Skalarprodukt

Das **Skalarprodukt** zweier **n-dimensionaler Vektoren** ist definiert als:

$$\mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^{n} a_i b_i$$

Beispiel:

Gegeben seien die Vektoren:

$$\mathbf{a} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix}$$

- **Dimension von $\mathbf{a}$:** (3×1) (Spaltenvektor)
- **Dimension von $\mathbf{b}$:** (3×1) (Spaltenvektor)

Transponieren von $\mathbf{a}$ oder $\mathbf{b}$

Um das Skalarprodukt zu berechnen, müssen wir sicherstellen, dass wir das Skalarprodukt zwischen einem **Zeilenvektor** und einem **Spaltenvektor** haben. Also transponieren wir $\mathbf{a}$ (von einem Spaltenvektor (3×1) zu einem Zeilenvektor (1×3)):

$$\mathbf{a}^\top = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$$

- **Dimension von $\mathbf{a}^\top$:** (1×3) (Zeilenvektor)
- **Dimension von $\mathbf{b}$:** (3×1) (Spaltenvektor)

Schritt 2: Skalarprodukt berechnen

Jetzt, wo wir die Dimensionen korrekt haben, können wir das Skalarprodukt berechnen:

$$\mathbf{a}^\top \mathbf{b} = (1 \cdot 4) + (2 \cdot 5) + (3 \cdot 6)$$

$$= 4 + 10 + 18 = 32$$

Schritt 3: Ergebnis und Dimension des Ergebnisses

Das Ergebnis des Skalarprodukts ist ein **Skalar** (also eine (1×1)-Matrix):

$$\mathbf{a}^\top \mathbf{b} = 32$$

Zusammenfassung der Dimensionen:

1. **Vor der Transposition:**
    - Dimension von $\mathbf{a}$: (3×1)
    - Dimension von $\mathbf{b}$: (3×1)
2. **Nach der Transposition:**
    - Dimension von $\mathbf{a}^\top$: (1×3)
    - Dimension von $\mathbf{b}$: (3×1)
3. **Ergebnis des Skalarprodukts:**
    - Dimension des Ergebnisses: (1×1) (Skalar)

Spezialfall in NumPy (1D-Arrays)

NumPy erlaubt `np.dot(a, b)` auch für 1D-Arrays der Form `(n,)`, weil sie **weder explizite Zeilen- noch Spaltenvektoren** sind, sondern einfach Listen von Zahlen.

In NumPy funktioniert daher:

```
import numpy as np
a = np.array([1, 2, 3])  # (3,)
b = np.array([4, 5, 6])  # (3,)
print(np.dot(a, b))  # 32
```

Falls explizite **Zeilen- oder Spaltenvektoren** genutzt werden, muss man auf die Dimensionen achten:

```
a = np.array([[1, 2, 3]])  # (1,3) Zeilenvektor
b = np.array([[4, 5, 6]])  # (1,3) Zeilenvektor
print(np.dot(a, b.T))  #
```

Matrix-Vektor-Multiplikation

Falls X eine Matrix mit mehreren Zeilen (Samples) ist, kann die Matrix-Vektor-Multiplikation mit einem Gewichtsvektor w berechnet werden. Dabei entspricht jede Zeile von X einem Sample, und das Ergebnis ist ein Vektor mit den Skalarprodukten jeder Zeile mit w:

$$y = Xw$$

Hierbei wird jede Zeile von X mit dem Vektor w als Skalarprodukt berechnet. Das Ergebnis ist ein Vektor y, dessen Einträge die gewichteten Summen der jeweiligen Zeilen von X sind.

```
X = np.array([[1, 2],
              [3, 4],
              [5, 6]])  # Dimension: (3,2)

w = np.array([0.5, -1])  # Dimension: (2,)

result = np.dot(X, w)  # Matrix-Vektor-Produkt

print(result)
# [1*0.5 + 2*(-1), 3*0.5 + 4*(-1), 5*0.5 + 6*(-1)]
# [-1.5, -2.5, -3.5]

print(result.shape)  # (3,)  => Ein Ergebnis pro Zeile
```