variable:

```
from sklearn.linear_model import LinearRegression

X = np.array([weights, heights]).T
y = shoe_sizes

model = LinearRegression()
model.fit(X, y)

print(model.predict([[80, 180]]))

    ```bash
    [43.]
    ```
```

.T betyder, at vi transponerer matricen, så hver række er et datapunkt, og hver kolonne er en variabel.

Svagheder ved lineær regression

- Lineær regression kan kun bruges til at finde en lineær model, der passer til et datasæt. Hvis datasættet ikke er lineært, kan lineær regression ikke bruges.
- Lineær regression kan kun bruges til at forudsige værdier for nye data, der ligger inden for intervallet for de data, der er brugt til at finde den lineære model. Hvis vi bruger lineær regression til at forudsige værdier for nye data, der ligger uden for intervallet for de data, der er brugt til at finde den lineære model, kan vi få forkerte resultater.

Styrker ved lineær regression

- Lineær regression er
- enkel og hurtig at implementere.
- enkel og hurtig at forstå.
- enkel og hurtig at bruge.
- enkel og hurtig at teste.
- godt udgangspunkt for mere avancerede modeller.
- giver gode resultater, hvis datasættet er lineært.